WebSphere Portal Express v8.5

Version 8 Release 5

Contents

Chapter 17. Troubleshooting 3537

Chapter 1. Overview

IBM® WebSphere® Portal Express® provides a single access point to web content and applications, while it delivers differentiated, personalized experiences for each user.

Multiple solutions are available to fit your needs.

WebSphere Portal Server

This foundation offering provides a single access point to web content and applications and delivers socially infused, differentiated, personalized experiences for each site visitor. WebSphere Portal Server supports workflows, limited content management, simplified usability and administration, development tools and Web 2.0, open standards, security, and scalability. The result delivers exceptional web experiences across multiple channels, in context with the correct business applications and data, to engage and collaborate effectively, improve business productivity, and deliver better business results.

WebSphere Portal Enable and Extend

These offerings add capability on to the foundation WebSphere Portal Server offering. The Enable and Extend offerings add web content management capabilities. The offerings deliver the correct information to the correct audience. They include rich text editing tools, web content templates, workflow, advanced enterprise search, and personalization services. Used together, these features help you build rich websites that can deliver highly targeted and dynamic content to customers, partners, and employees in more relevant ways.

IBM Web Content Manager and Standard EditionIBM Web Content Manager

IBM Web Content Manager increases efficiency and accuracy of website deployments. This is achieved by placing content creation in the hands of content experts to create, maintain, and deliver online content while IT retains control.

WebSphere Portal Express

WebSphere Portal Express offers application integration, document management, web content management, and collaboration capabilities in a single, easy-to-deploy solution that is targeted at small, and medium businesses. WebSphere Portal Express can help your small and midsize business, or large department, easily deploy and customize websites, achieving faster time to value.

For additional information about WebSphere Portal solutions visit:
http://www.ibm.com/software/websphere/portal/

"What's new" on page 2
IBM WebSphere Portal Express provides new features for administrators, developers, and content authors. Start with Version 8.5 to leverage these new features on your site and then continue to get fixes, improvements, and new features with the latest cumulative fix. Use this section to get the quick highlights on these features and improvements.

"Product capabilities" on page 21
IBM WebSphere Portal Express capabilities let you quickly implement web experiences that are engaging, flexible, and high performing.

"Types of websites" on page 45
Different types of websites require different solutions and use different applications and features. These examples describe details of different websites and the types of applications and features that are required to deliver them.

"Accessibility features" on page 51
Accessibility features help users who have a physical disability, such as restricted mobility or limited vision, to use software products successfully.

What's new

IBM WebSphere Portal Express provides new features for administrators, developers, and content authors. Start with Version 8.5 to leverage these new features on your site and then continue to get fixes, improvements, and new features with the latest cumulative fix. Use this section to get the quick highlights on these features and improvements.

"What's new in the combined cumulative fixes"
Learn what's new in the IBM WebSphere Portal Express Version 8.5 combined cumulative fixes.

"What's new in V8.5" on page 10
Learn what's new in IBM WebSphere Portal Express Version 8.5 .

What's new in the combined cumulative fixes

Learn what's new in the IBM WebSphere Portal Express Version 8.5 combined cumulative fixes.

CF07 "What's new with CF07"
This Combined Cumulative Fix includes new features and improvements.

CF06 "What's new with CF06" on page 4
This Combined Cumulative Fix includes new features and improvements.

CF05 "What's new with CF05" on page 6
This Combined Cumulative Fix includes new features and improvements.

CF04 "What's new with CF04" on page 7
This Combined Cumulative Fix includes new features and improvements.

CF03 "What's new with CF03" on page 8
This Combined Cumulative Fix includes new features and improvements.

What's new with CF07

This Combined Cumulative Fix includes new features and improvements.

Some features are ready for immediate use when you apply the latest combined cumulative fix. Other features require extra configuration or a procedure to enable the feature.

Documentation resource: "Roadmap: Applying maintenance" on page 88

Updates to Dojo 1.9 modules

Use new theme modules to write Dojo dijits. The Dojo 1.9 Claro Theme Artifacts module, **dijit_theme_basic_19**, contains basic styles to render dijits. The Dojo 1.9 Basic Theme Artifacts module, **dijit_theme_claro_19**, contains the Claro theme from Dojo.

Documentation resource: “Modules that are provided with the modularized theme” on page 2608

Worklight is now IBM Worklight®

MobileFirst replaces Worklight to provide multi-channel support to your web communities and is now supported by the v7.0 server.

Documentation resource: “Integrating with IBM MobileFirst” on page 1002

Improved manual syndication options

Manual syndication has been improved to give users more choice when updating a syndication relationship to include a **Rebuild with mirror** option. If you select the mirror option, all items on the subscriber are reset to mirror the syndicator. All items that are newer on the syndicator are sent to the subscriber. Items that are newer on the subscriber are overwritten with the older version from the syndicator. Items that are created on the subscriber that do not exist on the syndicator are removed from the subscriber. Version history is not syndicated.

Documentation resource: Manually syndicating items

Page template mappings

The selection of a presentation template is now optional when creating template mappings for Page templates:

Documentation resource: Defining page item properties

The ML Translations text element is now hidden on forms

The **ML Translations** text element is displayed in the content item form only if the Multilingual Solution is enabled. Otherwise, this element is hidden in the content item form, but still visible in the Manage Elements dialog.

Documentation resource: “Edit-time navigation creation extension” on page 2464

Text provider strings can now be stored in content items and site areas

Previously, the strings used by the text provider plug-in had to be stored in a custom plug-in. Now they can be stored in content items and site areas, allowing users to create and maintain text provider strings from within Web Content Manager.

Documentation resource: “How to store translated text in a content item or site area” on page 1848

Additional rendering modes for web content

Additional rendering modes for web content have been added to allow users to define separate presentation templates for different media, such as JSON, XML, HTML, plus a summary mode.

Documentation resource: “Rendering modes for web content” on page 2093

Web content tag behavior and enhancements

Web content tag enhancements have been added to simplify and improve tag creation and usage.

Documentation resource: "Web content tag behavior and enhancements" on page 1833

Project template updates

You can now select a project template as the default project template. You can now hide the **Editors and viewers** selection tool within the **Publish Options**, **Approval Settings**, and **Custom Action** sections of the project form.

Documentation resource: Creating project templates

What's new with CF06

This Combined Cumulative Fix includes new features and improvements.

Some features are ready for immediate use when you apply the latest combined cumulative fix. Other features require extra configuration or a procedure to enable the feature.

Documentation resource: "Roadmap: Applying maintenance" on page 88

Configuration wizard supports database transfer to multiple databases for Oracle

Use the Database Transfer configuration option using the configuration wizard to transfer your Oracle database to multiple databases or schemas. The multiple databases or schemas option can configure multiple databases with different data sources and different users per schema. You can also use the multiple database option to configure a single database with different data sources and different users.

Documentation resource: Database Management Systems; then go to **Oracle: Database transfer** > **Oracle: Oracle worksheet: Transfer to multiple databases**.

Improving page loading performance with asynchronous web content rendering

You can now increase page loading performance by separating portal page content delivery from web content rendering.

Documentation resource: "Improving page loading performance with asynchronous web content rendering" on page 2065

Integrating remote JSON data by using the Digital Data Connector

Starting with CF06, the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework provides a generic JSON DDC plug-in that is ready to use for integrating external JSON data of your choice. You can use this plug-in to render external JSON data on your portal pages without having to write custom Java code.

Documentation resource: "Integrating remote JSON data" on page 3286

Web Application Bridge support for SAML

Starting with CF06, the Web Application Bridge now supports single sign-on using Security Assertion Markup Language (SAML).

Documentation resource: Content provider policy single sign-on

WSRP

You can now configure the WSRP Consumer to invalidate the remote session when a user explicitly logs out of the Consumer portal.

Documentation resource: "Configuring remote session invalidation" on page 1493

Updates to the Web Content Manager REST service

The Web Content Manager REST service now supports creating and updating authoring templates.

Documentation resource: "How to use REST with authoring templates" on page 3337

Updates to image and file elements

Image components and file components can now be selected when using image elements in content items and site areas:

Documentation resource: Adding a file resource element to an item

Documentation resource: Adding a file resource element to a template

Documentation resource: Adding an image element to an item

Documentation resource: Adding an image element to a template

Updates to custom search box in the portal theme

You can now redirect search requests issued by a custom search form to the Search Center.

Documentation resource: "Redirecting search requests from a custom search form to the Search Center" on page 645

OneUI is deprecated

OneUI styles were deprecated in the theme profiles of the default Portal 8.5 theme in this combined cumulative fix. It simplifies theme development and CSS content by excluding OneUI styles from the default theme. Core components that are embedded in your customer theme, such as the Web Content Viewer or inline editing, do not depend on OneUI styles any more. However, existing themes and applications continue to work with the OneUI theme. Some static resources were replaced with new `.css` and `.js` files. You might also need to remove old social rendering libraries and deploy the new ones.

Documentation resource: "Static resources" on page 2522

Documentation resource: "Removing the previous version of social rendering" on page 2135

Documentation resource: "Enabling social rendering in a virtual portal" on page 2133

Document Services feeds for Federated Documents

Support for Document Service feeds for Federated Documents are now disabled by default, but can be enabled using the `wp.federated.documents.document.services.enabled` configuration setting.

Documentation resource: “Configuring the federated documents feature” on page 407

The textbox.io feature for Ephox is now supported

The textbox.io feature for Ephox is now supported. For further information, and to obtain the application, visit the Ephox website.

What's new with CF05

This Combined Cumulative Fix includes new features and improvements.

Some features are ready for immediate use when you apply the latest combined cumulative fix. Other features require extra configuration or a procedure to enable the feature.

Documentation resource: “Roadmap: Applying maintenance” on page 88

Updating the content menu to open on click instead of on hover

This feature allows you to change the behavior of the content menu.

Documentation resource: “Updating the content menu to open on click instead of on hover” on page 2783

IBM UX Screen Flow Manager

Screen Flow Manager works like a wizard and uses screens to represent steps for a task. Starting with CF05, Screen Flow Manager provides you with several options to configure transitions between portlets, pages, and dialogs within a screen flow. The Screen Flow Manager also provides the dialog chaining and nesting option. You can configure another dialog to start after the current dialog ends or after the current dialog pauses. For example, in a travel site, you can configure the Car booking dialog to start after the Flight booking dialog ends or to start by pausing the Flight booking dialog.

Documentation resource: “Transitions” on page 3487

Documentation resource: “Dialog chaining and nesting” on page 3494

Project publish and validation updates

The process of publishing and validating projects is improved in CF05 to make it easier to monitor and manage projects.

Documentation resource: “Project Publishing” on page 1925

Documentation resource: “Project Validation” on page 1927

Folder updates

Folders can now be workflowed, and added to projects.

Syndication modes

The syndication user interface has been updated to allow for the selection of different modes of syndication.

Documentation resource: Creating a syndication relationship

SmartCloud for Social Business integration by using Active Directory Federation Services (ADFS)

Integration between WebSphere Portal Express and IBM Connections in SmartCloud for Social Business can now be completed by using Active Directory Federation Services (ADFS), which supports single sign-on. Integration with Tivoli Federated Identity Manager (TFIM) is still supported.

Documentation resource: "Configuration settings for Active Directory Federation Services (ADFS)" on page 3048

Searching in a multilingual environment

If your site is multilingual, you can now enable search for users in languages other than their preferred language.

Documentation resource: "Configuring search for multilingual sites" on page 684

Content as a Service pages

Use Content as a Service pages to render content that is managed by your IBM Web Content Manager in different data formats such as JSON or XML. Content as a Service pages allow the content that is centrally authored and maintained on your website, to be accessed by the other data clients in raw data formats.

Documentation resource: "Content as a Service pages" on page 1881

WSRP

The WSRP Consumer markup caching feature now offers better performance. You can now use WSRP markup caching without enabling the portlet container fragment caching. With a new configuration parameter, you can enable and disable markup caching specifically for selected remote portlets or for all remote portlets of a Consumer portal.

The WSRP Consumer provides multiple new configuration parameters for defining two-phase rendering behavior, WSRP response timeouts, and a limit for the size of file uploads.

Documentation resource: "WSRP Markup Caching" on page 1499

Updating the content menu to open on click instead of on hover

The content menu opens when a user hovers over a portlet containing IBM Web Content Manager items. In Combined Cumulative Fix 05, you can set the content menu to open when a user clicks an icon in the portlet skin instead.

Documentation resource: Updating the content menu to open on click instead of on hover

What's new with CF04

This Combined Cumulative Fix includes new features and improvements.

Some features are ready for immediate use when you apply the latest combined cumulative fix. Other features require extra configuration or a procedure to enable the feature.

Documentation resource: "Roadmap: Applying maintenance" on page 88

Validation improvements for the Configuration Wizard

Prevent a possible database transfer failure by validating your entries in the wizard. When you chose to validate settings for the Database Transfer option, field syntax and database connection validations are performed before you run the configuration. Examples of fields that are validated can include (depending on your database type) host name and port number.

Renditions

Renditions are different versions of an image component or element. Renditions can be thumbnails or smaller versions of an image formatted for mobile devices.

Documentation resource: Renditions

What's new with CF03

This Combined Cumulative Fix includes new features and improvements.

Some features are ready for immediate use when you apply the latest combined cumulative fix. Other features require extra configuration or a procedure to enable the feature.

Documentation resource: “Roadmap: Applying maintenance” on page 88

Theme analyzer for portlet modules

The Theme Optimization Analyzer now supports Resource Aggregation for Portlets. You can view modules, contributions, and capabilities by page. You can see what resources are coming from the theme and what resources are coming from the portlets on the page. Support for pages was also extended to the export theme data function, so your output includes page, theme, and profile information.

Resource Aggregation for portlets

The system resource aggregator automatically loads dependencies on capabilities for portlets, if the theme metadata **`resourceaggregation.autoLoadPortletCapabilities`** is set to `true`. These capabilities and modules load automatically regardless of the profile on the page, if the capabilities and modules are installed and active anywhere on the system. The resource aggregator for portlets:

- Reduces the profile size
- Reduces the number of profiles that are required
- Reduces the need to modify the theme (a profile) or page (which profile assigned) to use certain portlets on a page
- Makes the system overall simpler and easier to use.

Documentation resource: Resource Aggregator overview

Public ResourceCombinerService API

Use the ResourceCombinerService public API for portlets that must link multiple resources on a page. The ResourceCombinerService public API links these resources in a combined, optimized way to keep the number of requests to a minimum.

Syndication extension points

The Web Content Manager API has been updated with new extension points for syndication to give developers additional tools to control the syndication of items. See the Web Content Manager Javadoc for details.

Worklight 6.2 integration

Integration with IBM Worklight is upgraded to support version 6.2. Use the features in Apache Cordova and the Worklight Client APIs to use local device capabilities in your Portal solution. With this upgrade, you can use the improved operational analytics, client-side log capture, rolling server upgrades, and more.

SmartCloud for Social Business Search (SAML)

You can now establish single sign-on between your portal installation and IBM Connections in SmartCloud for Social Business which provides an integrated digital experience for users. Integration between WebSphere Portal Express and IBM Connections in SmartCloud for Social Business is completed by using Tivoli Federated Identity Manager, which supports SAML-based single sign-on. For more information on integrating IBM Connections in SmartCloud for Social Business with WebSphere Portal Express, review the documentation resource.

Documentation resource: "Configuring Portal to work with Connections in SmartCloud for Social Business" on page 762

Invalid friendly URLs

You can now validate friendly URLs for web content. Additionally, you can create customized error messages that display when a user tries to access an invalid URL.

Documentation resource: "How to validate friendly URLs for web content" on page 2045

SmartCloud for Social Business HTTP Outbound

You can now activate outbound HTTP connections to remote resources with IBM Connections in SmartCloud for Social Business by using SAML 2.0 tokens.

Documentation resource: "Establishing SSO connections through SAML 2.0 tokens" on page 3036

@mentions

Starting with IBM Connections 5.0 you can use the @mentions feature to mention other IBM Connections users in your content or to notify them of the post. You can insert the feature @username anywhere in the body of the post. The feature is supported in the body and comment sections of the Wiki page, Blog post, Forum topic, and file comments. A site visitor can get details about a user mentioned in IBM Connections content that is rendered on WebSphere Portal Express pages. @mention reference is displayed by using the portlet person card when you hover over the link. If WebSphere Portal Express is configured to use IBM Connections business card, then the business card is shown instead of portal person card.

SmartCloud for Social Business Connections integration

You can now use social rendering with content items from IBM Connections in SmartCloud for Social Business. The IBM Connections integration assets that are

used to integrate IBM WebSphere Portal Express with IBM Connections can also be used to integrate WebSphere Portal Express with IBM Connections in SmartCloud for Social Business as well.

Web Application Bridge Mobile support

Mobile support is provided for web applications that were developed and tested for rendering inside mobile device browsers.

Windows Mobile Support

Device detection has been added for Windows Mobile devices. Windows smart phones can render the mobile theme instead of the full desktop version of the theme.

Web Content Manager Rest API

The Web Content Manager Rest API has been extended to include projects and workflow item types.

Documentation resource: "How to manage web content items by using REST" on page 3324

Inplace editing for Web content

The default setting for inplace editing of Web content has been changed to "embed mode" for text and rich text fields. See the section **Default inplace editing mode** in the Web content authoring options topic.

Documentation resource: "Web content authoring options" on page 396

What's new in V8.5

Learn what's new in IBM WebSphere Portal Express Version 8.5 .

What's new for administrators in V8.5

Version 8.5 includes new features and improvements for administrators, such as syndication troubleshooting tools, staging-to-production tools and more.

If you are migrating from version 7.0, review the *What's new* content for version 8.0 too. A link is provided at that end of the page.

Roadmaps

Roadmaps provide high-level steps that help you set up your environment. For example, an environment might be a development environment or an authoring environment. The roadmaps are intended to help you simplify the complex tasks and interfaces that are required to help you achieve your environment.

Document resource: Chapter 3, “Roadmaps,” on page 69

The roadmaps include links to essential supporting documentation resources. The installation roadmaps include topology diagrams to help you visualize your environment.

Installation

The installation and deployment is improved. Start with the appropriate roadmap for your environment. Then, use the Installation Manager to install a working portal with a Derby database. After a successful installation, go to the Configuration Wizard to deploy your environment.

- Chapter 4, “Installing,” on page 101
- “Configuration Wizard” on page 232

These changes simplify the installation and configuration process.

DB2® is no longer preinstalled with the Installation Manager installation. You can upgrade your WebSphere Portal Express installation to Enable, Extend, Web Content Manager, or Web Content Manager Standard Edition. These changes simplify the installation and create a flexible deployment.

Configuration Wizard

For new deployments, the configuration wizard provides you with a guided path to apply a topology to your environment. The topologies guide you through the steps that are required for your environment. You can transfer your database, enable federated security, and create your cluster. For existing deployments, you can also use it to add on features, migrate, or upgrade your existing product offering.

Migration

Review the migration roadmaps to develop your migration plan. When you are ready to migrate, start with the roadmap to prepare your source and target environments. Then, use the Configuration Wizard to run the migration from your previous version to Version 8.5. The Configuration Wizard automatically migrates your data.

Security

Attribute based security for Web Content Manager content is a new access filter in the product filter chain. You can extend the access control permission checks for Web Content Manager content beyond the user or group-based decisions. You can define your own criteria. The criteria might involve categories, keywords, textComponents, htmlComponents, or shortTextComponents for an item. For more information, read “Enabling Attribute Based Security” on page 1572.

Search

The remote search service installation requires fewer manual steps. Install the remote search service with the Installation Manager.

The **Did you mean?** Search feature improves the search experience. It recommends suggestions or corrections for keywords in a search query. The **Did you mean?** Search feature is available for immediate use.

New search configuration parameters provide higher-quality search results for site visitors. The **boostingSettings** parameter enables the allocation of extra weight to search terms in a query that are found in specific metadata fields of a document, such as title or description. You can also change the default search operator from `Or` to `And`. This change ensures that all terms listed in a search query are found in all of the documents that are returned in the search results list.

Staging to production

Use a Portal Application Archive (PAA) file to package and deploy your initial server. Export your initial server into the PAA file. Then, you can deploy the single PAA file instead of importing multiple resources.

After you set up your initial server, you can create a PAA file for your differential release. Then, you can deploy the single PAA file instead of deploying multiple differential resources. You must create PAA files for all of your virtual portals.

Syndication

Additional tools have been added to the syndication and subscriber views to help troubleshoot failed items during syndication.

Each failed item for a selected syndicator is displayed in the **Failed Items** view. Information is displayed about each failed item, including information about what the appropriate action is to fix the issue.

Root and Impact columns have been added. These new views are used to find the root cause of a syndication failure, and what secondary items are impacted by the root cause. By finding and fixing the root cause of the syndication failure, you also potentially fix the syndication failures of the items that are impacted by the root cause.

Web application bridge

Use the new web application bridge interface to create pre-configured web dock applications for your content authors. Create content provider profiles that include at least one policy. The policy provides directives and tuning parameters for a specific context root or a regex to provision certain applications or resource paths. Then, create web dock applications that are pre-configured with the correct settings. The content author finds the appropriate web dock application and adds it to their page.

Digital Data Connector

The IBM Digital Data Connector (DDC) for WebSphere Portal Express is a new feature to integrate data from external data sources on portal pages by using IBM Web Content Manager presentation components. Use the Digital Data Connector

for WebSphere Portal Express to create list-rendering profiles that define the set of attributes available for your content authors to integrate external data sources.

Information mode and user assistance

Information mode displays more inline information, examples, and hovers help in the user interface, such as the site toolbar. Information mode gives you more assistance when you need it. After you are comfortable and understand the user interface, you can turn off information mode. Information mode is a global setting. Information mode is not implemented for the entire user interface. For example, it is not available for the administration portlets. From the site toolbar, you can turn it on and off to suit your needs. The user assistance is now divided into two help sections. One section is specific to the administrator and one is specific to the content author.

Accessing administration options

You can now access Administration from the toolbar. Click  and select a specific area of administration from the menu. You can access all the Administration options in the navigation section after you open an administration page from the menu.

IBM Knowledge Center

IBM Knowledge Center requires active scripting. If active scripting is disabled, IBM Knowledge Center displays a gray page, and the content is not available.

Related information:

 WebSphere Portal V8.0: What's new for administrators

 IBM Web Content Manager V8.0: IBM Web Content Manager new features and improvements

What's new for developers in V8.5

Version 8.5 includes new features, such as theme optimization analyzer and simple modules.

If you are migrating from version 7.0, review the *What's new content* for version 8.0. A link is provided at that end of the page.

What's new for front-end developers

Configuring outbound HTTP connections

IBM WebSphere Portal Express now provides an easier way to configure outbound HTTP connections. In WebSphere Portal Express Version 8.0 and earlier versions, outbound HTTP connections were accessible through the Ajax Proxy service. The Ajax Proxy service was configured by a configuration document named `proxy-config.xml`. You find this document in the `/WEB-INF` directory of the web module that uses the Ajax Proxy service. Starting with WebSphere Portal Express Version 8.5 and the new outbound connection service, the configuration of outbound HTTP connections is now part of the standard datastore-based portal configuration.

Theme optimization analyzer

The theme optimization analyzer creates a validation report that analyzes your theme and theme components for known issues and reports the

number of errors, warnings, and informational messages. It also includes a detailed explanation about how to fix the errors that occur.

Simple modules
Simple modules for the resource aggregator framework are provided in the WebDAV folder. You can define modules quickly with a limited set of features with these simple modules.

Collaborating with Worklight
You can expand the capabilities available to your application when you create a hybrid application that adds native device functions to your portal with Worklight. When your hybrid application runs with your WebSphere Portal Express pages rendered in a native application, WebSphere Portal Express loads the appropriate native resources for the device. These resources are loaded automatically through modules that are provided in WebSphere Portal Express.

What's new for portlet and application developers

Attribute Based Security for Web Content Manager content

Attribute based security for Web Content Manager content is a new access filter in the product filter chain. You can extend the access control permission checks for Web Content Manager content beyond the user or group-based decisions. You can define your own criteria. The criteria might involve categories, keywords, textComponents, htmlComponents, or shortTextComponents for an item. For more information, read "Enabling Attribute Based Security" on page 1572.

Dojo is no longer required for tagging and rating
The tag and rating widgets of earlier portal versions required Dojo to be interactive. The new inline tag and rating widgets do not require Dojo to be interactive.

Use the Configuration Wizard to deploy PAA files
You can now install and deploy Portal Application Archive (PAA) files with the Configuration Wizard. Start the wizard and then go to **Add On New Capability** > **Install Add-ons**.

Social Rendering List templates
WebSphere Portal Express page editors can use social rendering to feature social data that is hosted on a remote IBM Connections server in the context of portal pages.

IBM UX Screen Flow Manager
The new IBM UX Screen Flow Manager enables you to quickly build wizard-like applications that guide users through sequences of screens. In many cases such flows can be modeled declaratively, without any programming efforts by interconnecting multiple portlets. Thus, IBM UX Screen Flow Manager fosters reuse of portlets across multiple flows and eases the maintainability of your applications.

What's new for web content developers

Custom HTML editor integration
Custom HTML fields are used to integrate third-party editors into the web content authoring interface. You can use custom HTML editors in all HTML fields of the web content authoring interface, or in single HTML elements that are defined in an authoring template.

Default inplace editing mode

The default inplace editing mode can now be configured. Supported fields can be pre-configured to either use inplace editing mode, embedded editing mode, or dialog editing mode. The default inplace editing mode can be overridden in EditableElement tags using the mode parameter, or in the content template for content items.

Related information:

WebSphere Portal V8.0: What's new for user interface developers

Theme Enhancements in WebSphere Portal 8.5

Theme Analyzer Enhancements in WebSphere Portal 8.5

What's new for content authors in V8.5

Version 8.5 includes new capabilities for content authors to make content creation easier.

Enhanced authoring user interface

Use the new integrated site toolbar to quickly create and edit content. Within the new site toolbar, you can drag content to the page you are editing.

You can associate vanity URLs with portal pages and labels. Vanity URLs are short URLs that people can easily remember. They are shorter than full WebSphere Portal Express URLs. They are sometimes also called marketing URLs. You can publish vanity URLs for marketing campaigns through different channels, such as email or print. This way, you can use vanity URLs to direct customers to a specific portal page or content item. Interested site visitors who want to view your campaign can then remember or copy the short vanity URL and type it into the browser address field.

You can now add social content from external sources to your page.

You can integrate third-party content on your site with web dock application portlets.

Page editors can integrate social content in the context of portal pages.

Learn more: Site toolbar

Targeted content

Targeted content matches the content that displays in a spot to the site visitor that views the spot. Content authors do not need IT experience to configure a spot on a page to show targeted content. The administrator creates profiler rules for the content authoring and marketing team in the personalization area of the site. The profiles that are created in the profiler rules appear as segments when you configure your spot. Specifically, the segment name that is displayed in the UI matches the label that is specified for the profiler rule. Then, the content author uses the **Targeted Content** selection in the Configure Spot menu to map content to segments.

Learn more: Configure your content spot

Digital Data Connector

Use the new IBM Digital Data Connector (DDC) for WebSphere Portal Express to integrate data from external data sources on portal pages by using IBM Web Content Manager presentation components. With Digital Data Connector, content authors and designers can use Web Content Manager presentation components to generate the web page markup for external data.

Syndication status

A new information window in the authoring portlet user interface that displays the current syndication status for each item. Users can determine whether an item is synchronized between the syndicator and subscriber, pending syndication, failed, or configured to be syndicated.

Information mode and user assistance

Information mode displays more inline information, examples, and hovers help in the user interface, such as the site toolbar. Information mode gives you more assistance when you need it. After you are comfortable and understand the user interface, you can turn off information mode. Information mode is a global setting. From the site toolbar, you can turn it on and off to suit your needs. You can now view online help that is specific to content authors. Click **Learn More** links or the question mark icon to find help that is specific to that topic.

What's changed

WebSphere Portal Express includes changes to existing features.

The following areas are changed in WebSphere Portal Express:

Installation Manager
: The Installation Manager installs WebSphere Portal Express with an Apache Server database and a default file repository. Then, you must use the Configuration Wizard to set up a stand-alone or clustered environment.

Roadmaps
: Go to Chapter 3, “Roadmaps,” on page 69 to find information about installation and deployment options, migration options, and integration options.

Configuration Wizard
: You can now use the Configuration Wizard to set up a stand-alone server and set up a cluster. Use the wizard to transfer from the Apache Server database to another supported database, enable federated security, migrate your server, install add-ons, and more.

Site toolbar
: The system administrator can use the site toolbar to create the projects and templates for their websites. The content authors can use the site toolbar to create pages and to add content and applications to their websites.

  You can now access Administration from the toolbar. Click  and select a specific area of administration from the menu. You can access all the Administration options in the navigation section after you open an administration page from the menu.

Vanity URLs
Vanity URLs are a new feature within WebSphere Portal Express. You can use vanity URLs instead of URL mappings. Go to “Vanity URLs” on page 2094 for information.

Sun Java™ Directory Server
Sun Java Directory Server was rebranded to Oracle Directory Server.

Web application bridge
The system administrator can create and configure multiple web dock applications. Then, the content author adds the application to their page without having to configure the Web Dock portlet.

Remote search
You can install the remote search service with Installation Manager.

Online help
There is online help for system administrators and a separate online help for content authors.

IBM Web Content Manager
The Web Content Manager documentation is merged with the WebSphere Portal Express documentation.

Dojo Dojo is not required anymore in your custom theme even in edit mode. You now can write a Dojo free theme, and still use Edit mode. However, some components require Dojo. You must use Dojo to use `wcm_inplaceEdit`, `wp_federated_documents_picker`, `wp_content_mapping_picker`, and the Search and Tag Center Profile.

TopNav removal
The TopNav from earlier themes has been removed from desktop view. The TopNav itself still exists, but it is hidden on desktops. If you want to use the TopNav in the desktop view, you can update the dynamic content spot definition in your theme's `plugin.xml`.

Default theme profiles
The standard profiles are now Lightweight and Lightweight with Dojo, Deferred and Deferred with Dojo, Basic Content and Basic Content with Dojo. The Full profile has been removed. Profiles can now be hidden also.

Web Content Viewer (JSR 286) portlet
The display title of the JSR 286 Web Content Viewer portlet changed from “Web Content Viewer (JSR 286)” to “Web Content Viewer.” The portal shows the new display title in the toolbar and administration portlets.

The portlet application ID, `ilwwcm-localrenderingportlet-jsr.war`, and the portlet name, Web Content Viewer (JSR 286), that are used with the XML configuration interface and with the portal scripting interface did not change.

Related information:

WebSphere Portal V8.0: What's changed

Unsupported and deprecated features for V8.5

Review the features that were available in previous versions of IBM WebSphere Portal Express but are no longer available.

After a migration, features that are no longer supported or are deprecated can result in unpredictable behavior. Remove unsupported and deprecated features

before you start your migration. As they become available, links to more information are provided to help you move away from deprecated features.

If you are migrating from version 7.0, review the unsupported and deprecated page for version 8.0 too. A link is provided at that end of the page.

Unsupported features and themes in Version 8.5

Web Clipper

The **Web Clipper** portlet is no longer supported. Use the **Virtual Web Application Manager** portlet instead.

IBM themes from a previous version

The following themes are no longer supported: Portal, PortalWeb2, and Tab Menu - Page Builder. These themes are migrated as is to WebSphere Portal Express Version 8.5. However, they no longer work and are no longer supported. You must manually update those themes. Merge their function into a clean copy of a Portal 8.5 theme on the target server.

The PageBuilder2 and 7.0.0.2 themes are supported if migrated to Version 8.5.

Documentation resource: "Enabling new functionality in a migrated portal" on page 909

Composite applications

Composite applications are no longer supported. If you have a composite application in your system and you are migrating to Version 8.5, the migration fails. Ensure that all composite applications are deleted before you start the migration. When you delete a composite application, you must also run the resource cleaner, otherwise pages can still exist in the database.

Documentation resource: "Removing unsupported composite applications" on page 828

CAI/TAI portlets

When you are migrating from one version to another, your script can contain references to the CAI/TAI portlets. These portlets are no longer available and any reference to these portlets cause your script to fail. For more information, see "Virtual Portal tasks" on page 876.

IBM Portlet API

The IBM Portlet API is no longer supported. Go to "Converting IBM portlets (IBM i Linux Windows)" on page 3110 to learn how to convert your portlets that are based on IBM Portlet API to the Standard Portlet API.

Deprecated features

Shared pages

Shared pages are deprecated.

Enabler, Builder, and Mashups components and API

The Enabler and the Builder components are deprecated. The Mashups Enabler and the Builder API are deprecated.

DB2 pre-installation

The WebSphere Portal Express installation no longer installs DB2. You can now install DB2 locally or remotely from the WebSphere Portal Express media. You can also use an existing DB2. With this change, you must upgrade to an extended offering. Extended offerings include WebSphere

Portal Express Extend, WebSphere Portal Express Enable, Web Content Manager, and Web Content Manager Standard Edition.

Internet and intranet templates
The following items are deprecated and replaced with Site Builder and the Content Template Catalog:

- New Site Wizard
- Sample intranet template
- Sample internet template

Full and Base installation options
Before WebSphere Portal Express Version 8.5, a customer chose either a full deployment with all the same pages and artifacts or a base deployment to customize their portal. Starting with Version 8.5, the Configuration Wizard installs the full deployment. Customers can then remove pages to customize their portal. Then, they can package their customizations as a Portal Application Archive (PAA) file. Finally, customers can install their production server, run the **empty-portal** task, and install the customization PAA file.

LikeMinds and Feedback
The LikeMinds and Feedback database domains are deprecated.

Stand-alone LDAP user registry
The stand-alone LDAP user registry configuration is deprecated. Instead, configure the federated LDAP user registry. If you upgraded from WebSphere Portal Express Version 7.0 or 8.0 with a stand-alone LDAP user registry, you can continue to use your stand-alone LDAP user registry. However, run the **wp-modify-federated-security** to change to a federated LDAP user registry.

Documentation resource: "Changing from a stand-alone repository to a federated repository" on page 589

Active Credentials
Active credentials are deprecated from the Credential Vault portlet. Passive credentials are still available.

Parallel Portlet rendering
The **Parallel Portlet rendering** feature is deprecated.

Adding a category from an outside component.
You can no longer add a category from an outside component. You must use the site toolbar instead.

LTPA version 1 token support
The LTPA version 1 token is deprecated. WebSphere Application Server Version 8.5.5. disables the LTPA version 1 token by default. If you are integrating with third-party applications that rely on LTPA version 1, update the application to support LTPA version 2. If you cannot update the application, you must manually re-enable LTPA version 1 support after you complete the migration. For information on updating your application to support LTPA version 2, see the documentation for the application.

Tagging and rating dialog and inline widgets
In tagging and rating, the tag and rating widgets of previous portal versions are deprecated. This affects both pairs of the old widgets, the dialog widgets of the default user interface and the inline widgets of the

alternative user interface. They are replaced by a new pair of interactive inline widgets. The new widgets combine the functions of the separate widgets of earlier portal versions.

URL mappings
URL mappings are deprecated. If you upgrade from Version 8.0 to Version 8.5, you can continue to use your existing URL mappings, but creating new URL mappings is no longer supported. Use vanity or friendly URLs.

Vanity URL documentation resource: "Vanity URLs" on page 2094

Friendly URL documentation resource: "Using friendly URLs" on page 1280

IBM Portlet API Web Content Viewer portlet
The Web Content Viewer portlet that is based on the IBM Portlet API has been removed and is no longer supported. If you use the IBM Portlet API Web Content Viewer, and you migrate to Version 8.5, then you must replace the portlet and its clones with the JSR 286 Web Content Viewer portlet as a post-migration step.

Documentation resource: "Convert the IBM Portlet API Web Content Viewer to the JSR 286 Web Content Viewer" on page 870

Note: Note: In the version 8.5 user interface, the JSR 286 Web Content Viewer portlet has been renamed to Web Content Viewer.

IBM Portlet API Remote Web Content Viewer portlet
The Remote Web Content Viewer portlet that is based on the IBM Portlet API has been removed and is no longer supported. If you use the IBM Portlet API Remote Web Content Viewer, and you migrate to Version 8.5, then you must replace the portlet and its clones with the JSR 286 Web Content Viewer portlet as a post-migration step.

Documentation resource: "Convert the IBM Portlet API Remote Web Content Viewer to the JSR 286 Web Content Viewer" on page 871

Note: Note: In the version 8.5 user interface, the JSR 286 Web Content Viewer portlet has been renamed to Web Content Viewer.

Deprecated integration portlets

The following integration portlets were removed in WebSphere Portal Express Version 8.5:

IBM WebSphere Portal Integrator for SAP
The IBM WebSphere Portal Integrator for SAP portlet is now deprecated. For SAP integration of single iViews or iPages, use the web application bridge.

Documentation resource: "Integrating with web applications" on page 977

Deprecated Business portlets

The following Business portlets were removed in WebSphere Portal Express Version 8.5:

Remote Rendering Portlet
The remote rendering portlet is now deprecated. Use the **Web Content Viewer (JSR 286)** portlet and Web Services for Remote Portlets (WSRP) instead.

Web2Bookmarks portlet

The **Web2Bookmarks** portlet is no longer available for immediate use. Instead, download the portlet from the IBM WebSphere Portal Business Solutions Catalog.

Catalog: IBM Collaboration Solutions Catalog

Deprecated theme

The WebSphere Portal Express Page Builder 7.x theme was deprecated.

Related concepts:

"Configuring authentication filters" on page 265
The portal authentication filters are a set of plug-in points. You can use them to intercept or extend the portal login, logout, session timeout, and request processing by custom code, for example to redirect users to a specific URL.

Related tasks:

"Federating the LDAP user registry" on page 860
The stand-alone LDAP user registry configuration is deprecated. Instead, configure the federated LDAP user registry. Run the **wp-modify-federated-security** task to change to a federated LDAP user registry.

Solutions Catalog: WebSphere Portal

Related information:

WebSphere Portal V8.0: Unsupported and deprecated features

WebSphere Application Server Network Deployment 8.5.5: Deprecated features

Product capabilities

IBM WebSphere Portal Express capabilities let you quickly implement web experiences that are engaging, flexible, and high performing.

WebSphere platform

WebSphere is IBM's integration software platform. It includes the entire middleware infrastructure - such as servers, services, and tools - needed to write, run, and monitor 24x7 industrial-strength, on-demand Web applications and cross-platform, cross-product solutions. WebSphere provides reliable, flexible, and robust integration software.

WebSphere provides software for Service Oriented Architecture (SOA) environments that enables dynamic, interconnected business processes, and delivers highly effective application infrastructures for all business situations.

IBM WebSphere Application Server drives business agility by providing millions of developers and IT Architects with an innovative, performance-based foundation to build, reuse, run, integrate, and manage Service Oriented Architecture (SOA) applications and services. From business critical and key enterprise-wide applications to the smallest departmental level applications, WebSphere Application Server offers the highest levels of reliability, availability, security, and scalability.

Customization

You can customize WebSphere Portal Express to meet the needs of your organization, users, and user groups. You can adapt the look and feel of the portal to fit the standards of your organization and to customize page content for users and groups in accordance with business rules and user profiles. Users, such as business partners, customers, or employees, can further customize their own views of the portal. Users can add portlets to pages and arrange them as they want and control portlet color schemes. By aggregating portlets in one place and giving users the power to customize their own desktops, WebSphere Portal Express gives users a means for doing business efficiently and with high satisfaction.

Portlets

Portlets are central to WebSphere Portal Express. As special reusable Java servlet that appear as defined regions on portal pages, portlets provide access to many different applications, services, and web content. WebSphere Portal Express ships a rich set of standard portlets, including portlets for displaying syndicated content, transforming XML, and accessing search engines and web pages. More portlet solutions are available on the IBM Lotus and WebSphere Portal Business Solutions Catalog. These portlets are used to access Lotus Notes® iNotes, IBM Sametime®, IBM Connections, and Microsoft Exchange. Several third-party portlets are also available. Examples include Enterprise Resource Planning (ERP), Dashboards, Business Intelligence, Process Management, and Customer Relationship Management (CRM) portlets.

It is possible to develop custom portlets, too. WebSphere Portal Express supports the Java Standard API (JSR 286) that portlet developers can use to create custom portlets.

Widgets

WebSphere Portal Express now includes widgets inside the portal. You can even create mashups that consist of both portlets and widgets. Widgets are highly interactive user interface components that are written in JavaScript. These widgets are typically very narrow in scope and can easily be created using a script-based language. Widgets can also be a solution for creating a mashup between different backend technologies like a Java EE-based portal server and a PHP-based server.

For additional information about new features, main components, and what each component provides to the overall solution, explore the subtopics of this section.

"Streamlined site creation" on page 23
You can use the Site Builder to generate your own portal site.

"Web analytics" on page 23
WebSphere Portal Express includes a number of solutions to help you understand how visitors use your site, including server-side analytics and client-side analytics. Client-side analytics is also called active site analytics.

"Web content" on page 25
IBM WebSphere Portal Express and IBM Web Content Manager help you manage content, share information, and communicate your message.

"Social business" on page 31
WebSphere Portal offers wikis, blogs, and tagging and rating capabilities. In addition, you can integrate existing collaboration applications with your portal site, such as Lotus Connections.

"Integration" on page 36
WebSphere Portal integrates with many products.
"Mobile" on page 38
A portal is a website that provides users with a single point of access to Web-based resources by aggregating those resources in one place and by requiring that users log in only to the portal itself, and not to each portlet they use. WebSphere Portal Express can also deliver Web content to WAP-enabled devices, i-Mode phones, Smart phones, and to various Web browsers. In addition, the IBM Mobile Portal Accelerator multi-channel server and mobile device repository extends portal content dynamically to over 7000 mobile devices, with new updates and devices added as they reach the market.
"Versatile framework" on page 40
IBM WebSphere Portal Express provides users a consistent view of portal applications and allows users to define specific sets of applications that are presented in a single context. Depending on the requesting device, the rendering of this application set must vary to fulfill the requirements of the device.

Streamlined site creation

You can use the Site Builder to generate your own portal site.

Creating portal sites on demand

Use the Site Builder to generate your own site – without needing any portal development skills or assistance from an administrator. Select a template from the available samples, choose the look that you want, and then allow the Site Builder do the rest.

The Site Builder automatically creates sites as virtual portals; however, administrators and developers can extend the wizard to create any type of portal site. Portal developers can also enhance the Site Builder by creating custom templates, and then adding them to the Site Builder.

Download the Site Builder from the IBM WebSphere Portal Express Business Solutions Catalog. Package contents include the portlet `.war` (web application archive) file, supporting files and directories, and detailed instructions in a `.pdf` file. After you deploy the Site Builder, you must add the portlet to a page that users can access. For more information, read the `.pdf` file that comes with the Site Builder.

Web analytics

WebSphere Portal Express includes a number of solutions to help you understand how visitors use your site, including server-side analytics and client-side analytics. Client-side analytics is also called active site analytics.

With Portal analytics you can achieve the following goals:

- Measure the success of your portal site.
- Predict the demand to a portal in the future.
- Plan for changing site visitor needs.

Active site analytics

Through active site analytics (client-side) WebSphere Portal Express makes it easier to collect, process, and report on your site usage. Integration with leading web

analytics products is easy. For example, Coremetrics page view tags are immediately available on pages in your site. In addition to integration with Coremetrics, you can also integrate with other web analytics tools such as, Unica, Omniture, and Webtrends.

Active site analytics:

Captures visitor behavior data directly from the web browser to ensure that every action is recorded.

Dynamically accommodates all content changes to the site so adding a page, product, or category typically requires no tag maintenance.

Creates a comprehensive record of visitor interactions and clients also do not have to pre-define segments, visitor events, campaigns, or categories for analysis.

Active site analytics uses a flexible, backend business logic instead of hardcoding the logic into the site instrumentation. As a result, you enjoy reduced IT maintenance and maximum analytical flexibility.

You can analyze web analytics reports in context, without switching to an external tool. This capability is called overlay reports. Overlay reports provide a quick look at the success of a page, portlet, or web content. Overlay reports are immediately available for Coremetrics, but you can extend that capability to other analytics tools.

Server-side analytics

Using server-side analytics, you can gather information about page and site visitor management activities, site visitor page and portlet requests, and session activities. Page and visitor management activities include creating, reading, updating, and deleting. Session activities include login, logout, timeouts, and login failures.

Related concepts:

"Logging and analyzing server side site data" on page 1688
IBM WebSphere Portal Express implements a logging function for your usage data. The portal writes usage records to a dedicated log file if site analysis logging is enabled. Multiple types of site analyzer loggers allow portal administrators to collect statistical data in various areas. The portal server manages the collection of data on its own, but from a business point of view you can also log custom details of business events. You can configure the portal for site analysis logging for the web content viewer.

"Analyzing user behavior by Active Site Analytics" on page 1696
You can collect data about user behavior in your portal and send that data to a service for analysis. For this purpose the portal provides Active Site Analytics (ASA).

"Collecting analytics data" on page 1698
Before you can send data about user behavior in your portal to a service for analysis, you need to collect that data. See the following topics for information about how to do this.

"Displaying overlay analytics reports" on page 1714
You can use Active Site Analytics to show graphical statistics reports about individual portal resources, such as pages or portlets. These reports are called Active Site Analytics overlay reports.

"Analytics tags and site promotions" on page 1724
To obtain further analytics information from your portal, you can use analytics tags for your portal resources. You can also use analytics tags for site promotions.

Web content

IBM WebSphere Portal Express and IBM Web Content Manager help you manage content, share information, and communicate your message.

"IBM Web Content Manager"
IBM Web Content Manager accelerates the creation, maintenance, and delivery of content across intranet, extranet, Internet, and portal sites.

"Conceptual and functional divisions of a website" on page 26
When you build a website by using Web Content Manager, you break up your website into the following conceptual and functional divisions.

"Content Template Catalog" on page 28
The content template catalog is a set of templates that you can use to accelerate building a website. Using these templates you can build a basic site quickly with little or no customization.

"Targeted content and Portal Personalization" on page 28
Targeted content provides you with a way to deliver multiple pieces of content to different audiences. Targeted content matches the best content with the most appropriate group by using segments. Segments help you split your audience into meaningful groups with different interests or characteristics.

"Portlets" on page 29
Portlets are a central part of IBM WebSphere Portal Express. Portlets are small applications that are independently developed, deployed, managed, and displayed. Administrators and users compose personalized pages by choosing and arranging portlets, resulting in customized Web pages.

Related information:

IBM Web Content Manager Multilingual Solution

IBM Web Content Manager

IBM Web Content Manager accelerates the creation, maintenance, and delivery of content across intranet, extranet, Internet, and portal sites.

IBM Web Content Manager is a comprehensive solution for creating, managing, and delivering content on your website.

Use Web Content Manager to publish your information and pull in content from other sources, either RSS feeds or other content management systems. On a single page, you can publish your corporate news and highlight news from an external news source.

Authors can edit content inline on staging servers and editors can approve content inline.

Using Personalization, you can target content based on the website users authentication and preferences. Personalization rules let you control what information specific users or user groups can see.

Workflows let you control how content is reviewed, approved, and published. You can create custom workflows to reflect your existing business processes.

Use syndication to publish approved content to your live website.

Content

You can manage and store content in Web Content Manager, using an external content management system, or both.

Set up libraries to manage and store content in Web Content Manager.

Use IBM Web Content Integrator to import content in RSS feed format.

Integrate and link directly to external content management systems using CMIS support included with Web Content Manager.

Use WebDAV to import content from a file system.

Wikis and blogs are immediately available for you to add to pages in your site.

Management

Use projects to manage changes to a group of items. Projects can help you coordinate approvals and publication.

Workflow is immediately available and you can develop custom workflows if needed. Workflows help you control how content progresses from draft to publication. Projects can have associated workflows too.

Authoring

Multiple authoring home pages are immediately available. You can also develop a custom authoring environment if needed.

In addition to the authoring home pages, authors can edit content in context on the staging servers. Using the edit mode, authors can select to edit content on a page. Changes adhere to defined workflows and syndication publishes changes.

Multilingual websites

The IBM Web Content Manager Multilingual Solution download, available on the solution catalog, provides a reference implementation for a multilingual website. It includes a set of extension plugins that you can use to configure and deliver a multilingual site.

Conceptual and functional divisions of a website

When you build a website by using Web Content Manager, you break up your website into the following conceptual and functional divisions.

Table 1. Conceptual and functional divisions for layout

Content	Layout
• Content items • Site areas • Components	• Presentation templates • Component designs • Page layout and portlets • Themes

Table 2. Conceptual and functional divisions for style

Context	Style
• Site framework • Page hierarchy • Profiling • Current user • Personalization	• CSS • Themes

When you are developing a website by using Web Content Manager and WebSphere Portal, you are splitting the elements of your website between content, context, layout, and style.

Content:

There are two types of content:

Dynamic

Dynamic content is generated dynamically based on a set of preconfigured parameters, such as navigator or menu.

Static Static content is content where you store markup or files directly in a component, content item, or site area. Where you store your static content depends on how the content is used:

- Page-specific content is stored in content items.
- Content that is related to a section of your site can be stored in a site area.
- Content that is reused in multiple sections of your site is stored in components.

Layout:

The layout and structure of each page in your website are defined by using these features:

- The overall structure of each page is determined by the theme you are using, the page layout you choose, and the web content viewer portlets you add to the page layout.
- The layout of the content that is displayed within each web content viewer is determined by markup that is used by the current presentation template, and by the markup that is stored within the elements and components that are referenced within the presentation template.

Context:

The context of the content that is displayed is also important. The layout and design of the page where a content item is displayed is different depending on:

- The current portal page (Different pages can use different themes, layouts, web content viewer portlets, and even different presentation templates.)
- The current site area (Content that is linked to different site areas can use different template maps that map to different site areas.)
- The current user (Different users can have different access to various page elements, including individual pages, portlets, and web content items.)

Each of these contextual variables can be used to display content or components in different ways, depending on the current context. Additionally, the profile of the current portlet, content item, or current user can determine what is displayed on a page, as do any personalization features used by your website.

Style: While you can place stylistic elements directly within your HTML, it is becoming increasingly common to use CSS to store and manage all the stylistic elements of a website, including:

- Stored server-side and referenced within a WebSphere Portal theme
- Stored directly within a content item or component

One good practice is to store your CSS markup within an HTML field that is stored in a content item. This provides quick access to the CSS if you need to edit the CSS. In addition, you can use advanced features such as workflows and inline editing to help you maintain your CSS. By storing all

your stylistic elements in CSS, you can quickly make stylistic changes to your website by editing a CSS file instead of editing multiple items.

When a web page is rendered, Web Content Manager takes all these individual item types and combines them to build a complete web page.

Content Template Catalog

The content template catalog is a set of templates that you can use to accelerate building a website. Using these templates you can build a basic site quickly with little or no customization.

The content template catalog is a comprehensive set of templates, pre-configured portlets, content-oriented theme, and layouts. Use the palette of pre-configured portlets to drop components on your pages for navigation, teasers, slide shows, carousels, and more. The template pattern supports the creation of your own templates. A page created from a template can be modified and then turned into a new template. You could build up you own custom template library and provide a self-service site development offering to your users.

You can download the content template catalog form the IBM Lotus and WebSphere Portal Business Solution Catalog

Related information:

 IBM Web Content Manager Forum: Content Template Catalog Announcement

Targeted content and Portal Personalization

Targeted content provides you with a way to deliver multiple pieces of content to different audiences. Targeted content matches the best content with the most appropriate group by using segments. Segments help you split your audience into meaningful groups with different interests or characteristics.

A targeted spot displays different content to different segments. You can create a target spot by defining content that is targeted to specific segments:

- Add content items to your content spot in a web content viewer.
- Add segments to each content item to display your content to the correct audience. Segments help you define your target audience. For example, you can define the audience by users, device class, or other attributes.

Personalization of content

Targeted content applies Personalization concepts to a new user interface where you create Personalization rules as you work to target content to selected segments. Personalization allows a portal or website to choose which content must appear for a particular user. The WebSphere Portal Express Personalization component selects content for users based on information in their profiles and on business rules. Using Portal Personalization, business experts can classify site visitors into segments and target relevant content to each segment. For example, a site that is using Personalization might show different news articles to managers than to regular employees or different information to valued customers. Personalization offers analytic capabilities to record site usage patterns and includes the LikeMinds recommendation engine, which provides collaborative filtering capabilities. Collaborative filtering uses statistical techniques to identify groups of users with similar interests or behaviors. Inferences can be made about what a particular user might be interested in, based on the interests of the other members of the group.

You can define content through a number of applications, including IBM Web Content Manager. Personalization automatically detects the content definition from these applications. Definitions of database or LDAP content types can also be made through a Personalization wizard included with IBM Rational® Application Developer.

After you define the content type, attributes of the content are shown to the rule author. The rule author can use these attributes to make conditions that define if and when certain content is displayed, or even if certain actions like database updates and triggered emails occur. To create new Personalization rules, go to **Applications** > **Personalization** > **Business Rules**.

Benefits of Personalization

- The Personalization component selects content for users that are based on information in their profiles and on business logic. With Personalization facilities, subject matter experts can select content that is suited to the needs and interests of each site visitor. These web-based tools help companies quickly and easily use content that is created by business and subject matter experts.
- Personalization classifies site visitors into segments and then targets relevant content to each segment. Business experts create the rules for classifying users and selecting content, by using web-based tools.
- Personalization has built-in capabilities for the IBM Java Content Repository. This means that personalization rules can easily be used in your Web Content Manager solutions.
- Personalization also includes a recommendation engine that provides collaborative filtering capabilities. Collaborative filtering uses statistical techniques to identify groups of users with similar interests or behaviors. Inferences can be made about what a particular user might be interested in, based on the interests of the other members of the group.
- Campaign management tools are also included with Personalization. Campaigns are sets of business rules that work together to accomplish a business objective. For example, a Human Resources manager might want to run a campaign to encourage employees to enroll in a stock purchase plan or sign up for some other new benefit that is now available to employees. The Human Resources manager would define a set of rules that are shown to accomplish this business objective. Campaigns have start and stop dates and times and can be email and web-page based. Several campaigns can run simultaneously and can be prioritized.

Related concepts:

"Developing a personalized portlet" on page 2385
This exercise demonstrates how to use Personalization features of WebSphere Portal and Rational Application Developer to build your first personalized portlet. Your final result is a working portlet that uses Personalization rules and content spots to display personal news based on user attributes (or profiles).

Portlets

Portlets are a central part of IBM WebSphere Portal Express. Portlets are small applications that are independently developed, deployed, managed, and displayed. Administrators and users compose personalized pages by choosing and arranging portlets, resulting in customized Web pages.

WebSphere Portal Express ships a rich set of standard portlets. For the most up-to-date information about portlets, including the latest portlets that are

available for download, visit the IBM WebSphere Portal Business Solutions Catalog. Or, refer to Developing portlets for information on creating custom portlets.

Portlet applications

Portlets are more than simple views of existing Web content. A portlet is a complete application, following a standard model-view-controller design. Portlets have multiple states and view modes, plus event and messaging capabilities.

Portlets run inside the application server, similar to the way a servlet runs on an application server, but are aggregated to a complete Web page by the WebSphere Portal server. The portlet container provides a run-time environment where portlets are instantiated, used, and finally destroyed. Portlets rely on the WebSphere Portal Express infrastructure to access user profile information, participate in window and action events, communicate with other portlets, access remote content, look up credentials, and store persistent data.

Generally, portlets are administered more dynamically than servlets. For example, portlet applications that consist of several portlets can be installed or removed while the WebSphere Portal component is running. The settings and access rights of a portlet can be changed by an administrator while WebSphere Portal is running, even in a production environment.

Portlet modes allow a portlet to display a different user interface, depending on the task that is required of the portlet. A portlet has several modes of display that can be invoked by icons on the portlet title bar: View, Help, Edit, Configure, and Edit Shared Settings.

A portlet is initially displayed in View mode. As the user interacts with the portlet, the portlet can display a sequence of view states, such as forms and responses, error messages, and other application-specific states. Help mode provides user assistance. Edit mode lets the user change portlet settings. For example, a weather portlet might provide an Edit page for users to specify location. Users must be logged in to WebSphere Portal Express to access Edit mode. Configure mode changes the default look of the portlet for all portlet instances and Edit Shared Settings changes the look of the portlet on a specific page.

Each portlet mode can be displayed in normal, maximized, or minimized state. When a portlet is maximized, it is displayed in the entire body of a page, replacing the view of other portlets. When a portlet is minimized, by default, only the portlet title bar is displayed on the page.

Portlet API

The Java Portlet Specification addresses the requirements of aggregation, personalization, presentation, and security for portlets that run in a portal environment. WebSphere Portal Express supports both portlet standards JSR 168 and JSR 286. For more information about the standard portlet API, go to the topic *Standard portlet API*.

Portlet communications

WebSphere Portal Express allows portlets to communicate with each other. Portlet communication can be used to exchange data between portlets. This feature makes the portal easier to use.

The portal supports events as defined in the JSR 286 specification. It allows administrators to wire portlets by using the portal user interface.

For example, one portlet can display information about accounts, and a second portlet displays information about transactions that occurred for one of the accounts over the last 30 days. To display this information, the transactions portlet needs to obtain the appropriate account information when it displays the transaction details. This exchange of information is accomplished by communication between the two portlets, by using events as described in the JSR 286 specification. In this example, the account portlet defines a publishing event in its portlet descriptor. The transaction portlet defines this event as a processing event in its portlet descriptor. By using the portal user interface, you can now wire those two portlets together. After you did this wiring, when the account portlet throws an event, the transaction portlet receives this event and can show information about the transactions of this account.

Portlet services

Portlet services are used to provide common functionality to portlets. Each portlet service has its own service-specific interface for the functionality that it offers. WebSphere Portal Express supports portlet services for standard portlets. Standard portlets use a JNDI lookup to retrieve a `PortletServiceHome` object, which is used to retrieve a portlet service implementation. A portlet service can be started only by the code within a portlet. For more information about portlet services in the portal,go to the topic *Portlet services*.

Social business

WebSphere Portal offers wikis, blogs, and tagging and rating capabilities. In addition, you can integrate existing collaboration applications with your portal site, such as Lotus Connections.

"IBM Connections"
Integrate IBM Connections into your site by using community pages and Connections Portlets.

"IBM Sametime" on page 33
The IBM Sametime portlets are installed with WebSphere Portal Express and deployed automatically.

"Blogs and wikis" on page 33
A set of preinstalled web content libraries are supplied. With these libraries, you can add blog and wiki features to your websites. Use blogs, blog libraries, and wikis to tap into the power of the community and to change the way you work.

"Tagging and rating portal content" on page 34
Users can tag or rate portal content and view the tags and ratings. Tagging and rating allow users to better organize, categorize, and find portal content. This task includes Web Content Manager, Connections, and custom content. For example, users can tag or rate books in an online bookstore.

IBM Connections

Integrate IBM Connections into your site by using community pages and Connections Portlets.

Connections Portlets

Connection portlets are not installed with the portal but you can easily download them from IBM Lotus and WebSphere Portal Business Solutions Catalog.

The portlet includes activities, blogs, blog summary, bookmarks, bookmarks summary, profiles, profiles summary, wikis, forums, forums summary, community overview, and tags.

Community pages

Portlets on community pages are automatically scoped to the community membership and display content from the community in the portlets. For example, if your community contains a forum you can add the forum portlet to a community page. The portal site visitors can view and interact with the forum content from the portal site.

You can automatically create new communities for your pages during the page template instantiation.

Search integration

Integrating Connections content into your portal search makes it easier for site visitors to find community content. Visitors can search for a term and see results from your portal site and Connections. Each Connections component has a unique seedlist provider. To configure your portal search to include Connections content, you must configure a content source for each seedlist. Instructions are provided in the Connections product documentation.

Tag integration

You can configure you portal tag cloud to include tags from Connections. This configuration makes it easier for your site users to find content in the communities.

You can also add the Connections Tags Portlet to a page. Then, you can wire it to other Connections portlets. Then, site visitors can use tags to find community content faster in blogs, wikis, and more.

Profiles integration

You can configure your portal site to show Connections Business Card information. Names that are associated with Connections profiles display as active links. Your site visitors click the name to see the business card. From the business card visitors can link to communities, blogs, and more for the selected user.

Files integration

Content authors can easily create links to files stored on Connections. From the rich text editor, in IBM Web Content Manager, the author can browse Files. Authors can also use Web Content Manager markup generation to render the current contents of specific IBM Connections Folders.

Integration with Connections Files is based on the generic Content Management Interoperability Services (CMIS).

Related concepts:

“What's new for administrators in V8.5” on page 10
Version 8.5 includes new features and improvements for administrators, such as syndication troubleshooting tools, staging-to-production tools and more.

“Roadmap: Integrating with IBM Connections” on page 97

Related information:

IBM WebSphere Portal Business Solutions Catalog

IBM Connections Portlets for WebSphere Portal

How the Tags portlet interacts with the other IBM Connections portlets

IBM Sametime

The IBM Sametime portlets are installed with WebSphere Portal Express and deployed automatically.

The Sametime Web 2.0 Contact List portlet is available from the **Applications** > **Collaboration** area of the portal site.

People awareness

Users can view contact and other typical business card information for a registered user by displaying the Person card. The Person card is available through a wide range of portal components including Domino® and Sametime integration, personalization, and Web content authoring. To view the Person card, move the cursor over an active (underlined) person's name and then select **Click** for **Person Card**.

When Sametime is enabled in your portal configuration, users can work with the complete set of people awareness functionality, which includes instant messaging and application sharing through e-meetings. Person names appear aware - with a dynamic online status indicator. Click **Profile** to display full information about the person. Additional actions can include:

- **Send Mail**
- **Chat**
- **Add as Sametime Contact**

If you choose not to enable Sametime in your portal configuration, people awareness functionality is more limited. People's names appear as hyperlinks, but with no people awareness icon next to each name, and available actions on the Person card are limited to those that are native to WebSphere Portal Express.

Related concepts:

"Integrate with collaboration software " on page 927
IBM WebSphere Portal Express integrates with collaboration software to provide you with more effective and cost-efficient ways of accessing information, sharing ideas, communicating and working together. Key software that WebSphere Portal Express integrates with includes IBM Domino, IBM Sametime, and IBM Connections.

Blogs and wikis

A set of preinstalled web content libraries are supplied. With these libraries, you can add blog and wiki features to your websites. Use blogs, blog libraries, and wikis to tap into the power of the community and to change the way you work.

Blogs are a great tool to use when you want to generate ideas around a single topic. You can use blogs for your own individual work or to gain feedback on a single concept from the larger team. Blog libraries take blogs to the next level. Rather than creating a blog per topic, you can use blog libraries to track multiple

topics in a centralized location. Wikis also provide you with another alternative for authoring content. Simple inline editing, including the insertion of images and links, makes authoring wikis quick and easy. You can also tag and rate blog and wiki content.

Blogs, blog libraries, and wikis use the template libraries that are provided by IBM Web Content Manager. Each blog, blog library, and wiki have its own library. The page hierarchy that is provided for these components is the common one defined by the Web Content Manager template libraries.

Tagging and rating portal content

Users can tag or rate portal content and view the tags and ratings. Tagging and rating allow users to better organize, categorize, and find portal content. This task includes Web Content Manager, Connections, and custom content. For example, users can tag or rate books in an online bookstore.

Portal users can tag or rate portal content. This task includes the following types of resources:

- Portal resources, such as pages and portlets
- Web Content Manager resources, such as articles or images
- Custom resources. For example, these resources can be items in an online store or pictures in a portlet. Administrators can add these custom resources to the portal so that users can tag or rate them.

In general, all content in a portal that can be uniquely identified can be tagged or rated.

Users can apply tags and ratings both publicly and privately for the following purposes:

- Public tagging and rating helps users categorize, evaluate, and find portal content that is based on tags and ratings by other users.
- Private tagging and rating can help users create their own personal way to categorize, evaluate, and find portal content.

In detail, portal users can do the following tasks:

- Work with tags:
  - Tag portal content. Users can add tags to portal content. For example, the user can apply the tag `websphere` to a page that provides information about IBM WebSphere products. Users can remove tags that they applied themselves.
  - View tags and related portal content: Users can view the tags that are applied to individual portal resources, for example by starting the default tag and rating widgets. Users can also work with aggregated sets of selected tags:
    - Users can view tags that are applied to a set of resources by using a **tag cloud**. The tag cloud lists the tags in alphabetical order. Different font sizes indicate how often the tags are applied. Depending on how the administrator configures the tag cloud, the list can be portal wide or limited to particular items. For example, portal pages, or books available on the page where the user clicked the tag.
    - The tag cloud supports different views: users can switch between these views. For example, they can view all tags, only tags that they themselves applied, their own private tags, or the tags that were added most recently.

- Users can switch between different display modes. For example, they can have the tags to be displayed in a cloud as described earlier, or in a simple list.
- Users can use the tags that are displayed in the tag cloud to search for content. When a user clicks a tag, the portal shows a list of resources that have that tag that is applied. Clicking such a resource redirects the user to that resource itself. Users can also click multiple tags; in this case the list shows only resources that have **all** selected tags applied.
- When users work with the list of resources, they can have the list that is sorted by different criteria, for example, title, date, rating. The result list portlet also supports two different view modes: a summary and a detail view.

- Work with ratings:
  - Rate portal content:
    - Users can apply ratings to individual resources to show how much they "like" them. For example, a user can give a good book a rating of 4.
    - Users can change or remove ratings that they applied themselves. For example, a user can update the previous rating 4 to a 5.
  - View the ratings that they or other users apply to portal content.

Administrators can do the following tasks:

- All tasks that portal users can do as previously listed.
- Add content that users can tag or rate, for example, books in a bookstore.
- Assign users the access rights for tagging or rating content.
- Configure the tag clouds. By default, a tag cloud shows all tags that are applied to portal-wide resources. Administrators can add tag clouds to portal pages or themes and configure them as required. For example, they can limit a tag cloud to display only tags that is assigned to resources of a certain category, such as portlets. When users click a tag from that tag cloud, the result list shows only resources, in this case portlets.
- Move tags to a different portal system, for example, for staging or during migration.
- Obtain basic statistics about tagging and rating. For example, you can obtain the tags and tag counts for a specific portal page, or all tags that a specific user applied. You can write queries for more detailed statistics, or create a user interface to visualize them.

Developers can do the following tasks:

- Extend the tagging and rating capabilities of the portal by writing a custom user interface that uses the Java Model API or the REST API.
- Write queries to obtain statistics about tagging and rating, and write a user interface to visualize these statistics.
- Enable or disable filters, for example to prevent users from using unwanted words as tags. By default the portal provides a blacklist and a whitelist filter.

Note: Depending on your portal and your user groups, administrators or developers might consider creating user documentation for tagging and rating for their portal users.

Related concepts:

“Tagging and rating” on page 1297
Get an overview of the administrative tasks related to tagging and rating.

Integration

WebSphere Portal integrates with many products.

"Web application bridge integration"
Use the *Web Application Bridge* to integrate with web applications, such as Sharepoint.

"Process integration"
Use the Unified Task List portlet to integrate with business process solutions such as IBM Process Server, WebSphere Lombardi Edition, and other enterprise resource planning software.

"IBM Industry Toolboxes for WebSphere Portal" on page 37
Industry Toolboxes for WebSphere Portal provide a resource forum for customers and partners. They pair technology expertise with industry thought leadership to offer best-practice guidance optimized for the particular business you are in.

"Application integration" on page 38
A portal provides access to content, data, and services that are located throughout the enterprise. These services include predefined connectors and portlets, and tools for creating additional connectors and portlets.

Web application bridge integration

Use the *Web Application Bridge* to integrate with web applications, such as Sharepoint.

The web application bridge uses reverse proxy technology to integrate web-based content providers, such as the Microsoft SharePoint server, with IBM WebSphere Portal Express. Administrators must first define the virtual web applications or content providers. A lightweight iFrame portlet displays the content from the backend applications. Users can then access the iFrame on a page without requiring direct network access to the backend application. A special engine maps the Uniform Resource Identifiers (URIs) on the iFrame portlet to real URIs from the content providers.

Integrating the web application bridge with WebSphere Portal Express is a multistep process. First, create the content provider profile. Second, create at least one policy for each content provider profile. Then, create the web dock applications for the content providers. Finally, add the web dock application portlets to a new or existing page.

Related concepts:

"Integrating with web applications" on page 977
The web application bridge uses reverse proxy technology to integrate web-based content providers, such as the Microsoft SharePoint server, with IBM WebSphere Portal Express. Administrators must first define the virtual web applications or content providers. A lightweight iFrame portlet renders the content from the backend applications. Users can then access the iFrame on a page without requiring direct network access to the backend application. A special engine maps Uniform Resource Identifier (URIs) on the iFrame portlet to real URIs from the content providers.

Process integration

Use the Unified Task List portlet to integrate with business process solutions such as IBM Process Server, WebSphere Lombardi Edition, and other enterprise resource planning software.

Service-oriented architecture

For greater flexibility, the Unified Task List portlet retrieves data from the configured enterprise resource planning software. Then, the retrieved data is presented in the portal. The Unified Task List portlet can retrieve data from multiple enterprise resource planning systems and render the result in a cohesive user interface.

Developer resources

The Unified Task List portlet was developed by using Web Experience Factory. You can use Web Experience Factory builders to develop actions, customize the Unified Task List portlet, and develop custom forms.

You can download a Unified Task List Developer Pack from IBM Lotus and WebSphere Portal Business Solutions Catalog. The developer pack includes samples models and end-to-end business process transactions. Learn more about the developer pack in the Web Experience Factory wiki.

Related concepts:

"Overview of the Unified Task List portlet" on page 968
Review concepts about the Unified Task List portlet to understand the different elements.

Related tasks:

"Configuring Unified Task List portlet at run time" on page 970
You can customize the Unified Task List portlet at run time as a WebSphere Portal Express administrator. Customizing at run time involves accessing the deployed Unified Task List to configure the portlet settings.

"Configuring Unified Task List portlet for single sign-on" on page 970
Set up a single sign-on between IBM Process Server, IBM Forms Experience Builder, WebSphere Lombardi Edition, IBM Business Process Manager and WebSphere Portal Express. Single sign-on (SSO) provides a secure method of authenticating a user within an environment and applies that authentication to access other applications, systems, and networks for the duration of the session.

IBM Industry Toolboxes for WebSphere Portal

Industry Toolboxes for WebSphere Portal provide a resource forum for customers and partners. They pair technology expertise with industry thought leadership to offer best-practice guidance optimized for the particular business you are in.

Featured industries include:

- Government
- Healthcare
- Banking
- Insurance
- Retail
- Telecommunications
- Industrial
- Travel & Transportation

To learn more about IBM Industry Toolboxes and find available templates, visit IBM.com.

Related information:

IBM Industry Toolboxes for WebSphere Portal

Application integration

A portal provides access to content, data, and services that are located throughout the enterprise. These services include predefined connectors and portlets, and tools for creating additional connectors and portlets.

Enterprise resource planning (ERP) and customer relationship management (CRM) systems are excellent candidates for portlets because efficient, personalized access to these functions provides measurable return on your portal investment. IBM provides connectors to enterprise applications by using the Java Connector Architecture (JCA).

Standard Java connectors

JCA is a standard architecture for integrating Java 2 Enterprise Edition (J2EE) applications with enterprise information systems that are not relational databases. Each of these systems provides native APIs for identifying a function to call, specifying its input data, and processing its output data. The goal of the JCA is to provide an independent API for coding these functions.

JCA also defines a standard Service Provider Interface (SPI) for integrating the transaction, security, and connection management facilities of an application server with those of a transactional resource manager. Thus, JCA is a standards-based approach to managing connections, transactions, and secure access to enterprise application systems. IBM JCA connectors provide access to systems such as SAP, PeopleSoft, J.D. Edwards, Oracle Enterprise Edition, CICS, IMS, and Host-on-Demand. Leveraging its CrossWorlds acquisition, IBM plans to develop and integrate JCA connectors to many other systems.

Rational Application Developer provides a complete development and unit test environment for applications that use JCA connectors, Web services, and microflows. Rational Application Developer tools include support for Web Service Definition Language (WSDL), developer versions of the connectors, the Web Services Invocation Framework (WSIF), and the microflow engine.

Mobile

A portal is a website that provides users with a single point of access to Web-based resources by aggregating those resources in one place and by requiring that users log in only to the portal itself, and not to each portlet they use. WebSphere Portal Express can also deliver Web content to WAP-enabled devices, i-Mode phones, Smart phones, and to various Web browsers. In addition, the IBM Mobile Portal Accelerator multi-channel server and mobile device repository extends portal content dynamically to over 7000 mobile devices, with new updates and devices added as they reach the market.

"WebSphere Portal Mobile Experience" on page 39
Pages are optimized on smartphones, desktops, or almost any device. This optimization is made possible with the WebSphere Portal Mobile Experience, IBM Web Experience Factory smartphone builders, or the Mobile Portal Accelerator.

"Mobile Portal Accelerator" on page 39
If your site needs to support a broad range of mobile devices, the Mobile Portal Accelerator provides a mobile multi-channel server solution. Mobile Portal Accelerator offers a multi-channel server and mobile device repository capable

of automating web page presentation to over 8,400 mobile devices. Each of those devices can be optimized to the display and services capabilities that include smartphones, tablets, handheld phones, and kiosk devices. Site developers can write the content. Then, the multi-channel server solution automates the presentation for optimized display on thousands of mobile devices.

"Development tools"
Both IBM Web Experience Factory and IBM Rational Application Developer include functionality to help you develop exceptional web experiences for your mobile site visitors.

WebSphere Portal Mobile Experience

Pages are optimized on smartphones, desktops, or almost any device. This optimization is made possible with the WebSphere Portal Mobile Experience, IBM Web Experience Factory smartphone builders, or the Mobile Portal Accelerator.

The Mobile Experience demonstrated how to use a lightweight architecture that provides exceptional performance on mobile devices. You can start with the included sample code and then customize it to meet your specific business needs.

The Mobile Experience includes:

Sample mobile navigation widgets that follow common mobile application navigation patterns

Sample mobile page layouts that demonstrate common content interaction patterns

The Mobile Experience is available on the IBM Lotus and WebSphere Portal Business Solutions Catalog.

Related information:

 Mobile Solutions from the IBM Lotus and WebSphere Portal Business Solutions Catalog

Mobile Portal Accelerator

If your site needs to support a broad range of mobile devices, the Mobile Portal Accelerator provides a mobile multi-channel server solution. Mobile Portal Accelerator offers a multi-channel server and mobile device repository capable of automating web page presentation to over 8,400 mobile devices. Each of those devices can be optimized to the display and services capabilities that include smartphones, tablets, handheld phones, and kiosk devices. Site developers can write the content. Then, the multi-channel server solution automates the presentation for optimized display on thousands of mobile devices.

Related information:

 IBM Mobile Portal Accelerator information center

Development tools

Both IBM Web Experience Factory and IBM Rational Application Developer include functionality to help you develop exceptional web experiences for your mobile site visitors.

Web Experience Factory

Web Experience Factory includes new smartphone builders and samples to help developers generate mobile portlets and pages faster. Automation components include:

Native-looking navigation tabs, lists, buttons, and controls

Scrolling lists with display options, such as thumbnails and multiple line test with multiple styles

Access to smartphone features, such as HTML 5, orientation, and geographical location

Input UI patterns appropriate for smartphones, such as selectable lists and check box lists

Rational Application Developer

Rational Application Developer includes tools designed to help you develop portlet applications for WebSphere Portal Express. The portlet tools provide the following capabilities:

- Portlet project support for the standard portlet API.
- Web perspective views and editors for developing portlets.
- Portlet project wizard to create basic portlets, Faces portlets, and Struts portlets.
- Editing and validation of the portlet deployment descriptor (portlet.xml).
- Testing and debugging of portlets within the workbench using the WebSphere Portal Test Environment.
- Testing and debugging of portlets on a remote machine using the WebSphere Portal Server Attach.
- Visual tooling to insert portlet programming objects into JSP files, using Page Designer.
- Portlet sample applications, available in the Samples Gallery.
- Educational tutorials, available in the Tutorials Gallery.

Related information:

Web Experience Factory 8 Documentation

Versatile framework

IBM WebSphere Portal Express provides users a consistent view of portal applications and allows users to define specific sets of applications that are presented in a single context. Depending on the requesting device, the rendering of this application set must vary to fulfill the requirements of the device.

The tasks of the aggregation, which are repeated with each request that comes from the device, are:

- Gather information about the user, the device, and the selected language.
- Select the active portlets from the set of applications to which the user has access.
- Aggregate the output of the active portlets into a coherent, usable display.

WebSphere Portal Express also provides the ability to create a custom navigation model, which includes such features as:

- Multilevel navigation
- Customized themes and skins

- Custom navigation - navigation tree can be contributed to by portlets themselves
- Custom arrangement of portlets (and thus content) on a page

Another aspect of the versatile framework is the ability to personalize a user's portal experience, using "content spots" that render subscribed content based on the user and user's role in the portal.

"User experience"
Customizing the user's experience is one of the main goals of IBM WebSphere Portal Express. User and administrative portlets are provided for customizing content and the look and layout of pages.

User experience

Customizing the user's experience is one of the main goals of IBM WebSphere Portal Express. User and administrative portlets are provided for customizing content and the look and layout of pages.

Customizing pages

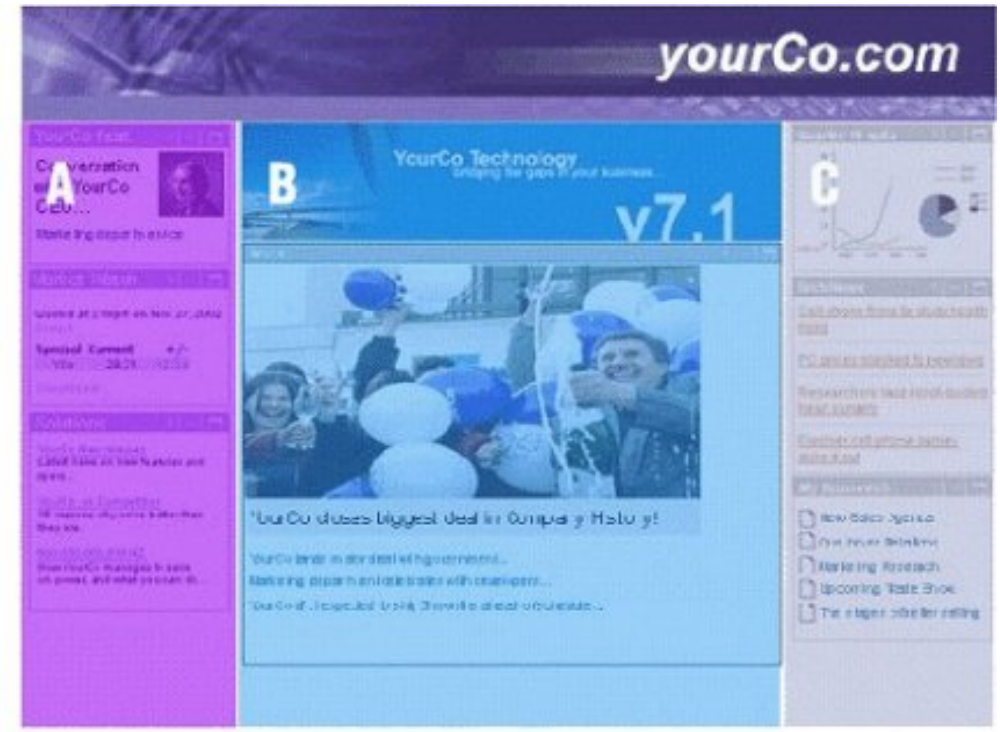
Users can have one or more custom pages and access each one through a different page. A page can contain a group of pages that is organized for a specific purpose. Each page can have a different set of portlets. Depending on authorizations, users can change the look and feel of their pages by using skins and page layouts. Also, page navigation hierarchy is tree-based, allowing any depth of nested pages.

The user or an administrator can set up the contents of each page. Administrators can specify that certain portlets be required, so that users are unable to move them or to remove them from the pages. Each page can have its own color scheme and column layout.

Cascading authorization

An administrator can grant or revoke access to customize a page or portion of a page to other administrators or users. The administrator can determine user's rights to modify a page. Administrators can control the edit authority that other administrators have on a page and its contents. This is designed to help organizations enforce policies and consistency, and create region-specific portals with some centrally managed content. This control is best explained through an example.

The first administrator can determine that a page will have three columns and not allow the column layout to be modified by any other administrators. A second administrator with lesser access cannot modify the column layout but can add portlets to these columns. The following figure shows a page split into three columns. Administrators can add portlets to these columns.



The second administrator adds a stock portlet to column one and a company news portlet to column two. This administrator wants these portlets to be available to everyone and does not want them to be removed. However, the administrator can add portlets to the columns. Therefore, the portlets are locked and cannot be removed by other administrators with lesser access. The following figure shows an example of how cascading authorization from one administrator to another would look.

Skins and themes

WebSphere Portal Express uses html, cascading style sheets, images, and other standard web design artifacts to define the look of pages. Java Server Pages (JSP) and other server-side dynamic techniques can also be used to help define the look of a site. You can add or modify elements to control visual aspects of WebSphere Portal Express, perhaps to add company-specific brand elements or to achieve a different color scheme and visual style. The system for defining color themes and skins supports multiple skins per theme, additional branding elements, navigation styles, and dynamic, browser-independent cascading style sheets.

You can apply skins and themes to a page, not only to the overall product. You can apply different skins individually to portlets as well, so that the appearance of a portal can be fine-tuned to meet any user need. By using a different theme for each page, a single installation can give the appearance of supporting many virtual portals.

Branding elements

You can change all visual elements of WebSphere Portal Express, including the masthead, the navigation areas, graphics, portlet title areas, and style sheets, to give WebSphere Portal Express a custom look. You can use standard file formats, such as JPEG, GIF, CSS, and JSP files, to define the look and layout.

The structure of the component installation folder contains folders named "skin" and "theme," with folders "html," "wml," and "chtml." These folders contain most of the files that are used for defining the basic structure of the home page, its color schemes, and portlet decorations. Portal designers can copy these folders and modify their contents to create a custom look and feel. The theme administration portlet registers the new files.

Changing portlet layout

You can change the placement of individual portlets on a page by using the drag-and-drop feature. To rearrange a portlet on a page, click the title bar of the portlet and then drag the portlet to a new location on the page. You can also add portlets to the page for quick and easy page customization by dragging portlets from the Portlet Palette to the page.

Universal access

The system of page templates, themes, skins, and portlet rendering is fully enabled for internationalization and accessibility by people with disabilities. For globally accessible portals, WebSphere Portal Express searches for and selects the proper JSP pages, based on the target browser and its settings for language and country.

Types of websites

Different types of websites require different solutions and use different applications and features. These examples describe details of different websites and the types of applications and features that are required to deliver them.

"Intranet portal"
An intranet portal site is designed to allow information to be quickly disseminated to employees, to make common internal business processes more efficient and to provide a sense of community within an organization.

"E-business site" on page 47
An e-business site is an externally facing site that is designed to market a company's products and services to consumers and allow them to purchase these items online. The focus of the site is on helping consumers match their needs to the appropriate product or service and maximizing their purchases.

"Brochure-ware site" on page 48
A brochure-ware site is an externally facing site that acts as an organization's presence on the World Wide Web. The overriding focus is on representing the organization's brand to its potential customers.

"E-library site" on page 49
An online library site is dedicated to providing access to a large amount of content. The prime example is a news site, where new content is created throughout the day, every day of the week and is published online and then archived as it becomes out of date. Other examples include journals, analyst reports, and software libraries.

"Partner site" on page 50
The audiences for a partner site are the partners of an organization. The site provides information and applications that are not applicable to the broader audience of consumers. A partner site would typically require a login but would not use an automated registration system. The business partners would be known to the company and given a login to access the partner site.

Intranet portal

An intranet portal site is designed to allow information to be quickly disseminated to employees, to make common internal business processes more efficient and to provide a sense of community within an organization.

The site provides access to:

- News about what is happening within the organization.
- Alerts that contain information that employees should be aware of and potentially action.
- Forms for working with various internal processes such as leave, purchases, and travel.
- A policies and procedures library with online versions of all policy and procedure documentation.
- Organizational communities with collaborative features like blogs and forums.
- A search system to enable employees to find content.

The intranet portal also has a personalized home page that is dynamically built by using a set of rules that retrieve content based on the current user's role, department, and location. By tagging the content, and then matching this content with the current user, only content that is appropriate for the current user is displayed.

A number of components are used together to make this intranet portal work:

- WebSphere Portal is used to provide a platform for the integration of content and applications that form the intranet portal.
- The overall theme and top-level navigation of the intranet is also managed by using WebSphere Portal.
- Web Content Manager provides the micro-level layout of the content within the portal, and is also used to directly build the news, alerts, and communities content.
- The forms are online applications that are built by using the Forms/Lists system, being pulled into the site by using Web Content Manager
- Policies and Procedures are document libraries managed using IBM FileNet Content Manager.
- A custom feed producer would be written to allow the Web Content Integrator to consume a feed from IBM FileNet Content Manager and Web Content Manager would be used to present the documents in a website.
- Personalization is used to associate content with users, dynamically generating the content that is displayed in the personalized home page
- A third-party discussion system is used to deliver collaborative forums in the communities section of the site using a custom portlet.
- WebSphere Portal search is used with the Web Content Manager categorization to provide both a simple text search and a more advanced category-based search.

The size of an intranet portal tends to scale with the size of the organization. A large organization has more information to disseminate, more business processes, more communities of employees. This means that the content and the user population tend to grow at the same rate.

Related concepts:

"Roadmap to building a web content system" on page 1781
To build a web content system you need to deploy hardware, configure servers, design an authoring system, configure a delivery environment and enable syndication. Get an overview of the steps required to build your web content system. Keep in mind the analysis and design documents developed during the planning phase of a project as you review the roadmap.

"IBM Web Content Integrator" on page 1936
The Web Content Integrator is a solution for integrating externally managed web content with WebSphere Portal Express. By using standard content syndication feed technologies based on RSS 2.0, the Web Content Integrator provides a loosely coupled mechanism for transferring published content and metadata to the portal after they are approved in the source system. When the content and metadata are transferred to the portal, it is possible to use the built-in content management features of Web Content Manager to secure, personalize, and display the content to users.

"Targeted content and Portal Personalization" on page 28
Targeted content provides you with a way to deliver multiple pieces of content to different audiences. Targeted content matches the best content with the most appropriate group by using segments. Segments help you split your audience into meaningful groups with different interests or characteristics.

Related information:


FileNet Content Manager

E-business site

An e-business site is an externally facing site that is designed to market a company's products and services to consumers and allow them to purchase these items online. The focus of the site is on helping consumers match their needs to the appropriate product or service and maximizing their purchases.

The site provides:

- Press releases about the latest news about the company's products and services.
- A promotions area with information about products and services that include sales, special package deals, and discounts.
- A products and services catalog area with the entire catalog displayed as a searchable taxonomy.
- A shopping area where users can see:
  - A searchable list of available products and services available for purchase online.
  - A summary of the items they are intending to purchase.
  - An area to calculate tax, shipping, and other costs.
  - An area to purchase these items online using mail order, or by using a follow-up call from a salesperson.
  - An area to track orders they initiated.
- Articles where the organization can help customers by using articles that discuss their products and services.
- A support area with technical documentation, contact numbers, support request forms, FAQs, and downloads.
- A service for subscribing to a newsletter where the user is sent a regular update that highlights new promotions, products, or articles.

The home page of the site is used to highlight the latest promotions, and to give a strong sense of the brand of the organization. Throughout the site, collaborative filtering is used to help suggest products and services to consumers based on their purchasing and navigational activity. A discussion system can also be used to allow users to build a community where they can comment on and rate various products and services.

A number of components are used together to build the e-business site:

- WebSphere Portal is used to provide a platform for the integration of content and applications that form the e-business site.
- The overall theme and top-level navigation of the intranet is also managed by using WebSphere Portal.
- Web Content Manager provides the micro-level layout of the content within the site, and is also used to directly build the press releases, FAQs, articles, and promotional content.
- The catalog and shopping areas are being run by using WebSphere Commerce.
- Personalization is used to suggest products to users based on collaborative filtering, and to email the newsletter.
- A third-party discussion system is used to deliver collaborative forums in the communities section of the site by using a custom portlet.

An e-business site can generate a large amount of traffic with a relatively small amount of content.

Related concepts:

Designing and setting up a portal site
WebSphere Portal Express 7 provides improved page builder and improved client side rendering and has a new default portal theme, the Page Builder theme. Themes define how your portal site will look. After installation, a default theme is deployed and you can either customize that theme or create your own. The new themes approach introduced in 7 involves less editing of JSPs and allows you to mix iWidgets and portlets on the same portal page and take advantage of both client side and server side rendering mode. WebSphere Portal 7 continues to support other themes, including your custom themes. If you have an existing portal site you can continue to use your existing themes or you can migrate your themes to the new standard.

"Roadmap to building a web content system" on page 1781
To build a web content system you need to deploy hardware, configure servers, design an authoring system, configure a delivery environment and enable syndication. Get an overview of the steps required to build your web content system. Keep in mind the analysis and design documents developed during the planning phase of a project as you review the roadmap.

"Targeted content and Portal Personalization" on page 28
Targeted content provides you with a way to deliver multiple pieces of content to different audiences. Targeted content matches the best content with the most appropriate group by using segments. Segments help you split your audience into meaningful groups with different interests or characteristics.

Related information:

IBM WebSphere Commerce

Brochure-ware site

A brochure-ware site is an externally facing site that acts as an organization's presence on the World Wide Web. The overriding focus is on representing the organization's brand to its potential customers.

This type of site is relatively static and does not need the aggregation capabilities of WebSphere Portal and is instead delivered by using both the servlet delivery and pre-rendering features of Web Content Manager. The site is also designed to be easily indexed by search engines like Google.

This site would include:

- Information about the organization that includes its ethics, goals, mission, and history.
- A news area where the organization can talk about what they are doing now and into the future.
- A contact area with lists of locations, phone numbers, and online contact forms where users can submit queries, ask for follow-up calls.
- A job opportunities area where an organization can advertise positions
- A partners area where partners of the organization are listed with information about them and their relationship with the organization, and links to partner websites and partner contact details
- Information about the organization's products and services that includes case studies and testimonials

The components in this site are limited with Web Content Manager providing all the presentation, navigation, and content.

A brochureware site can generate a large amount of traffic with a relatively small amount of content.

Related concepts:

"Roadmap to building a web content system" on page 1781
To build a web content system you need to deploy hardware, configure servers, design an authoring system, configure a delivery environment and enable syndication. Get an overview of the steps required to build your web content system. Keep in mind the analysis and design documents developed during the planning phase of a project as you review the roadmap.

E-library site

An online library site is dedicated to providing access to a large amount of content. The prime example is a news site, where new content is created throughout the day, every day of the week and is published online and then archived as it becomes out of date. Other examples include journals, analyst reports, and software libraries.

Users would typically register for these sites, and might then be sent emails regularly summarizing the latest content. On a news site, the latest news might be free, but users might pay for access to archived content. On other types of "library" sites, users might pay for access to any of the content.

A number of components are used together to deliver an e-library site:

- WebSphere Portal is used to provide a platform for the integration of content and applications that form the e-business site.
- The overall theme and top-level navigation of the intranet is also managed by using WebSphere Portal.
- Web Content Manager provides the micro-level layout of the content within the site, and is also used to directly build content.
- For e-libraries where the content is presented as documents, then IBM FileNet Content Manager would be used to store and manage the documents.
- A custom feed producer would be written to allow the Web Content Integrator to consume a feed from IBM FileNet Content Manager and Web Content Manager would be used to present the documents in a website.

An e-library site might have a large audience and a large amount of content.

Related concepts:

Designing and setting up a portal site
WebSphere Portal Express 7 provides improved page builder and improved client side rendering and has a new default portal theme, the Page Builder theme. Themes define how your portal site will look. After installation, a default theme is deployed and you can either customize that theme or create your own. The new themes approach introduced in 7 involves less editing of JSPs and allows you to mix iWidgets and portlets on the same portal page and take advantage of both client side and server side rendering mode. WebSphere Portal 7 continues to support other themes, including your custom themes. If you have an existing portal site you can continue to use your existing themes or you can migrate your themes to the new standard.

"Roadmap to building a web content system" on page 1781
To build a web content system you need to deploy hardware, configure servers, design an authoring system, configure a delivery environment and enable syndication. Get an overview of the steps required to build your web content system. Keep in mind the analysis and design documents developed during the

planning phase of a project as you review the roadmap.

"IBM Web Content Integrator" on page 1936

The Web Content Integrator is a solution for integrating externally managed web content with WebSphere Portal Express. By using standard content syndication feed technologies based on RSS 2.0, the Web Content Integrator provides a loosely coupled mechanism for transferring published content and metadata to the portal after they are approved in the source system. When the content and metadata are transferred to the portal, it is possible to use the built-in content management features of Web Content Manager to secure, personalize, and display the content to users.

Related information:

FileNet Content Manager

Partner site

The audiences for a partner site are the partners of an organization. The site provides information and applications that are not applicable to the broader audience of consumers. A partner site would typically require a login but would not use an automated registration system. The business partners would be known to the company and given a login to access the partner site.

A partner site can include the following areas:

- A news area that is focused on partners. Larger organizations that work in multiple areas of business might personalize the news area to suit each partner.
- A promotions area that details special deals for partners.
- Online services, providing useful tools and resources for partners to use, such as calculators, forms, and catalogs.
- A billing area where partners can see the current bill status for services that are contracted from the organization.
- A community area where partners might post information about themselves, and also interact to help each other and receive help from the organization.

A number of components are used together to deliver a partner site:

- WebSphere Portal is used to provide a platform for the integration of content and applications that form the intranet portal.
- The overall theme and top-level navigation of the intranet is also managed by using WebSphere Portal.
- Web Content Manager provides the micro-level layout of the content within the portal, and is also used to directly build the news and promotions content.
- Various IBM products plus third-party applications are used to build and deploy the online services area.
- The catalog and billing services in the site is run by using WebSphere Commerce.
- A third-party discussion system is used to deliver collaborative forums in the communities section of the site by using a custom portlet.
- Personalization is being used to filter content throughout the site to ensure that the appropriate content and inline services are reaching the appropriate partners.

A partner site is likely to have a limited audience but a large amount of content.

Related concepts:

Designing and setting up a portal site
WebSphere Portal Express 7 provides improved page builder and improved client side rendering and has a new default portal theme, the Page Builder theme. Themes define how your portal site will look. After installation, a default theme is deployed and you can either customize that theme or create your own. The new themes approach introduced in 7 involves less editing of JSPs and allows you to mix iWidgets and portlets on the same portal page and take advantage of both client side and server side rendering mode. WebSphere Portal 7 continues to support other themes, including your custom themes. If you have an existing portal site you can continue to use your existing themes or you can migrate your themes to the new standard.

"Roadmap to building a web content system" on page 1781
To build a web content system you need to deploy hardware, configure servers, design an authoring system, configure a delivery environment and enable syndication. Get an overview of the steps required to build your web content system. Keep in mind the analysis and design documents developed during the planning phase of a project as you review the roadmap.

"Targeted content and Portal Personalization" on page 28
Targeted content provides you with a way to deliver multiple pieces of content to different audiences. Targeted content matches the best content with the most appropriate group by using segments. Segments help you split your audience into meaningful groups with different interests or characteristics.

Related information:


IBM WebSphere Commerce

Accessibility features

Accessibility features help users who have a physical disability, such as restricted mobility or limited vision, to use software products successfully.

Accessibility by people with disabilities A US Section 508 Voluntary Product Accessibility Template (VPAT) can be requested with the IBM accessibility website. IBM Digital Experience offerings use the newest W3C Standard, WAI-ARIA 1.0 to ensure compliance to US Section 508 and Web Content Accessibility Guidelines (WCAG) 2.0. The best accessibility experience is provided by using the newest release of the browser and the newest release of the screen reader. For more information, see the following URLs:

- Accessible Rich Internet Applications (WAI-ARIA) 1.0
- Section 508 Standards for Electronic and Information Technology
- Web Content Accessibility Guidelines (WCAG) 2.0

Updated IBM Digital Experience V8.5 offerings include the following products.

- IBM Customer Experience Suite V8.5
- IBM Customer Experience Suite Rich Media Edition V8.5
- IBM Employee Experience Suite V8.5
- IBM Portal Server V8.5
- IBM Portal Enable V8.5
- IBM Portal Express V8.5
- IBM Web Content Manager V8.5
- IBM Web Content Manager Rich Media Edition V8.5
- IBM Web Content Manager Standard Edition V8.5

- IBM Content Accelerator V8.5
- IBM Content Accelerator and WebSphere Portal V8.5

This version of IBM WebSphere Portal Express:

- Supports installation through a command-line interface known as *console mode*. It is the accessible equivalent of installing by using the graphical user interface.
- Supports interfaces commonly used by screen readers and screen magnifiers (Windows only)
- Supports use of screen-reader software and digital speech synthesizers to hear what is displayed on the screen.
- Can be operated with only the keyboard
- Allows the user to request more time to complete timed responses
- Supports customization of display attributes such as color, contrast, and font size
- Communicates all information independently of color
- Supports the attachment of alternative input and output devices
- Supports alternatives to audio information
- Supports adjustable volume control
- Does not flash the screen at rates that can induce epileptic seizures
- Provides documentation in an accessible format

Supported browser: Firefox 25.

Recommended screen reader: JAWS 15.

The documentation includes the following features to aid accessibility:

- All documentation is available in HTML formats to give the maximum opportunity for users to apply screen-reader software technology.
- All images in the documentation are provided with alternative text so that users with vision impairments can understand the contents of the images.

When appropriate, the documentation for specific product features contains more information about accessibility.

See the related links for information about IBM's commitment to accessibility.

Related information:

IBM Human Ability and Accessibility Center

Chapter 2. Tutorials

Explore the following tutorials to learn more about using the IBM WebSphere Portal Express features.

"Tutorial: Creating and publishing content "
This tutorial demonstrates how you can use pages, projects, and workflow to create and approve draft pages. After approval, changes are published to a delivery server with syndication.

Related information:

Getting Started with Your Digital Web Experience

Tutorial: Creating and publishing content

This tutorial demonstrates how you can use pages, projects, and workflow to create and approve draft pages. After approval, changes are published to a delivery server with syndication.

During the tutorial, you will use different parts of the portal user interface. Most actions originate for the site toolbar.

Figure 1. Screen capture of the site toolbar with annotations.

1 Menu options

There are three menu options: Site, Application, and Administration.

Use the **Site** menu to switch between the sites that are hosted on a single portal server.

Use the **Application** menu to access application such as Content, Collaboration, Messaging, Personalization, and Unified Task List.

Use the **Administration** menu to access the administration portlets.

2 Toolbar

The toolbar provides access to authoring tools such as **Projects**, **Create**, and **Page**.

From the Projects tab, the content authors can create new projects to manage multiple changes. The tab displays the content that has been created or changed in the website for that project. Each content item can be managed individually. The content author can also create project templates.

From the Create tab, content authors can create pages, add applications to a page, or add content to a page. The content shelf contains Web Content Viewer portlets that are preconfigured with content associations. When the content author drags content to the page, a copy of the associated content is added to the page. Then the content author can customize the content.

From the Page tab, content authors can view and modify page settings, such as layout, vanity URLs, and more.

3 Project indicator

The project indicator provides access to the project menu and displays the currently active project. If there is no active project, then the indicator shows "Published Site" and you are viewing the currently published website.

4 Edit Mode

Use the **Edit Mode** control to turn editing capability on and off.

5 Information Mode

Turns inline user assistance on and off. Inline assistance includes descriptions and examples that appear in the user interface and hover help.

Currently this mode is only implemented in the site toolbar.

6 Menu

Provides access to additional page and preview actions.

Page actions include: View Page Properties, Show Hidden Pages, Show Hidden Content, Move Page, and Delete Page.

Preview actions include: Preview as a specific user and preview as an unauthenticated user.

Contents

When the project is submitted, SiteAdmin publishes the project, and the changes are then available for all users on the authoring server.

"Step 7. Syndicate changes to delivery server" on page 66

After SiteAdmin publishes the project, the updates are ready for syndication to the delivery server. Although syndication runs at scheduled intervals, SiteAdmin accesses the delivery server and manually starts the syndication process.

Create users and groups

The tutorial uses multiple user roles to demonstrate how content is created and flows through the system. To complete the steps in the tutorial as they are documented, you must create the groups and user IDs.

About this task

This tutorial uses the following user roles, IDs, and group assignments.

Table 3. User roles in this tutorial and their related user IDs and group assignments

User Role	User ID	Group
Site administrator	SiteAdmin	wpsadmins This group already exists and you do not need to create it.
Content author	Author1	ContentEditors
Content reviewer	Reviewer1	ContentReviewers
Content approver	Approver1	ContentApprovers

The tutorial highlights four user roles, the site administrator, content author, content reviewer, and content approver. Each user role performs specific tasks in the tutorial.

Site administrator (SiteAdmin)

The site administrator has administrator privileges in the portal and in Web Content Manager. SiteAdmin is part of the user group wpsadmins, and does these tasks:

- Sets up workflow stages and creates a custom workflow.
- Creates a page template that uses the custom workflow.
- Specifies access control settings where needed.
- Sets up a syndication relationship between the authoring server and the delivery server.
- Publishes projects for syndication.

Content author (Author1)

The content author is responsible for a certain portion of the site and can create pages as needed. In addition the content author adds and edits content on those pages. Author1 is part of the user group ContentEditors.

Content reviewer (Reviewer1)

Reviewer1 reviews changes submitted by content authors and either approves or rejects the individual changes. Upon approval, Reviewer1 submits the complete project for approval. Reviewer1 is part of the group ContentReviewers.

Content approver (Approver1)

Approver1 oversees changes to the site and is responsible for approving projects for publishing. Approver1 is part of the group ContentApprovers.

Procedure

1. Click the **Administration menu** icon. Then, click **Access** > **Users and Groups**.
2. Click **All Portal User Groups**.
3. Click **New Group** and type `ContentEditor` as the **ID**. Repeat this step to create ContentReviewers and ContentApprovers.
4. Click **ContentApprovers** to open the group. Then click **New User** to create Approver1.
5. Click **All Portal User Groups** in the breadcrumb.
6. Click **ContentEditors** to open the group. Then click **New User** to create Author1.
7. Click **All Portal User Groups** in the breadcrumb.
8. Click **ContentReviewers** to open the group. Then click **New User** to create Reviewer1.
9. Click **All Portal User Groups** in the breadcrumb.
10. Click **wpsadmins** to open the group. Then click **New User** to create SiteAdmin.

Step 1. Set up library access

Before the content authors, editors, and approvers can get started, the SiteAdmin must grant library access to content authors and approvers.

About this task

In this step, access is defined for the following libraries:

Portal Site
: Page objects and content for pages are stored in the Portal Site library. Contributor access is required for the ContentEditors group.

Web Content
: The custom workflow and workflow stages are stored in the Web Content library. Contributor access is required for the ContentEditors, ContentReviewers, and ContentApprovers groups.

Procedure

1. Log in as SiteAdmin.
2. Click the **Administration menu** icon. Then, click **Portal Content** > **Web Content Libraries**.
3. Set **permissions** for the Portal Site library.
   a. Click the **Set permissions** icon for the Portal Site library.
   b. Click the **Edit Role** icon for the Editor role.
   c. Click **Add**. Then, click **Search** to show all groups.
   d. Select the **ContentEditors** group.
4. Click **Portal Content** > **Web Content Libraries**.
5. Set **permissions** for the Web Content library.
   a. Click the **Set permissions** icon for the Web Content library.
   b. Click the **Edit Role** icon for the Contributor role.
   c. Click **Add**. Click **Search** to show all groups.
   d. Select the **ContentEditors**, **ContentReviewers**, and **ContentApprovers** group.

What to do next

Remember: In addition to appropriate access to libraries, content authors require Editor access to any pages and portlets that are involved. This access enables authors to create pages and add portlets to pages. This tutorial relies on sample content with predefined access. However, if you use your own content, ensure that you specify sufficient access on these resources in the portal so authors can work with them.

Step 2. Set up syndication

Updates to pages and content are transferred from the authoring server to the delivery server with syndication. When you publish a project, you can use syndication to automatically publish new pages and content updates. To enable this behavior, SiteAdmin sets up syndication between the required web content libraries of the two servers.

About this task

In this syndication relationship, the delivery server is the subscriber, and the authoring server is the syndicator.

Procedure

1. Log in to the delivery server as an administrator.
2. Create a shared credential vault slot for accessing the syndicator.
   a. Click the **Administration menu** icon. Then, click **Access** > **Credential Vault**.
   b. Click **Add a vault slot**.
   c. Enter a name for the vault slot. For example, `syndication-slot`.
   d. Click **new**, and enter a vault resource to associate with the slot. For example, `syndication-resource`
   e. Click **Vault slot is shared**, and enter the user ID and password to access the slot from the authoring server. This user ID and password must be defined on the authoring server. This tutorial uses the credentials for SiteAdmin.
   f. Click **OK**.
3. Then, click **Portal Content** > **Subscribers**.
4. Click **Subscribe Now**.
5. Specify the details for the syndicator and subscriber.
   a. In the **Syndicator URL** field, enter the URL to access the authoring server.

      This URL takes the form `http://hostname:port_number/wcm_context_root`. The default context root for Web Content Manager is `wps/wcm`. For example:

      `http://www.example.com:10039/wps/wcm`
   b. In the **Syndicator Name** field, enter a name to identify the authoring server in the syndication relationship. This name is used for the syndicator item that is created on the authoring server.
   c. In the **Subscriber Name** field, enter a name to identify the delivery server in the syndication relationship. This name is used for the subscriber item that is created on the delivery server.
   d. In the **Credential Vault Slot** field, select the credential vault slot that you created previously. For example, `syndication-slot`
   e. Click **Next**, and select the libraries that you want to syndicate.

      The following libraries are needed for this tutorial:

      - Portal Site: Contains page artifacts and related content.

- Web Content: Contains workflow items.
- Web Content Templates 3.0: Required for sample web content items.
- Template Page Content 3.0: Required for sample web content items.

For each library, select **Live items** in the **Scope** field.

f. Click **Finish**.

g. To verify the syndication connection, click the **Test connection** icon.

Step 3. Create workflow

Pages that are created in this tutorial are subject to custom workflow that is attached to the page template. Before the SiteAdmin creates the page template on the authoring server, the SiteAdmin creates the custom workflow to be used with new pages.

About this task

The custom workflow is stored in the Web Content library and is a simple, three-stage workflow that contains the following stages: Draft, Review, Publish. A Draft and Publish stage is available for immediate use. Therefore you only need to create the Review stage.

Table 4. Workflow stage access

Workflow stage	Access	Group	Description
Draft (already exists)	**Grant Manager Access**	ContentEditors	Enables authors to edit content that is in the draft state.
	Grant Reviewer Access	ContentEditors	Enables authors to move items to the next stage in the workflow.
Publish (already exists)	**Grant Draft Creator Access**	ContentEditors	Enables authors to create drafts of published content.
Review	**Grant User Access**	ContentEditors	Enables authors to view the status of content in the review stage.
	Grant Reviewer Access	ContentReviewers	Ensures that only reviewers can submit draft content for approval.

Procedure

1. Log in to the authoring server as SiteAdmin.
2. Create the Review workflow stage.
   a. Click the **Applications menu** and then click **Content**.
   b. Click the **Web Content Authoring** tab.
   c. Click **Web Content** > **Workflow Items** > **Workflow Stages**.
   d. Click **New** > **Workflow Stage**.
   e. Enter **Review Stage** for the workflow stage name.
   f. Click **Save and Close**.
3. Edit the access for Draft workflow stage.
   a. Click the **Draft** workflow stage.
   b. Click **Edit**.
   c. Expand **Workflow Defined Access** section.
   d. Click **Grant Manager Access**.
   e. Type `Co` in the search field and click **Search**.
   f. Select ContentEditors and click **Add**. Then click **OK**.
   g. Click **Grant Reviewer Access**.
   h. Type `Co` in the search field and click **Search**.

i. Select ContentReviewers and click **Add**. Then click **OK**.
j. Click **Save and Close**.

4. Edit the access for Publish workflow stage.
   a. Click the **Publish** workflow stage.
   b. Click **Edit**.
   c. Expand **Workflow Defined Access** section.
   d. Click **Grant Draft Creator Access**.
   e. Type `Co` in the search field and click **Search**.
   f. Select ContentEditors and click **Add**. Then click **OK**.
   g. Click **Save and Close**.
5. Edit the access for Review workflow stage.
   a. Click the **Review** workflow stage.
   b. Click **Edit**.
   c. Expand **Workflow Defined Access** section.
   d. Click **Grant User Access**.
   e. Type `Co` in the search field and click **Search**.
   f. Select ContentEditors and click **Add**. Then click **OK**.
   g. Click **Grant Reviewer Access**.
   h. Type `Co` in the search field and click **Search**.
   i. Select ContentReviewers and click **Add**. Then click **OK**.
   j. Click **Save and Close**.

   In the following example screen capture shows the access settings for Review workflow stage. The ContentEditors group is listed with User access, and the ContentReviewers group is listed with Approver access.

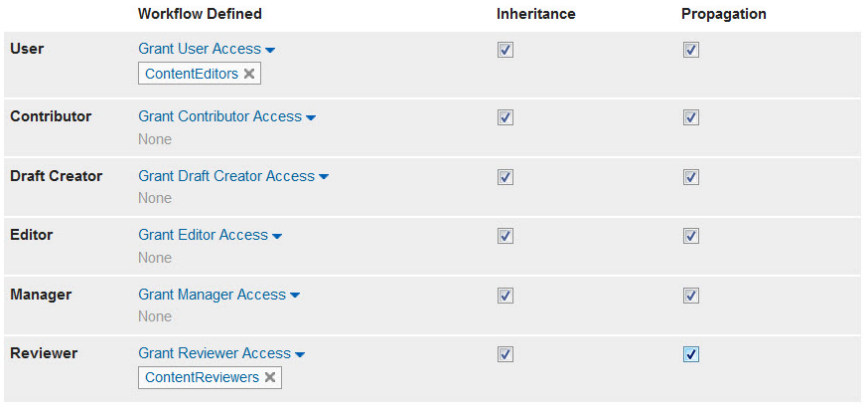


Figure 2. Screen capture of access settings for Review workflow stage.

6. Create the custom workflow named Workflow1.
   a. Click **Web Content** in the breadcrumb.
   b. Click **Workflow Items** > **Workflows**.
   c. Click **New** > **Workflow**.
   d. Enter **Workflow1** for the workflow name.
   e. In the **Workflow Properties** section, click **Add Workflow Stages**.
   f. Add the following stages: Draft Stage, Review Stage, Publish Stage.

g. Use the up and down arrows to order them accordingly: Draft Stage, Review Stage, Publish Stage. The following example screen capture shows the workflow of these stages.

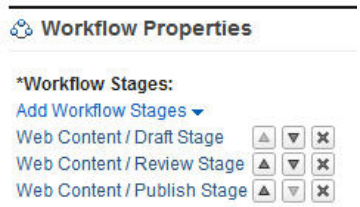


Figure 3. Screen capture of workflow stages in the Workflow1 workflow.

h. Save your changes.

Step 4. Create page template

SiteAdmin creates a page template that content authors can use to create pages.

About this task

Because page changes need to be approved before they are published in this tutorial, the SiteAdmin specifies a custom workflow (Workflow1) when they set page properties for the template. When a content author creates a page from the template, changes to the page are subject to the Workflow1 workflow.

Procedure

1. From the **Site** menu, click **Home** to go to the default site.
2. Click **Projects**.

   Tip: You can set a workflow only on items that are in a draft state. Because of this behavior, you must create the template as a draft in the context of a project. By working in a project, you can also verify the workflow to make sure that the stages behave as expected.
3. Type `New Template` and click **Create**.
   The new project is shown as the active project.

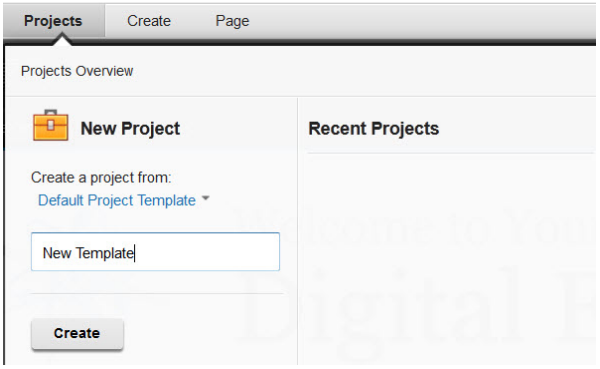


Figure 4. Screen capture of the open **Projects** *tab.*

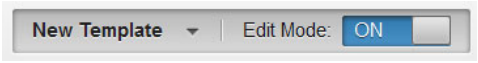


Figure 5. Screen capture of the active project indicator.

4. Create the page template.
   a. Turn on **Edit Mode**.
   b. Click the **Administration menu** icon and then click **Portal User Interface**.

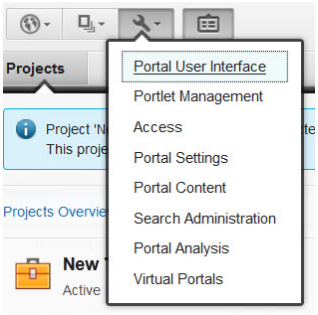


Figure 6. Screen capture of the **Administration menu**.

   c. Click **Page Templates**
   d. From the toolbar click **Create** > **Page**.
   e. Use the Basic template and type `Template1` as the page title of the new page template. You must be in Edit mode.
   f. Click **Create Page**. Parentheses around the page name indicate that the page is in a draft state. Close the toolbar in order to view the page.

Figure 7. Screen capture of the Template1 page.

5. Add the custom workflow to the page template.
   a. From the toolbar click **Page** > **General**.
   b. Click **Edit Page Properties**.

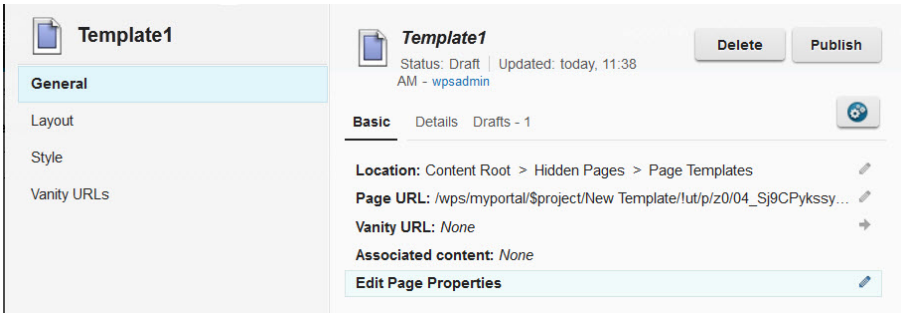


Figure 8. Screen capture of General section of the Page tab.

c. Click the **Security** tab.

d. For the Workflow, click **Select**. Click the Web Content library and select **Workflow1** and click **OK**.

   As soon as you select the workflow, the page draft enters the selected workflow. You can confirm the current workflow stage by examining the workflow properties. In this case, the Draft stage is the current stage and the Review stage is the next stage.

e. Close the Manage Page Properties dialog.

6. Complete the workflow for the draft template. Before you can publish the project and make the new page template available to users, you must move the template through the workflow.

   a. On the General page, click **Submit for Review**.

      Note: The **Submit for Review** button is only available to users that have Approver access on the current workflow stage.

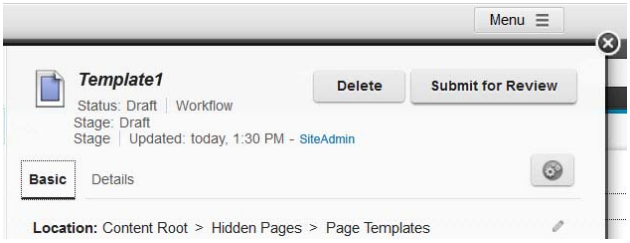


Figure 9. Screen capture of the Submit for Review button in the General view for the Page tab.

When you click **Submit for Review**, the draft item progresses to the next stage in the current workflow. In this scenario, the template moves to the Review Stage of the Workflow1 workflow. The General view displays the current workflow stage.

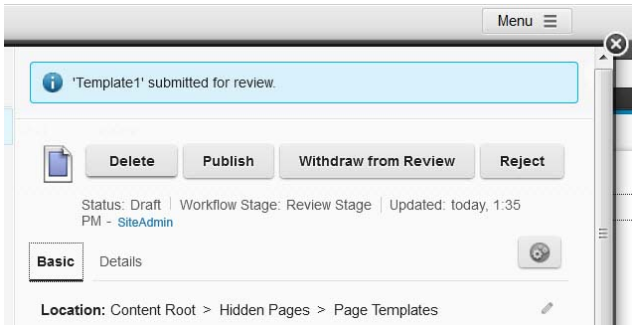


Figure 10. Screen capture of the General view of the Template1 page, showing the updated workflow stage.

b. Click **Publish** to move the template into the publish state.

7. Click **Projects**. With the **New Template** project as the current project, click **Publish**.

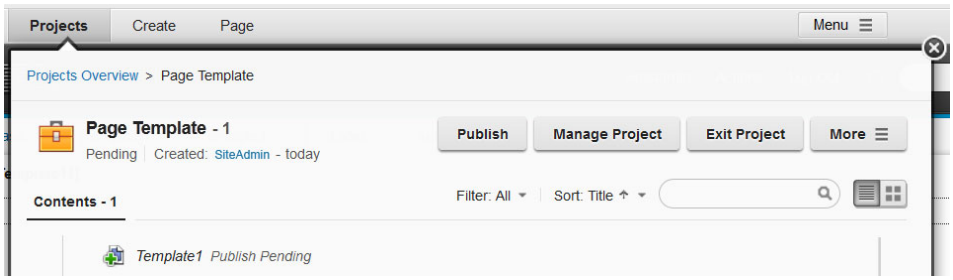


Figure 11. Screen capture of the General view of the Template1 page, showing the updated workflow stage.

8. After the project is published, you are taken out of the project context and restored to the published site. Template1 is listed with the other page templates on the Create Page tab.

What to do next

Important: Workflow changes are not dynamically applied to page templates that use the workflow. If you change the workflow or its stages after you publish the project that contains the template, repeat the steps to set the workflow:

1. Create a project.
2. Edit the page template, and specify the workflow again on the **Security** tab in the page properties.
3. Move the template through the workflow, and publish the project.

Step 5. Create page from template and submit changes for review

After SiteAdmin adds the Template1 page template, content authors can create pages from the template.

About this task

In this scenario, Author1 is responsible for the section of the website that is identified by a page called **News**. The **News** page itself was created by SiteAdmin and was based on the Template1 page template.

Procedure

1. Log in as Author1.
2. Create a project to manage updates.
   a. Open the toolbar and click **Projects**.
   b. Enter **News Section Update** as the project name.
   c. Click **Create**.
   d. Click **Manage Project**.
   e. Click **Edit**. Expand the **Approval** section and click **Add**.
   f. Search for Approver1 and add the user. When the project changes are reviewed and submitted, Approver1 can do the final approval of the complete project.

      Note on project approval: This review step is a separate review from the review of individual items in the project. Even if item workflow is complete for all items in the project, the updates cannot be published until the project is approved.
   g. On the **Properties** tab, expand the **Access** section.
   h. For the Editor access level, click **Grant Editor Access**.
   i. Search for the ContentReviewers group and add it. This access is needed to enable the reviewer to view the items in the project and do the approval steps.
   j. Click **Save and Close**.
3. Create the **News** page.
   a. Go to the **Home** page.
   b. From the toolbar, click **Create** > **Page**.
   c. Select the **Template1** page template, and type **News** for the name of the page.
   d. Click **Create Page**.

   Pages that you create in the portal are stored in the Portal Site library. For example, in this tutorial, when Author1 creates the **News** page, a corresponding site area is created in the Portal Site library. The path to the portal page site area includes the entire page hierarchy up to the content root:

   ```
   Content > Content Root > Home > News
   ```
4. Add the News page to the project. You can use access settings on the Publish stage of your workflow to enable authors to create drafts of published content by granting authors Approver access. This approach ensures that authors cannot directly edit published content. In this scenario, the ContentEditors group has Approver access on the Publish stage of the Workflow1 workflow. As Author1 is a member of the ContentEditors group, Author1 can create a draft of the **News** page.
   a. From the open project, News Section Update, click **More** > **Add to Project**.
   b. Click **Portal SiteContentContent RootHome**.
   c. Check **News** and click **Finish**.
   d. Close the Projects.
5. Create the **Business** page.
   a. Go to the **News** page.
   b. From the toolbar, click **Create** > **Page**.

Note: As Author1 has access only to the **News** page, the **Page** tab only lets you create a child page.

c. Select the **Template1** page template, and type **Business** for the name of the page.

d. Click **Create Page**.

Pages that you create in the portal are stored in the Portal Site library. For example, in this tutorial, when Author1 creates the **Business** page, a corresponding site area is created in the Portal Site library. The path to the portal page site area includes the entire page hierarchy up to the content root:

```
Content > Content Root > Home > News > Business
```

6. Add content to the **Business** page.
   a. Click **Content**.
   b. Click **Web Content** from the list of categories. The site toolbar shows a list of preconfigured Web Content Viewer portlets that you can add to the page.
   c. Add **Rich Text** and an **Image** to the page. Close the toolbar to see the new content on the page.
7. Click **Projects**. As you update the page and add content, the project information reflects the changes.

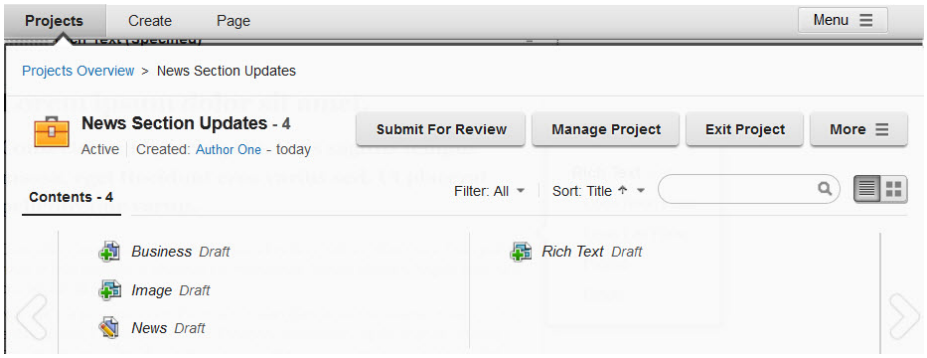


Figure 12. Screen capture of the project properties in the site toolbar, listing the recent items in the project that are updated.

8.

9. When content changes are complete, submit each item in the project for review.

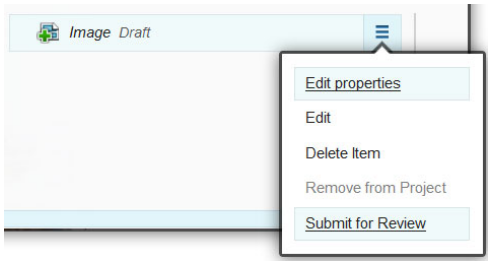


Figure 13. Screen capture of content item menu option listing the Submit for Review option highlighted.

Step 6. Approve and publish changes

After Author1 submits the changes in the **News Section Updates** project for review, Reviewer1 reviews the changes and submits the project for approval. Approver1 then accesses the project and approves the updates for publishing.

When the project is submitted, SiteAdmin publishes the project, and the changes are then available for all users on the authoring server.

Procedure

1. Reviewer1 opens the **News Section Updates** project in edit mode.
   a. Open the site toolbar and click **Projects**.
   b. Select **News Section Updates** from the list of projects.

      Note: The list of projects includes only those projects to which you have access.
   c. For each item, click **Publish**.

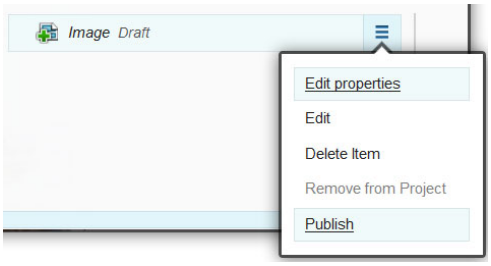


Figure 14. Screen capture of the content item menu with the Publish option highlighted.

2. Then Approver1, can open the **News Section Updates** project and approve the project.

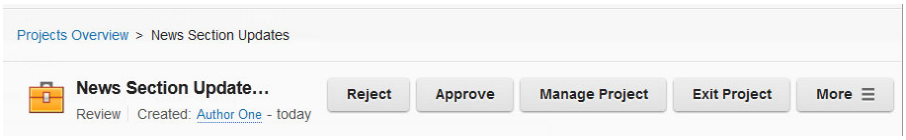


Figure 15. Screen capture of the buttons that are available in project overview to Approver1.

3. When the project changes are approved, SiteAdmin publishes the project by making the **News Update** project the current project and clicking **Publish** from the project menu.

Related tasks:

"Previewing as another user" on page 1922
You can preview changes to your website without logging out and logging on again as another user. This preview capability is used to quickly verify that users with different access levels see only content that they are authorized to see. You can preview changes as a specific user or as an unauthenticated user.

Step 7. Syndicate changes to delivery server

After SiteAdmin publishes the project, the updates are ready for syndication to the delivery server. Although syndication runs at scheduled intervals, SiteAdmin accesses the delivery server and manually starts the syndication process.

Procedure

1. Log in to the delivery server as an administrator.
2. Click the **Administration menu** icon. Then, click **Portal Content** > **Subscribers**.
3. For the syndication relationship to the authoring server, click the **Update subscriber** icon.

Results

After syndication completes, the delivery server is automatically updated with the **Business** page and its associated content.

Figure 16. Screen capture of the delivery server, with the new **Business** *page shown.*

Chapter 3. Roadmaps

Review the roadmaps to understand the common deployment patterns that are supported by the configuration wizard.

"Roadmaps for installation and deployment"
Installation and deployment roadmaps are designed for different environments, such as a development or authoring environment. The environment reflects how the server configuration is used. Under each environment is a topology. The topology describes how the server and other elements are arranged. Each roadmap includes a topology and high-level instructions for setting up the selected environment.

"Roadmap: Applying maintenance" on page 88
Portal maintenance is delivered through two mechanisms: individual fixes (iFixes) and combined cumulative fixes. iFixes are provided, when necessary, for severe or security-related bugs. Combined cumulative fixes are delivered on a regular schedule. They provide a mechanism to deliver fixes faster, improve existing features, deliver new features, update documentation, and provide new documentation on a frequent basis. To deliver continuous improvements for your digital experience software, it is recommended that you apply the latest combined cumulative fix to your environment.

"Roadmaps for migration" on page 92
Choose the appropriate migration roadmap for your environment.

"Roadmaps for integration" on page 97
Choose the appropriate integration roadmaps for your environment.

"Roadmaps for web content" on page 98
Some roadmaps to help you create and integrate web content into your site.

Roadmaps for installation and deployment

Installation and deployment roadmaps are designed for different environments, such as a development or authoring environment. The environment reflects how the server configuration is used. Under each environment is a topology. The topology describes how the server and other elements are arranged. Each roadmap includes a topology and high-level instructions for setting up the selected environment.

"Roadmaps for stand-alone servers"
A stand-alone server topology is useful for many environments. While the underlying topology is similar for many environments, the configuration steps to achieve a desired environment varies.

"Roadmap: Web content servers" on page 83
A web content server is ideal to create presentation templates, content components, and other presentation elements of your website. In this roadmap, the web server, database, and user registry software are distributed to different physical servers.

Roadmaps for stand-alone servers

A stand-alone server topology is useful for many environments. While the underlying topology is similar for many environments, the configuration steps to achieve a desired environment varies.

Roadmaps for WebSphere Portal Express are also available in IBM Knowledge Center. For WebSphere Portal Express roadmaps, see WebSphere Portal Express: Roadmaps for stand-alone servers.

"Roadmap: Demonstration environment"
This roadmap is ideal for running product, feature, and application demonstrations. After the installation, you have an Apache Derby database and a default file-based repository. You might not need to make any additional changes to your demonstration environment. If you need to demonstrate a specific database or user registry, use the Stand-alone server roadmap.

"Roadmap: Portlet and theme development environment" on page 72
A portlet and theme development server is ideal to develop and test portlets and themes.

"Roadmap: Test or small production environment" on page 75
A stand-alone server topology is ideal for a test or small production environment. In this roadmap, the web server, database, and user registry software are distributed to different physical servers.

"Roadmap: Failover environment" on page 79
Set up your IBM WebSphere Portal Express environment as an idle standby configuration for failover. Your idle standby cluster server is not operational to service user transactions or to query workloads. The idle standby cluster server becomes active only if the primary node fails.

Roadmap: Demonstration environment

This roadmap is ideal for running product, feature, and application demonstrations. After the installation, you have an Apache Derby database and a default file-based repository. You might not need to make any additional changes to your demonstration environment. If you need to demonstrate a specific database or user registry, use the Stand-alone server roadmap.

Who should use this roadmap

Use this roadmap if you are an organization with the following requirements:

- An organization that needs to explore new features or functions.
- An organization that needs to demonstrate Version 8.5 to customers, clients, or colleagues.
- An organization that needs an environment to demonstrate applications and designs.

Topology diagram

A demonstration environment is a portal server with a local database. Since Apache Derby is installed, configured, and ready for use, a database is not included in the topology diagram.

Preparing for the installation process

Gather information and software before you install WebSphere Portal Express.

Procedure

1. Check requirements.

   Documentation resource: Detailed system requirements

2. Get the software.

   Documentation resource: “Getting the software” on page 135

Installing the Exceptional Digital Experience

Installing WebSphere Portal Express involves preparing your operating system, installing or upgrading the installation manager, and running the installation program.

About this task

Documentation resource: “Installing the digital experience software” on page 167

Applying the latest cumulative fix

Portal maintenance is delivered through two mechanisms: individual fixes (iFixes) and combined cumulative fixes. iFixes are provided, when necessary, for severe or security-related bugs. Combined cumulative fixes are delivered on a regular schedule. They provide a mechanism to deliver fixes faster, improve existing features, deliver new features, update documentation, and provide new documentation on a frequent basis. To deliver continuous improvements for your digital experience software, it is recommended that you apply the latest combined cumulative fix to your environment.

About this task

Documentation resource: “Roadmap: Applying maintenance” on page 88

Tuning the servers in your environment

Tuning the servers is important to the performance of your portal environment. WebSphere Portal Express is not tuned for a production environment after installation and deployment. Your database needs tuning for improved performance. You can organize your database now or soon after you finish your configuration. You need to tune and maintain your database on a regular basis.

Procedure

Run the performance tuning tool to complete an initial tuning of your servers.

Documentation resource: "Portal server performance tuning tool" on page 240

Roadmap: Portlet and theme development environment

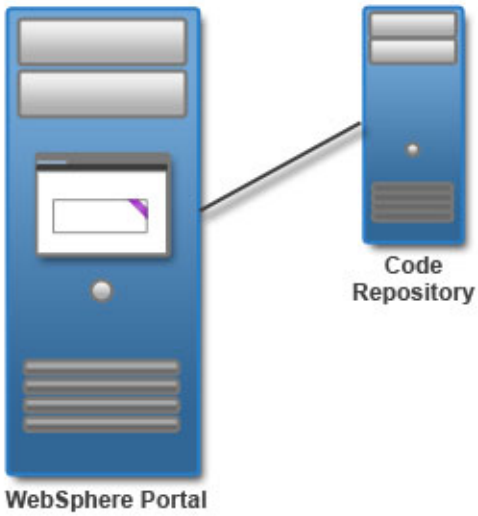
A portlet and theme development server is ideal to develop and test portlets and themes.

Who should use this roadmap

Use this roadmap if you are an organization that needs an environment to develop and test your portlets and themes before you go live to the server. Do not use this roadmap to develop web content. Go to "Roadmap: Web content servers" on page 83 to develop web content.

Topology diagram

The topology for a portlet and theme environment is the same as the demonstration environment. It includes a portal server with a local database. However, the software configuration is different. The topology diagram depicts a portal server that is connected to a code repository for group development. While the topology diagram depicts a server, the portlet and theme environment might be configured on a notebook.



Preparing for the installation process

Gather information and software before you install WebSphere Portal Express.

Procedure

1. Check requirements.

   Documentation resource: Detailed system requirements

2. Get the software.

   Documentation resource: “Getting the software” on page 135

Installing prerequisites

Before you install WebSphere Portal Express, install any prerequisites that are necessary for your environment. You can use existing prerequisite software installations. Verify that your existing version is supported. If it does not, upgrade to the appropriate version.

Procedure

1. Prepare a database server.

   Documentation resource: “Installing and preparing the database software” on page 149

2. Prepare a user registry.

   Documentation resource: “Preparing the user registry software” on page 158

Installing the Exceptional Digital Experience

Installing WebSphere Portal Express involves preparing your operating system, installing or upgrading the installation manager, and running the installation program.

About this task

Documentation resource: “Installing the digital experience software” on page 167

Applying the latest cumulative fix

Portal maintenance is delivered through two mechanisms: individual fixes (iFixes) and combined cumulative fixes. iFixes are provided, when necessary, for severe or security-related bugs. Combined cumulative fixes are delivered on a regular schedule. They provide a mechanism to deliver fixes faster, improve existing features, deliver new features, update documentation, and provide new documentation on a frequent basis. To deliver continuous improvements for your digital experience software, it is recommended that you apply the latest combined cumulative fix to your environment.

About this task

Documentation resource: “Roadmap: Applying maintenance” on page 88

Transferring your database

After you install your web experience, Apache Server is your available database. Depending on your requirements, you might need to transfer to a different database. The **Database Transfer** configuration option in the Configuration Wizard assigns users and permissions, creates databases, obtains support for database collation, and transfers your database.

Before you begin

Log in to WebSphere Portal Express to verify that you have a working portal:

`http://`*`hostname.example.com`*`:`*`10039`*`/wps/portal,`
`where` *`hostname.example.com`* `is the fully qualified`
`host name of the server where Portal is running and` *`10039`* `is`
`the default transport port that is created by WebSphere Application Server. The port number might be different for your environment.`

Procedure

1. To get the latest updates for the wizard, apply the most recent combined cumulative fix. For more information about applying the latest fix pack, see WebSphere Portal and Web Content Manager V8.5.0.0 combined cumulative fix overview.

   Note: Skip this step, if you have the most recent fix pack applied.
2. Access the Configuration Wizard. Go to `http://your_server:10200/ibm/wizard`.
3. Log in to the Configuration Wizard with the administrative ID for the configuration wizard profile, `cw_profile`.

   Note: The wizard user interface might not be available in all languages. If the language is not currently supported, you might see the English version. For details on the supported languages for all of the WebSphere Portal Express user interfaces, see "Supported languages" on page 3622.
4. Select **Set Up a Stand-alone Server** > **Database Transfer**.
5. Provide information about your environment.
6. Save your configuration settings.
7. Choose one of the following options:
   - Click **Download Files** to run the steps remotely.
   - Click **Run All Steps** to run the steps locally.
8. Log in to WebSphere Portal Express to verify that you have a working portal server.

Enabling federated security

After you install your web experience, a default file-based repository is your available user registry. Depending on your requirements, you might need to enable a federated LDAP user registry.

About this task

Note: If you set **Use Administrator IDs stored in your LDAP user registry** to `yes`, the WebSphere Application Server and WebSphere Portal Express user IDs and passwords are changed to the LDAP user ID and password. If you do not want to change both user IDs and passwords to match the LDAP user ID and password, set this value to `no`. After you configure your LDAP user registry, you can manually change the user IDs and passwords.

Documentation resource: "Updating user ID and passwords" on page 1670

Procedure

1. To get the latest updates for the wizard, apply the most recent combined cumulative fix. For more information about applying the latest fix pack, see WebSphere Portal and Web Content Manager V8.5.0.0 combined cumulative fix overview.

   Note: Skip this step, if you have the most recent fix pack applied.
2. Access the Configuration Wizard. Go to `http://your_server:10200/ibm/wizard`.
3. Log in to the Configuration Wizard with the administrative ID for the configuration wizard profile, `cw_profile`.

   Note: The wizard user interface might not be available in all languages. If the language is not currently supported, you might see the English version. For

details on the supported languages for all of the WebSphere Portal Express user interfaces, see “Supported languages” on page 3622.

4. Select **Set Up a Stand-alone Server** > **Enable Federated Security**.

   Note: If you set **Use Administrator IDs stored in your LDAP user registry** to `yes`, the WebSphere Application Server and WebSphere Portal Express user IDs and passwords are changed to the LDAP user ID and password. If you do not want to change both user IDs and passwords to match the LDAP user ID and password, set this value to `no`. After you configure your LDAP user registry, you can manually change the user IDs and passwords.

   Documentation resource: “Updating user ID and passwords” on page 1670
5. Provide information about your environment.
6. Save your configuration settings.
7. Choose one of the following options:
   - Click **Download Files** to run the steps remotely.
   - Click **Run All Steps** to run the steps locally.
8. Log in to WebSphere Portal Express to verify that you have a working portal server.

Tuning the servers in your environment

Tuning the servers is important to the performance of your portal environment. WebSphere Portal Express is not tuned for a production environment after installation and deployment. Your database needs tuning for improved performance. You can organize your database now or soon after you finish your configuration. You need to tune and maintain your database on a regular basis.

Procedure

Run the performance tuning tool to complete an initial tuning of your servers.

Documentation resource: “Portal server performance tuning tool” on page 240

Change to developer mode

Change your stand-alone server to developer mode to improve startup performance.

About this task

Documentation resources: “Changing to developer mode” on page 2511

Roadmap: Test or small production environment

A stand-alone server topology is ideal for a test or small production environment. In this roadmap, the web server, database, and user registry software are distributed to different physical servers.

Remember: Ensure that you configure the web server plug-in after you transfer your database.

Who should use this roadmap

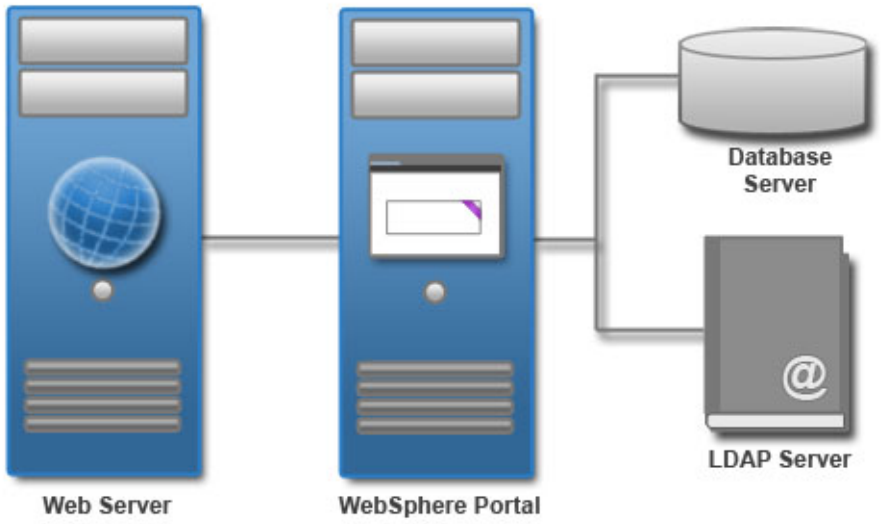
Use this roadmap if you are an organization with the following requirements:

- An organization that needs an environment to test their applications and designs before they go live to the server.
- An organization that does not need a clustered environment for failover or high availability.

- An organization with limited server resources that wants to set up a department server or small website.

Topology diagram

A stand-alone server topology is versatile. It is the foundation for a portal farm, authoring environment, test or rendering environment, small department-level deployments, and more. A typical stand-alone portal server topology includes a remote database and LDAP server. For many environments, it also includes a web server to direct incoming traffic.



Preparing for the installation process

Gather information and software before you install WebSphere Portal Express.

Procedure

1. Check requirements.

   Documentation resource: Detailed system requirements
2. Get the software.

   Documentation resource: “Getting the software” on page 135

Installing prerequisites

You can use existing prerequisite software installations. Verify that your existing version is supported. If it is not, upgrade to the appropriate version. Otherwise, install a web server, database server, and user registry server. Typically the database and user registry servers are already installed and configured. However, there might be specific configuration steps that are required to integrate them with the portal server.

Procedure

1. Install web server.

   Documentation resource: “Preparing a remote web server” on page 166
2. Prepare a database server.

   Documentation resource: “Installing and preparing the database software” on page 149
3. Prepare a user registry.

Documentation resource: “Preparing the user registry software” on page 158

Installing the Exceptional Digital Experience

Installing WebSphere Portal Express involves preparing your operating system, installing or upgrading the installation manager, and running the installation program.

About this task

Documentation resource: “Installing the digital experience software” on page 167

Applying the latest cumulative fix

Portal maintenance is delivered through two mechanisms: individual fixes (iFixes) and combined cumulative fixes. iFixes are provided, when necessary, for severe or security-related bugs. Combined cumulative fixes are delivered on a regular schedule. They provide a mechanism to deliver fixes faster, improve existing features, deliver new features, update documentation, and provide new documentation on a frequent basis. To deliver continuous improvements for your digital experience software, it is recommended that you apply the latest combined cumulative fix to your environment.

About this task

Documentation resource: “Roadmap: Applying maintenance” on page 88

Setting up a stand-alone server

Start the configuration wizard to set up your stand-alone server. First, transfer your database. The **Database Transfer** configuration option in the Configuration Wizard assigns users and permissions, creates databases, obtains support for database collation, and transfers your database. After you transfer your database, enable your federated LDAP user registry.

Before you begin

Log in to WebSphere Portal Express to verify that you have a working portal:
`http://hostname.example.com:10039/wps/portal`,
where *hostname.example.com* is the fully qualified
host name of the server where Portal is running and *10039* is
the default transport port that is created by WebSphere Application Server. The port number might be different for your environment.

Procedure

1. To get the latest updates for the wizard, apply the most recent combined cumulative fix. For more information about applying the latest fix pack, see WebSphere Portal and Web Content Manager V8.5.0.0 combined cumulative fix overview.

   Note: Skip this step, if you have the most recent fix pack applied.
2. Access the Configuration Wizard. Go to `http://your_server:10200/ibm/wizard`.
3. Log in to the Configuration Wizard with the administrative ID for the configuration wizard profile, `cw_profile`.

   Note: The wizard user interface might not be available in all languages. If the language is not currently supported, you might see the English version. For

details on the supported languages for all of the WebSphere Portal Express user interfaces, see "Supported languages" on page 3622.

4. Select **Set Up a Stand-alone Server** > **Database Transfer**.
5. Provide information about your environment.
6. Save your configuration settings.
7. Choose one of the following options:
   - Click **Download Files** to run the steps remotely.
   - Click **Run All Steps** to run the steps locally.
8. Optional: If you changed the context root during the installation or configuration of IBM WebSphere Portal Express, then you must complete the optional next step from the Configuration Wizard to update parameters with the new context path after you complete the Create a Deployment Manager configuration option. For more information about this configuration option and completing the next steps, see Create a deployment manager.
9. Log in to WebSphere Portal Express to verify that you have a working portal server.
10. Select **Set Up a Stand-alone Server** > **Enable Federated Security**.

    Note: If you set **Use Administrator IDs stored in your LDAP user registry** to `yes`, the WebSphere Application Server and WebSphere Portal Express user IDs and passwords are changed to the LDAP user ID and password. If you do not want to change both user IDs and passwords to match the LDAP user ID and password, set this value to `no`. After you configure your LDAP user registry, you can manually change the user IDs and passwords.

    Documentation resource: "Updating user ID and passwords" on page 1670
11. Provide information about your environment.
12. Save your configuration settings.
13. Choose one of the following options:
    - Click **Download Files** to run the steps remotely.
    - Click **Run All Steps** to run the steps locally.
14. Log in to WebSphere Portal Express to verify that you have a working portal server.

Configuring the web server

Procedure

Move the web server plug-in from the WebSphere Application Server to the web server.

Documentation resource: "Web servers" on page 242

Tuning the servers in your environment

Tuning the servers is important to the performance of your portal environment. WebSphere Portal Express is not tuned for a production environment after installation and deployment. Your database needs tuning for improved performance. You can organize your database now or soon after you finish your configuration. You need to tune and maintain your database on a regular basis.

Procedure

1. Run the performance tuning tool to complete an initial tuning of your servers.

   Documentation resource: "Portal server performance tuning tool" on page 240

2. Check the tuning guide for more instructions. Use the tuning guide for the previous product version when the tuning guide for the current release is unavailable.

   Documentation resource: Performance tuning guide

Next steps

Depending on the choices that you made during the installation and set up, there are additional tasks to configure your environment.

The following options are available to continue configuring your environment:

Configure global settings
Documentation resource: "Configuring portal behavior" on page 248

Change the default portal Uniform Resource Identifier (URI)
If you changed the context root on the Configuration for IBM WebSphere Portal: Profile configuration details: Advanced panel during installation: "Completing the portal URI change started during installation" on page 362

If you want to change the context root after installation: "Changing the portal URI after an installation" on page 368

Adapt the attribute configuration to match the LDAP server
Documentation resource: "Adding more attributes to VMM" on page 568

Configure syndication
Documentation resource: "Syndication" on page 448

Update your user registry
Documentation resource: "User registry" on page 562

Configure your search
Documentation resource: "" on page 667

Roadmap: Failover environment

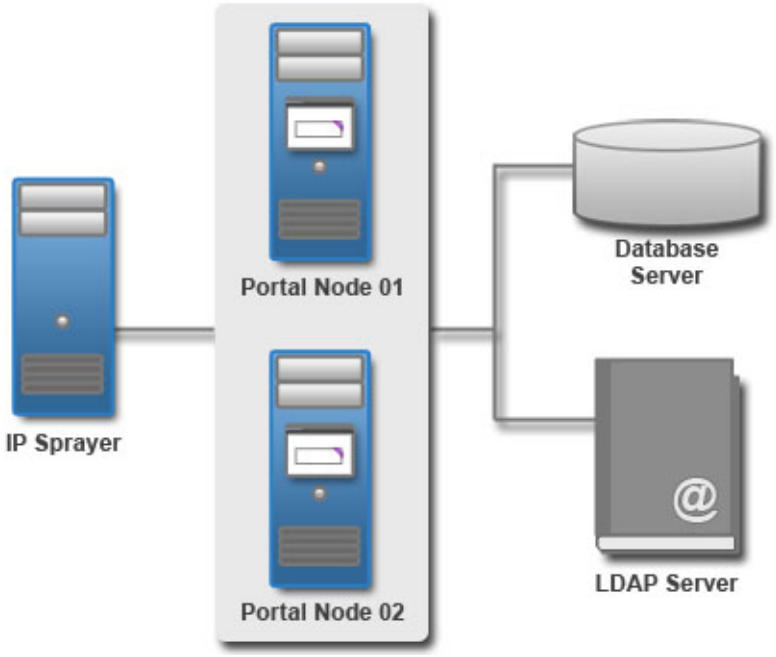
Set up your IBM WebSphere Portal Express environment as an idle standby configuration for failover. Your idle standby cluster server is not operational to service user transactions or to query workloads. The idle standby cluster server becomes active only if the primary node fails.

Who should use this roadmap

Use this roadmap if you installed WebSphere Portal Express and want a failover configuration.

Topology diagram

The idle-standby topology includes a portal cluster, database server, and user registry server. The cluster includes two nodes. The second node is only used when the primary node fails.



Preparing for the installation process

Gather information and software before you install WebSphere Portal Express.

Procedure

1. Check requirements.

   Documentation resource: Detailed system requirements
2. Get the software.

   Documentation resource: “Getting the software” on page 135

Installing prerequisites

You can use existing prerequisite software installations. Verify that your existing version is supported. If it is not, upgrade to the appropriate version. Otherwise, install a web server, database server, and user registry server. Typically the database and user registry servers are already installed and configured. However, there might be specific configuration steps that are required to integrate them with the portal server.

Procedure

1. Install web server.

   Documentation resource: “Preparing a remote web server” on page 166
2. Prepare a database server.

   Documentation resource: “Installing and preparing the database software” on page 149
3. Prepare a user registry.

   Documentation resource: “Preparing the user registry software” on page 158

Installing the Exceptional Digital Experience

Installing WebSphere Portal Express involves preparing your operating system, installing or upgrading the installation manager, and running the installation program.

About this task

Documentation resource: “Installing the digital experience software” on page 167

Setting up an idle standby

Start the configuration wizard to get a running portal and to configure the portal with your database and user registry.

Before you begin

Log in to WebSphere Portal Express to verify that you have a working portal:

`http://hostname.example.com:10039/wps/portal`,
where *hostname.example.com* is the fully qualified
host name of the server where Portal is running and *10039* is
the default transport port that is created by WebSphere Application Server. The port number might be different for your environment.

Procedure

1. Access the Configuration Wizard. Go to `http://your_server:10200/ibm/wizard`.
2. Log in to the Configuration Wizard with the administrative ID for the configuration wizard profile, `cw_profile`.

   Note: The wizard user interface might not be available in all languages. If the language is not currently supported, you might see the English version. For details on the supported languages for all of the WebSphere Portal Express user interfaces, see “Supported languages” on page 3622.

Complete the following steps to transfer your database:

3. Select **Set Up a Cluster** > **Database Transfer**.
4. Provide information about your environment.
5. Save your configuration settings.
6. Choose one of the following options:
   - Click **Download Files** to run the steps remotely.
   - Click **Run All Steps** to run the steps locally.
7. Log in to WebSphere Portal Express to verify that you have a working portal server.

Complete the following steps to create your deployment manager:

8. Select **Set Up a Cluster** > **Create a Deployment Manager**.
9. Provide information about your environment.
10. Save your configuration settings.
11. Choose one of the following options:
    - Click **Download Files** to run the steps remotely.
    - Click **Run All Steps** to run the steps locally.
12. Log in to WebSphere Portal Express to verify that you have a working portal server.

Complete the following steps to create a cluster node:

13. Select **Set Up a Cluster** > **Create a Cluster**.
14. Provide information about your environment.
15. Save your configuration settings.
16. Choose one of the following options:
    - Click **Download Files** to run the steps remotely.

- Click **Run All Steps** to run the steps locally.

17. Log in to WebSphere Portal Express to verify that you have a working portal server.

Complete the following steps to enable federated security:

18. **Set Up a Cluster** > **Enable Federated Security**.

    Note: If you set **Use Administrator IDs stored in your LDAP user registry** to `yes`, the WebSphere Application Server and WebSphere Portal Express user IDs and passwords are changed to the LDAP user ID and password. If you do not want to change both user IDs and passwords to match the LDAP user ID and password, set this value to `no`. After you configure your LDAP user registry, you can manually change the user IDs and passwords.

    Documentation resource: "Updating user ID and passwords" on page 1670

19. Provide information about your environment.
20. Save your configuration settings.
21. Choose one of the following options:
    - Click **Download Files** to run the steps remotely.
    - Click **Run All Steps** to run the steps locally.
22. Log in to WebSphere Portal Express to verify that you have a working portal server.

Complete the following steps to create an idle standby:

23. Install WebSphere Portal Express on the idle standby.

    Documentation resource: "Installing the digital experience software" on page 167.

    Tip: For additional nodes, you only need to install the WebSphere Portal Express product binary files. Therefore, on the Features screen of the IBM Installation Manager, ensure that **Portal Server Profile** is not selected.

24. **Set Up a Cluster** > **Create an Additional Cluster Node**.
25. Provide information about your environment.
26. Save your configuration settings.
27. Choose one of the following options:
    - Click **Download Files** to run the steps remotely.
    - Click **Run All Steps** to run the steps locally.
28. Log in to WebSphere Portal Express to verify that you have a working portal server.
29. Complete the following steps to configure the Idle Standby node as a backup server:
    a. Log on to the deployment manager WebSphere Integrated Solutions Console.
    b. Select **Servers** > **Server Types** > **WebSphere application servers** > *server_name* > **Web server plug-in properties**.
    c. From the **Server Role** menu, select **Backup**.
    d. Apply and save the changes.
    e. Log out of the deployment manager WebSphere Integrated Solutions Console.

Configuring the web server
Procedure

Move the web server plug-in from the WebSphere Application Server to the web server.

Documentation resource: "Web servers" on page 242

Tuning the servers in your environment

Tuning the servers is important to the performance of your portal environment. WebSphere Portal Express is not tuned for a production environment after installation and deployment. Your database needs tuning for improved performance. You can organize your database now or soon after you finish your configuration. You need to tune and maintain your database on a regular basis.

Procedure

1. Run the performance tuning tool to complete an initial tuning of your servers.

   Documentation resource: "Portal server performance tuning tool" on page 240
2. Check the tuning guide for more instructions. Use the tuning guide for the previous product version when the tuning guide for the current release is unavailable.

   Documentation resource: Performance tuning guide

Next steps

Depending on the choices that you made during the installation and set up, there are additional tasks to configure your environment.

The following options are available to continue configuring your environment:

Configure global settings
Documentation resource: "Configuring portal behavior" on page 248

Adapt the attribute configuration to match the LDAP server
Documentation resource: "Adding more attributes to VMM" on page 568

Configure syndication
Documentation resource: "Syndication" on page 448

Configure search
Documentation resource: "Configuring search in a cluster" on page 662

Update your user registry
Documentation resource: "User registry" on page 562

Roadmap: Web content servers

A web content server is ideal to create presentation templates, content components, and other presentation elements of your website. In this roadmap, the web server, database, and user registry software are distributed to different physical servers.

Who should use this roadmap

Use this roadmap if you are an organization with the following requirements:

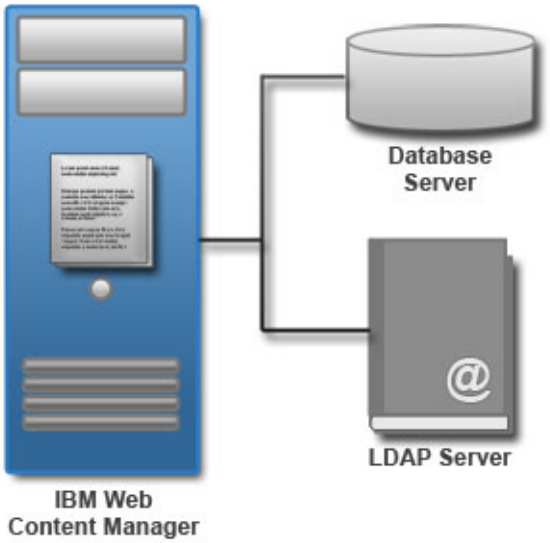
- An organization that needs an environment to create and test your web content before they publish to other staging or production servers
- An organization that needs to create presentation templates (Authoring or Presentation) for content authors and content rendering

- An organization that needs to create the site structure and entitlements
- An organization that needs to develop content with HTML, CSS, JavaScript, JSP, or Java.
- An organization needs a staging server to perform final quality tests before publishing.
- An organization needs a server to publish content to end users.

Important: Developing or publishing web content requires a supported database other than Apache Server. An Apache Server database is available after installation. Therefore, you must transfer to a supported database; for example DB2 or Oracle.

Topology diagram

The topology for a web content server includes a remote database and LDAP server. The topology depicts an IBM Web Content Manager server, instead of a portal server. When you install IBM WebSphere Portal, it includes Web Content Manager. The configuration steps for the web content development server are different from a basic stand-alone server. The web content server configuration includes syndication with the rendering or test server.



Preparing for the installation process

Gather information and software before you install WebSphere Portal Express.

Procedure

1. Check requirements.

   Documentation resource: Detailed system requirements
2. Get the software.

   Documentation resource: “Getting the software” on page 135

Installing prerequisites

Before you install WebSphere Portal Express, install any prerequisites that are necessary for your environment. You can use existing prerequisite software installations. Verify that your existing version is supported. If it does not, upgrade to the appropriate version.

Procedure

1. Prepare a database server.

   Documentation resource: “Installing and preparing the database software” on page 149
2. Prepare a user registry.

   Documentation resource: “Preparing the user registry software” on page 158

Installing the Exceptional Digital Experience

Installing WebSphere Portal Express involves preparing your operating system, installing or upgrading the installation manager, and running the installation program.

About this task

Documentation resource: “Installing the digital experience software” on page 167

Applying the latest cumulative fix

Portal maintenance is delivered through two mechanisms: individual fixes (iFixes) and combined cumulative fixes. iFixes are provided, when necessary, for severe or security-related bugs. Combined cumulative fixes are delivered on a regular schedule. They provide a mechanism to deliver fixes faster, improve existing features, deliver new features, update documentation, and provide new documentation on a frequent basis. To deliver continuous improvements for your digital experience software, it is recommended that you apply the latest combined cumulative fix to your environment.

About this task

Documentation resource: “Roadmap: Applying maintenance” on page 88

Transferring your database

After you install your web experience, Apache Server is your available database. Depending on your requirements, you might need to transfer to a different database. The **Database Transfer** configuration option in the Configuration Wizard assigns users and permissions, creates databases, obtains support for database collation, and transfers your database.

Before you begin

Log in to WebSphere Portal Express to verify that you have a working portal:

`http://hostname.example.com:10039/wps/portal,`
`where hostname.example.com is the fully qualified`
`host name of the server where Portal is running and 10039 is`
`the default transport port that is created by WebSphere Application Server. The port number might be different for your environment.`

Procedure

1. To get the latest updates for the wizard, apply the most recent combined cumulative fix. For more information about applying the latest fix pack, see WebSphere Portal and Web Content Manager V8.5.0.0 combined cumulative fix overview.

   Note: Skip this step, if you have the most recent fix pack applied.

2. Access the Configuration Wizard. Go to `http://your_server:10200/ibm/wizard`.
3. Log in to the Configuration Wizard with the administrative ID for the configuration wizard profile, `cw_profile`.

   Note: The wizard user interface might not be available in all languages. If the language is not currently supported, you might see the English version. For details on the supported languages for all of the WebSphere Portal Express user interfaces, see "Supported languages" on page 3622.
4. Select **Set Up a Stand-alone Server** > **Database Transfer**.
5. Provide information about your environment.
6. Save your configuration settings.
7. Choose one of the following options:
   - Click **Download Files** to run the steps remotely.
   - Click **Run All Steps** to run the steps locally.
8. Log in to WebSphere Portal Express to verify that you have a working portal server.

Enabling federated security

After you install your web experience, a default file-based repository is your available user registry. Depending on your requirements, you might need to enable a federated LDAP user registry.

About this task

Note: If you set **Use Administrator IDs stored in your LDAP user registry** to `yes`, the WebSphere Application Server and WebSphere Portal Express user IDs and passwords are changed to the LDAP user ID and password. If you do not want to change both user IDs and passwords to match the LDAP user ID and password, set this value to `no`. After you configure your LDAP user registry, you can manually change the user IDs and passwords.

Documentation resource: "Updating user ID and passwords" on page 1670

Procedure

1. To get the latest updates for the wizard, apply the most recent combined cumulative fix. For more information about applying the latest fix pack, see WebSphere Portal and Web Content Manager V8.5.0.0 combined cumulative fix overview.

   Note: Skip this step, if you have the most recent fix pack applied.
2. Access the Configuration Wizard. Go to `http://your_server:10200/ibm/wizard`.
3. Log in to the Configuration Wizard with the administrative ID for the configuration wizard profile, `cw_profile`.

   Note: The wizard user interface might not be available in all languages. If the language is not currently supported, you might see the English version. For details on the supported languages for all of the WebSphere Portal Express user interfaces, see "Supported languages" on page 3622.
4. Select **Set Up a Stand-alone Server** > **Enable Federated Security**.

   Note: If you set **Use Administrator IDs stored in your LDAP user registry** to `yes`, the WebSphere Application Server and WebSphere Portal Express user IDs and passwords are changed to the LDAP user ID and password. If you do not

want to change both user IDs and passwords to match the LDAP user ID and password, set this value to `no`. After you configure your LDAP user registry, you can manually change the user IDs and passwords.

Documentation resource: "Updating user ID and passwords" on page 1670

5. Provide information about your environment.
6. Save your configuration settings.
7. Choose one of the following options:
   - Click **Download Files** to run the steps remotely.
   - Click **Run All Steps** to run the steps locally.
8. Log in to WebSphere Portal Express to verify that you have a working portal server.

Tuning the servers in your environment

Tuning the servers is important to the performance of your portal environment. WebSphere Portal Express is not tuned for a production environment after installation and deployment. Your database needs tuning for improved performance. You can organize your database now or soon after you finish your configuration. You need to tune and maintain your database on a regular basis.

Procedure

Run the performance tuning tool to complete an initial tuning of your servers.

Documentation resource: "Portal server performance tuning tool" on page 240

Configuring the Authoring portlet

Configure your Authoring portlet on your WebSphere Portal Express server.

Procedure

1. Configure extra Authoring portlet parameters.

   Documentation resource: "Further authoring portlet configuration options" on page 395
2. Configure the workflow, profiling, and version control settings.

   Documentation resource: "Web content authoring options" on page 396
3. Configure the Authoring portlet search.

   Documentation resource: "How to configure authoring portlet search" on page 402
4. Configure your server to import large files and images.

   Documentation resource: Importing large files and images
5. Configure your server to avoid time-out issues.

   Documentation resource: "Increase timeouts to prevent save errors" on page 403
6. Configure remote server access so you can link to files and documents on remote content management systems.

   Documentation resource: "Configuring remote server access for links" on page 404
7. Set up support for federated documents.

   Documentation resource: "Setting up support for federated documents" on page 405

Syndication

Use syndication to synchronize content between authoring, staging, and publishing environments.

Procedure

1. Plan your syndication strategies.

   Documentation resource: “Syndication relationships” on page 449
2. Define syndication properties.

   Documentation resource: “Syndication properties” on page 454
3. Tune your syndication strategy to improve performance.

   Documentation resource: “Syndication tuning” on page 456
4. Create your syndication relationships.

   Documentation resource: Creating a syndication relationship by using the Administration Portlet

Roadmap: Applying maintenance

Portal maintenance is delivered through two mechanisms: individual fixes (iFixes) and combined cumulative fixes. iFixes are provided, when necessary, for severe or security-related bugs. Combined cumulative fixes are delivered on a regular schedule. They provide a mechanism to deliver fixes faster, improve existing features, deliver new features, update documentation, and provide new documentation on a frequent basis. To deliver continuous improvements for your digital experience software, it is recommended that you apply the latest combined cumulative fix to your environment.

Use this roadmap to learn more about the following information:

- What the latest combined cumulative fix contains
- How to apply the latest combined cumulative fix
- How to configure and enable new combined cumulative fix features

Who should use this roadmap

Use this roadmap if you:

- Installed WebSphere Portal V8.5 for the first time and want to apply an iFix or the latest combined cumulative fix.
- Need a high-level view of which features or improvements are ready to use by default after you apply the latest combined cumulative fix and which features need additional configuration.

What is continuous delivery

Continuous delivery is short development cycles with continuous integration and automated tests. The result is a releasable product at any time. Continuous delivery has the following benefits:

- Accelerated time to market
- Building the right product
- Improved productivity and efficiency
- Reliable releases
- Improved product quality

- Improved customer satisfaction

The continuous delivery features are delivered with the combined cumulative fixes.

Learning about this combined cumulative fix

Some combined cumulative fixes focus on fixing known issues that are identified by APARs. Other cumulative fixes are focused on new features.

Procedure

1. View a list of APAR fixes delivered a combined cumulative fix.

   Documentation resource: Fixes integrated in WebSphere Portal 8.5.0.0 Combined Cumulative Fixes
2. Get an overview of the features and improvements delivered in the latest combined cumulative fix.

   Documentation resource: “What's new with CF07” on page 2

Applying a combined cumulative fix

To apply a combined cumulative fix, you must update the product files with IBM Installation Manager. You must also update properties files, download the cumulative fix, and run ConfigEngine tasks to apply the changes to each profile in your system. The IBM WebSphere Portal Express V8.5.0.0 combined cumulative fix readme file provides detailed instructions on how to apply a fix and planning information.

Before you begin

CF03 Use the Health Checker tool to identify issues with your installation. You should use the Health Checker tool on your target system before you apply a combined cumulative fix. You can learn more about how to apply the Health Checker tool in the readme instructions.

Individual fixes: Individual fixes (iFixes) are included in the combined cumulative fix. However, if you need a fix before the combined cumulative fix is available, you can install the iFix. iFixes are found on IBM Fix Central (http://www.ibm.com/support/fixcentral/). To install an iFix, download it from Fix Central and use the IBM Installation Manager to update the portal server. An iFix can be installed with the Installation Manager GUI or command line interface. It can also be automated with a response file. Most iFixes also require a command to update your profile. This requirement differs with each fix. Refer to the readme file that comes with the iFix for details.

Procedure

To apply this cumulative fix, follow the readme instructions that apply to your environment. You can find separate instructions for stand-alone, cluster, farm, and remote search environments from the overview topic.

Documentation resource: IBM WebSphere Portal V.8.5.0 combined cumulative fix instructions: stand-alone

Documentation resource: IBM WebSphere Portal V.8.5.0 combined cumulative fix instructions: cluster

Documentation resource: IBM WebSphere Portal V.8.5.0 combined cumulative fix instructions: farm

Documentation resource: IBM WebSphere Portal V.8.5.0 combined cumulative fix instructions: remote search

Configuring features in CF06

Some features are configured by default after you apply your cumulative fix. Other features require additional configuration tasks or enablement steps before you can use them. This section provides documentation resources for features and improvements that require additional configuration.

Updates to custom search box in the portal theme
This feature requires additional configuration to redirect search requests issued by a custom search form to the Search Center.

Documentation resource: "Redirecting search requests from a custom search form to the Search Center" on page 645

Using remote session invalidation with WSRP
This feature requires additional configuration to enable remote session invalidation.

Documentation resource: "Configuring remote session invalidation" on page 1493

Configuring features from previous cumulative fixes

After you install the current combined cumulative fix, you might decide that you want to use a feature from a previous fix. This section provides documentation resources for those features and improvements that require additional configuration.

CF05

Features from CF05

The following features, included with the fifth combined cumulative fix, requires additional configuration:

SmartCloud for Social Business Search (SAML)
This feature requires configuration to integrate WebSphere Portal Express and IBM Connections in SmartCloud for Social Business. As of CF05, you can integrate by using either Active Directory Federation Services (ADFS) or Tivoli Federated Identity Manager (TFIM).

Documentation resource: "Establishing single sign-on (SSO) between the portal installation and IBM Connections in SmartCloud for Social Business" on page 763.

Searching in a multilingual environment
This feature requires configuration. By default, the Search Center uses the preferred language of the user to analyze search terms and refine search results. To enable searching in other languages, configure the Search Center portlet.

Documentation resource: "Configuring search for multilingual sites" on page 684.

Content as a Service pages
This feature requires configuration. Content as a Service pages require the new functions of WebSphere Portal Express and IBM Web Content Manager version 8.5 CF05 to be enabled. Run the configuration task `install-caas` to enable the Content as a Service pages.

Documentation resource: “Setting up Content as a Service” on page 1884.

CF04
Features from CF04

The following feature, included with the fourth combined cumulative fix, requires additional configurations:

Renditions
This feature requires configuration. Renditions are disabled by default. You must run a ConfigEngine task to enable them.

Documentation resource: “Enabling renditions” on page 2015

CF03
Features from CF03

The following feature, included with the third combined cumulative fix, requires additional configurations:

Resource aggregation for portlets
This feature requires configuration if your custom theme was created from the Portal 8.5 theme on a system before CF03. Connect to your WebDAV and edit the local copy of your `metadata.properties` file. Set the theme `metadata resourceaggregation.autoLoadPortletCapabilities` to `true`.

Documentation resource: “Change the auto-loading of portlet capabilities” on page 2538

Blogs and wikis
This feature requires enablement. You must run a configuration task to update the content that is used by Blogs and Wikis to apply the newest updates.

If you customized your Blogs and Wikis, you lose your customizations and you must reapply those customizations after you run the task. Enablement is manual and optional.

Documentation resource: “Blogs and wikis” on page 887

SmartCloud for Social Business HTTP Outbound
This feature requires configuration to use social rendering with a Connections server that runs in the Smart Cloud for Social Business.

To configure the Connections server type, set the custom property **`server.type`** in the WP Connections Integration Service resource environment provider to **`SC4SB`** in the WebSphere Integrated Solutions Console.

Documentation resource: “Configuring the Connections server type” on page 2122.

Invalid friendly URLs
This feature requires enablement and configuration. Set the **`friendly.pathinfo.validation.enabled`** property to `true` in the WebSphere Portal Express Configuration Service Resource Environment Provider.

Documentation resource: “How to enable the validation of friendly URLs for web content” on page 2046.

Documentation resource: “Configuring the validation of friendly URLs for web content” on page 2046.

Roadmaps for migration

Choose the appropriate migration roadmap for your environment.

"Roadmap: Migrating a stand-alone server environment"
Roadmaps provide a high-level overview of complex tasks such as migrating a stand-alone server environment to a new version of IBM WebSphere Portal Express.

"Roadmap: Migrating a clustered environment" on page 94
Roadmaps provide a high-level overview of complex tasks such as migrating a clustered environment to a new version of IBM WebSphere Portal Express.

Roadmap: Migrating a stand-alone server environment

Roadmaps provide a high-level overview of complex tasks such as migrating a stand-alone server environment to a new version of IBM WebSphere Portal Express.

Who should use this roadmap?

Use this roadmap if you are:

- Migrating a stand-alone server environment from Version 7.0.0.x to Version 8.5.
- Migrating a stand-alone server environment from Version 8.0.0.x to Version 8.5.

Note: If you are migrating from IBM WebSphere Portal Express Version 8.0.0.1 on WebSphere Application Server Version 8.5.5.2 to WebSphere Portal Express Version 8.5, you must follow a different migration process. For more information, see "Migrating from Portal 8.0.0.1 on WebSphere Application Server 8.5.5.2" on page 800.

You must apply the latest cumulative fix and one of the two most recent fix packs to your source environment, and the latest cumulative fix and the most recent fix pack to your target environment before you can migrate to WebSphere Portal Version 8.5. For more information, see Supported migration paths.

Planning for migration

Gather information and create a plan for migration.

Check the requirements and considerations for migration:

- Documentation resources: Planning for migration
  - Backup and recovery of data files and databases is an essential operation for any business system, particularly for data and applications that are running in production environments. Create and follow a plan for backing up and recovering data on all tiers of your WebSphere Portal Express deployment. For more information, see Backup and recovery.
  - Remove unsupported or deprecated features before you start migration. For more information, see Deprecated features.
  - If you need to migrate multiple environments, such as a production environment or development environment, you can use staging to production techniques. For more information, see Migrate multiple tier environments.

Plan a local migration if your source and target environments are on the same system, and plan for a remote migration if your source and target environments

are on separate systems. For more information about local and remote migrations, see Local versus remote.

Preparing your source environment

Prepare the source portal that you want to use for migration.

- Documentation resources: Preparing your source environment
  - To keep the earlier portal environment in production and reduce the amount of downtime during migration copy the earlier portal server JCR and Release domains. For more information, see "Using copies of source database domains to minimize downtime" on page 835.

Setting up your target environment

Set up your target environment for migrating to Version 8.5.

- Documentation resource: Setting up your target environment
  - The portal migration attempts to install custom applications in the target environment, but it does not automatically copy the files that are required for those applications. If the files are not copied over, it is possible that the applications will fail to install or not work properly. For more information about copying these files, see "Copying files for third party and custom applications" on page 835.
  - To effectively set up your target environment, install the Portal and WebSphere binary files on all target systems. For more information, see "Installing Portal and WebSphere binary files" on page 833.

Migrating to WebSphere Portal Express Version 8.5

Start the Configuration Wizard to migrate data, applications, databases, property files, security settings, and more.

1. To get the latest updates for the wizard, apply the most recent cumulative fix. For more information about applying the latest fix pack, see WebSphere Portal and Web Content Manager V8.5.0.0 combined cumulative fix overview.

   Note: Skip this step, if you have the most recent fix pack applied.
2. Access the Configuration Wizard using your target environment and system host name. Go to: `http://your_server:10200/ibm/wizard`.
3. Log in to the Configuration Wizard with the administrative ID for the configuration wizard profile, cw_profile.
4. Select **Migrate to a New Version** > **Migrate a Stand-alone Server**.
5. Provide information about your environment.
6. Save your wizard selections.
7. Choose one of the following options:
   - Click **Download Configuration Scripts** to run the steps remotely.
   - Click **Start Configuration** to run the steps locally. This option starts to run the automated steps until a manual step is encountered.

Migration is partially complete after you do the tasks in the configuration wizard. You must return to the product documentation to complete the final steps of the migration process.

Next steps

Migration is not complete until you review the Next steps section in the product documentation. Complete the post-migration activities and enabling new functionality tasks that are applicable to your environment. Do not complete any of the enabling new functionality tasks until all post-migration activities tasks are finished.

- Documentation resources: Next steps
  - Then, you must run the **PRE-APPLY-FIX** and **APPLY-FIX** tasks to ensure that your migrated system is up-to-date. This step is required. For more information, see "Applying the latest combined cumulative fix updates" on page 857.
  - More tasks must be completed depending on how you customized the source portal environment and which components you used. For example, if you use a virtual portal, then complete the virtual portal post-migration activities. For more information, see Post-migration activities.
  - New functionality that was not available in the earlier portal version requires extra attention after migration is complete. For more information, see Enabling new functionality.

After you complete the tasks in the Next steps section of the product documentation, migration is complete.

Roadmap: Migrating a clustered environment

Roadmaps provide a high-level overview of complex tasks such as migrating a clustered environment to a new version of IBM WebSphere Portal Express.

Who should use this roadmap?

Use this roadmap if you are:

- Migrating a clustered environment from Version 7.0.0.x to Version 8.5.
- Migrating a clustered environment from Version 8.0.0.x to Version 8.5.

Note: If you are migrating from IBM WebSphere Portal Express Version 8.0.0.1 on WebSphere Application Server Version 8.5.5.2 to WebSphere Portal Express Version 8.5, you must follow a different migration process. For more information, see "Migrating from Portal 8.0.0.1 on WebSphere Application Server 8.5.5.2" on page 800.

You must apply the latest cumulative fix and one of the two most recent fix packs to your source environment, and the latest cumulative fix and the most recent fix pack to your target environment before you can migrate to WebSphere Portal Version 8.5. For more information, see Supported migration paths.

Planning for migration

Gather information and create a plan for migration.

Check the requirements and considerations for migration:

- Documentation resources: Planning for migration
  - Backup and recovery of data files and databases is an essential operation for any business system, particularly for data and applications that are running in production environments. Create and follow a plan for backing up and

recovering data on all tiers of your WebSphere Portal Express deployment. For more information, see Backup and recovery.
  - Remove unsupported or deprecated features before you start migration. For more information, see Deprecated features.
  - If you need to migrate multiple environments, such as a production environment or development environment, you can use staging to production techniques. For more information, see Migrate multiple tier environments.

Plan a local migration if your source and target environments are on the same system, and plan for a remote migration if your source and target environments are on separate systems. For more information about local and remote migrations, see Local versus remote.

Preparing your source environment

Prepare the source portal that you want to use for migration.

- Documentation resources: Preparing your source environment
  - Review the considerations for a multiple cluster environment for information on supporting multiple clusters that use different database credentials. For more information, see "Multiple cluster environments" on page 799.
  - To keep the earlier portal environment in production and reduce the amount of downtime during migration copy the earlier portal server JCR and Release domains. For more information, see "Using copies of source database domains to minimize downtime" on page 835.
  - The target environment initially uses the same ports as the source environment. There are three important steps you must complete to ensure that the source and target environments do not become corrupted. For more information, see "Disabling automatic synchronization to protect your clustered source environment" on page 817.

Setting up your target environment

Set up your target environment for migrating to Version 8.5.

- Documentation resource: Setting up your target environment
  - The portal migration attempts to install custom applications in the target environment, but it does not automatically copy the files that are required for those applications. If the files are not copied over, it is possible that the applications will fail to install or not work properly. For more information about copying these files, see "Copying files for third party and custom applications" on page 835.
  - To effectively set up your target environment, install the Portal and WebSphere binary files on all target systems. For more information, see "Installing Portal and WebSphere binary files" on page 833.

Migrating to WebSphere Portal Express Version 8.5

Start the Configuration Wizard to migrate data, applications, databases, property files, security settings, and more.

1. During a cluster migration, you might need to enter information into the Configuration Wizard more than once. Use the following worksheet to identify the information that is entered multiple times, and use these values during the migration process.

Table 5. Worksheet: Migration information for the Configuration Wizard

Field description:	Value:
Deployment manager host name	
Deployment manager cell name	
Deployment manager node name	
Administrator user name	
Administrator password	
Soap port	

2. To get the latest updates for the wizard, apply the most recent cumulative fix. For more information about applying the latest fix pack, see WebSphere Portal and Web Content Manager V8.5.0.0 combined cumulative fix overview.

   Note: Skip this step, if you have the most recent fix pack applied.
3. Access the Configuration Wizard using your target environment and system host name. Go to: `http://your_server:10200/ibm/wizard`.
4. Log in to the Configuration Wizard with the administrative ID for the configuration wizard profile, cw_profile.
5. Select **Migrate to a New Version** > **Migrate a Cluster Step 1: Migrate the Deployment Manager Profile**.
6. Provide information about your environment.
7. Save your wizard settings.
8. Click **Download Configuration Scripts** to run the steps on the deployment manager.
9. After you complete the steps from Step 1, select **Migrate to a New Version** > **Migrate a Cluster Step 2: Migrate Node Profiles**.
10. Provide information about your environment.
11. Save your wizard settings.
12. Choose one of the following options:
    - Click **Download Configuration Scripts** to run the steps remotely.
    - Click **Start Configuration** to run the steps locally. This option starts to run the automated steps until a manual step is encountered.
13. Complete these steps on all nodes.
14. After you complete the steps for Step 2, select **Migrate to a New Version** > **Migrate a Cluster Step 3: Upgrade Node Profiles**.
15. Provide information about your environment.
16. Save your wizard settings.
17. Choose one of the following options:
    - Click **Download Configuration Scripts** to run the steps remotely.
    - Click **Start Configuration** to run the steps locally. This option starts to run the automated steps until a manual step is encountered.
18. Complete these steps on all nodes

Next steps

Migration is not complete until you review the Next steps section in the product documentation. Complete the post-migration activities and enabling new

functionality tasks that are applicable to your environment. Do not complete any of the enabling new functionality tasks until all post-migration activities tasks are finished.

- Documentation resources: Next steps
  - Then, you must run the **PRE-APPLY-FIX** and **APPLY-FIX** tasks to ensure that your migrated system is up-to-date. This step is required. For more information, see "Applying the latest combined cumulative fix updates" on page 857.
  - More tasks must be completed depending on how you customized the source portal environment and which components you used. For example, if you use a virtual portal, then complete the virtual portal post-migration activities. For more information, see Post-migration activities.
  - New functionality that was not available in the earlier portal version requires extra attention after migration is complete. For more information, see Enabling new functionality.

After you complete the tasks in the Next steps section of the product documentation, migration is complete.

Roadmaps for integration

Choose the appropriate integration roadmaps for your environment.

"Roadmap: Integrating with IBM Connections"

Roadmap: Integrating with IBM Connections

Integrating your portal site with IBM Connections is a multiple step process. Some steps are required and others are optional. Optional steps depend on the level of integration that you need. This roadmap is intended to provide an overview of the process. Links to detailed instructions are provided from the roadmap.

Installing Connections and WebSphere Portal

Before you begin, the following programs must be installed and configured.

Install Connections

Documentation resource: Installing IBM Connections

Install IBM WebSphere Portal Express.

Documentation resource: Installing WebSphere Portal

Configure portal to use a federated LDAP server.

Documentation resource depends on your WebSphere Portal Express installation scenario, stand-alone server, or cluster.

Note: Do not remove the file system user repository after you federate your LDAP server.

Configuring your portal to work with IBM Connections

Procedure

1. Import the SSL certificate from Connections to your portal server.

   Documentation resource: Importing a certificate to support SSL
2. Configure the portal Ajax proxy. The Ajax proxy is updated for the base Connections URLS during the IBM Connections Portlets installation. If FileNet

is used, you must still configure the Ajax proxy manually to update for FileNet. Configure the Ajax proxy so that direct requests that the CCM portlet makes to the FileNet server are allowed to pass through the proxy Server.

Documentation resource: Configuring the Ajax proxy to allow FileNet requests

3. Configure single sign-on.

   Documentation resource: "Set up single sign-on" on page 748

   Documentation resource for portlets: Configuring authentication for the portlets

4. Install and Deploy the Portlets

   Note: Before you install the portlets, steps 1 - 3 must be completed.

   a. Download the Connections portlets for WebSphere Portal Express.

      IBM Connections Portlets for WebSphere Portal

   b. Install the portlets.

      Documentation resource: Installing the IBM Connections Portlets for WebSphere Portal

   c. Add modules that are needed by the Connections portlets to the portal theme default profile.

      Documentation resource: Updating the portlet theme

5. Configure a common directory service for the portlets.

   Documentation resource: Configuring common Directory Services for your security configuration

6. Integrate community membership to allow access control administration that is based on community membership.

   a. Documentation resource: Integrating community membership with Portal security

   b. Documentation resource: Automatically grant page access to community members

7. Enhance the deployment of Connections portlets for WebSphere Portal Express with optional configuration steps.

   Documentation resource:Optional and Recommended Configurations

   Documentation resource: Social rendering

   Documentation resource: Social Media Publisher

   Documentation resource: IBM Social Rendering Templates for Digital Data Connector

8. Configure search with the Remote Content Server Search Service Type (RCSS), for a portal cluster setup that is integrated with Connections.

   Documentation resource:Configuring search using Remote Content Server Search Service Type (RCSS)

Roadmaps for web content

Some roadmaps to help you create and integrate web content into your site.

"Roadmap: Importing web content" on page 99
When you set up a new site, you might have content in an existing system that you need to migrate or import into your site.

Roadmap: Importing web content

When you set up a new site, you might have content in an existing system that you need to migrate or import into your site.

Using REST to import web content into Web Content Manager

Application developers can use Representational State Transfer (REST) services to work with Web Content Manager. The REST service for Web Content Manager provides authoring access to content items and elements. The service follows the Atom Publication Protocol, and atom feeds, and entries are accessible in XML (application/atom+xml) and JSON (application/json) format.

This is useful when you need to perform a once-off migration of Web content from an external system into an Web Content Manager authoring environment.

See "REST service for Web Content Manager" on page 3306 for further information.

Using WCI to import web content managed in an external system

The Web Content Integrator is a solution for integrating externally managed Web content with WebSphere Portal Express. Through the use of standard content syndication feed technologies based on RSS 2.0, the Web Content Integrator provides a loosely-coupled mechanism for transferring published content and metadata to the portal after they have been approved in the source system. When the content and metadata have been transferred to the portal, it is possible to use the built-in content management features of WebSphere Portal Express to secure, personalize, and display the content to users.

This is useful when you need to continue to use externally managed Web content in a site delivered using Web Content Manager.

See "IBM Web Content Integrator" on page 1936 for further information.

Chapter 4. Installing

IBM WebSphere Portal Express provides flexible deployment options that range from proof-of-concept where you can examine and test functionality to a highly available and scalable production environment. Review the planning information to learn more about hardware and software requirements, high availability, scalability, supported topologies, and much more. Select your operating system and then select the installation pattern that most reflects your business needs.

1. "Planning to install WebSphere Portal Express"
   Before you install IBM WebSphere Portal Express in a production environment, you need to assess your hardware and software needs, possible database configurations, security options, and LDAP server options. Skipping this important step can lead to unexpected results and costly delays.
2. "Getting the software" on page 135
   There are different ways for you to get WebSphere Portal Express and Web Content Manager Version 8.5 software.
3. "Installing and preparing the prerequisite software" on page 148
   Before you install the digital experience software, make sure that the prerequisite software is installed and configured. Depending on your environment, you might already have the prerequisites, such as a database server and user registry. Verify that the prerequisite software is the correct version, has the required fix packs applied, and is configured to work with the digital experience software.
4. "Installing the digital experience software" on page 167
   IBM's Exceptional Digital Experience is designed to help create, manage, simplify, and integrate your processes into an engaging online experience. IBM WebSphere Portal and IBM Web Content Manager are a part of the Exceptional Digital Experience. The product documentation uses digital experience software as a shorthand for IBM WebSphere Portal and IBM Web Content Manager.
5. "Installing add-ons" on page 210
   You can use the Solution Installer through the Configuration Wizard to install and uninstall add-ons to an IBM WebSphere Portal Express server instance. The Solution Installer uses the Portal Application Archive (PAA) format as the standard format for application distribution. Portal Application Archive (PAA) updates are not supported in the configuration wizard currently. For more information about updating add-ons using a command prompt, see the Managing your existing PAA file section.
6. "Uninstalling the digital experience software" on page 218
   Uninstalling the digital experience software is a multiple step process. Manual uninstallation instructions are provided for a single-server configuration in case of an error situation.

Planning to install WebSphere Portal Express

Before you install IBM WebSphere Portal Express in a production environment, you need to assess your hardware and software needs, possible database configurations, security options, and LDAP server options. Skipping this important step can lead to unexpected results and costly delays.

About this task

Restriction: The serverName is hardcoded to WebSphere Portal Express. The serverName cannot be changed in a stand-alone environment. If you do change it, the ConfigEngine scripts do not work. For a clustered environment, see Technote 1370392 for options on replacing the WebSphere Portal Express JVM.

"System requirements" on page 103
Before you install IBM WebSphere Portal Express, review the hardware and software requirements to ensure that you have the supported versions of prerequisite and corequisite software and the required hardware.

"Release notes" on page 103
Known issues and problems are centrally available on the support page. Links into the support knowledge base are integrated throughout the documentation to make sure that you have the most current information. Before you start the installation process, check the IBM Support site for the most current information about known limitations or issues. Use the following dynamic queries to find late breaking information about this release.

"WebSphere Portal Express Support Statement" on page 103
This support statement proposes a revision to the definition of "supported" and "unsupported" about the various products of which IBM WebSphere Portal Express depends on for proper operation.

"User IDs and passwords" on page 107
Understanding character limitations for user IDs and passwords is important because they are used throughout the system to provide access and secure content. The character limitations provided here apply to the IBM WebSphere Portal Express administrator, IBM WebSphere Application Server administrator, database administrator, LDAP server administrator, and user IDs. Database and LDAP servers can have more restrictive limitations than provided here. Therefore, check the database and LDAP server product documentation for restrictions. Failure to correctly define user IDs and passwords during the installation process can result in installation failure. In addition, your company might have more restrictive user ID and password requirements that you must also follow.

"Web Content Manager environments" on page 109
To use a Web Content Manager system, you need to deploy a set of Web Content Manager environments within your overall WebSphere Portal system. Reviewing the Web Content Manager environments help you understand what happens in each environment and how you might want to set up your physical severs. Web Content Manager is installed by using the WebSphere Portal installation user interface.

"Evaluation license" on page 119
IBM WebSphere Portal Express comes with a special license that you can use to install and deploy the product for a 60-day evaluation at no charge. If you then purchase WebSphere Portal Express, with a simple utility you can convert the evaluation license to a full production license. Users who prefer to purchase and install WebSphere Portal Express directly, without the evaluation license, can do so without having to install or convert the evaluation license.

"Database" on page 119
IBM WebSphere Portal Express includes an Apache Derby database that is configured and ready for immediate use. As a result, you have a running portal that is ready for exploration or portlet and theme development. But for a production environment or any environment for Web Content Manager, you must use one of the other supported database management systems.

"User registry considerations" on page 128
A user registry or repository authenticates a user and retrieves information about users and groups to do security-related functions, including authentication and authorization.

"WebSphere Portal Express high availability" on page 135
IBM WebSphere Portal Express is licensed for use in a single-server configuration and might not be used in either a cloned configuration or a clustered configuration except when implementing idle standby for the purpose of failover.

System requirements

Before you install IBM WebSphere Portal Express, review the hardware and software requirements to ensure that you have the supported versions of prerequisite and corequisite software and the required hardware.

See the detailed system requirements document at the following URL: WebSphere Portal detailed system requirements

Release notes

Known issues and problems are centrally available on the support page. Links into the support knowledge base are integrated throughout the documentation to make sure that you have the most current information. Before you start the installation process, check the IBM Support site for the most current information about known limitations or issues. Use the following dynamic queries to find late breaking information about this release.

Related information:

Technotes for installation and configuration issues

Technotes for migration issues

Technotes for database connectivity issues

All technotes for this release

WebSphere Portal Express Support Statement

This support statement proposes a revision to the definition of "supported" and "unsupported" about the various products of which IBM WebSphere Portal Express depends on for proper operation.

Introduction

WebSphere Portal Express requires the use of several collateral products for its normal operations. In particular, it requires WebSphere Application Server, a database, a repository for user information (typically an LDAP), and other products depending on specific customer requirements.

During the testing of a new release, Development generally tests WebSphere Portal Express with a prescribed list of these collateral products. These products are designated as "Supported Products" in the documented hardware and software requirements for that release.

Because the list of "Supported Products" cannot reasonably describe all possible configurations that a customer might need to use, some customers voiced concerns

about the level of support that is provided for configurations that are not designated as “Supported”. This document is intended to provide clarification of the level of support that can be expected for the current release with various combinations of dependent products.

Note: Although the statements in this document reflect the general level of support that can be expected for WebSphere Portal Express, the terms and conditions of any specific support offering, license or other Agreement you might have with IBM will determine the actual delivered support for the product. Nothing herein shall be construed as supplementing, modifying or superseding the terms of your IBM license agreement for WebSphere Portal Express or any other agreement you might have with IBM, nor shall it create any obligation for IBM to deliver a level of support other than might be set forth in such Agreements.

Categories of Support

There are three categories of support for collateral products to WebSphere Portal Express. They are “Supported Products”, “Newer Versions, and Releases of Supported Products” and “Unsupported Products”. The definition and support statement for each category follows:

Supported Product

A “Supported Product” is a product (at a specified version, release and fix level) that was tested by Development and is known to work with WebSphere Portal Express.

Products in this category are supported according to the terms of your WebSphere Portal Express License Agreement. PMRs (Problem Management Records) are accepted by IBM Support in accordance with the conditions of the WebSphere Portal Express License Agreement.

Newer Versions and Releases of Supported Products

Many products outside the specific version(s), release(s), or fix pack(s) of the “Supported” version (referenced in the documented hardware and software requirements) might not be explicitly tested by IBM WebSphere Development, yet can reasonably be expected to perform within the accepted bounds of reliability, function, and performance by a customer.

Products that fall into this category are typically newer releases or fix levels of a product already in the “Supported Product” category or a product that adheres to a standard API that WebSphere Portal Express supports (such as an LDAP server). Some specific examples might include a newer operating system fix level, a WebSphere Application Server (WAS) fix pack newer than the original “Supported” fix pack level, an IBM Java (JVM) fix pack, a new fix pack, or release of DB2 or an updated LDAP server.

For products that fall into this category, support is as follows:

For IBM products, such as IBM Directory Server or Domino LDAP, IBM DB2, IBM JDKs (JVMs) and WebSphere Application Server, WebSphere Portal Express will fully support fix-pack, release and version updates that do not significantly change interfaces or other underlying support that WebSphere Portal Express depends on for its functionality. If and when a newer release of one of these products is shipped that WebSphere Portal Express cannot accommodate, that fact will be noted as described in the next section entitled “Unsupported Products”. Note that in order for WebSphere Portal Express to support an update to a database or LDAP product, WebSphere Application Server must support that update as well.

For non-IBM products, the Support team makes a commercially reasonable effort to support products in this category. Support accepts problem reports (PMRs) for the appropriate releases that use these untested products. If Support is able to re-create the reported problem with a "Supported" version of the product, we will attempt to fix the problem.

If Support is not able to re-create the problem with a "Supported" version of the product in question and is not able to resolve the problem on the untested version of the product in question, Support will look to the support organization for the product in question to provide resolution. Please note that varying degrees of customer involvement may be necessary to handle this process for non-IBM products.

If the support organization for the untested product in question is unable to resolve the problem, Support will deem that version, release or fix pack level of the untested product in question to now be an "Unsupported Product".

Unsupported Products

An "Unsupported Product" is a product (at a specified version, release and fix level) that is known to not work with WebSphere Portal Express and is therefore not supported. A product can be included in this category as a result of an explicit test effort by Development or as a result of discovery from a prior customer problem. The WebSphere Portal Express Support team maintains a list, by release, of all known "Unsupported Products". The list is published as a techdoc and is available to all customers.

WebSphere Application Server has a similar support statement, which can be found on the web.

Note: WebSphere Application Server uses specially customized builds of the IBM Java SDKs on certain platforms. Updates to these builds must be obtained from WebSphere Application Server support.

WebSphere Portal Express can be sensitive to changes in the underlying WebSphere Application Server. Upgrading to a new fix pack level of the application server is well tolerated and encouraged (such as from WebSphere Application Server version 8.0 to 8.0.x) as long as all required fixes for WebSphere Application Server are available as integrated into that fix pack or by applying an interim fix specifically for that maintenance level. However, upgrading from one version of WebSphere Application Server to the next (such as from 7.0 to 8.0) is problematic if not done within the context of a migration of versions and must never be attempted with an "in-place" system.

For example, an existing instance of WebSphere Portal Express version 8.0 that is installed and functioning on WebSphere Application Server version 8.0 cannot be successfully migrated to WebSphere Application Server version 8.x by using the WebSphere Application Server Migration Tools. Such attempts might result in a non-functional system. For more information on such scenarios, consult IBM WebSphere Portal Express support.

Support for LDAP Servers

LDAP support spans two categories:

Fully tested and supported LDAP servers:

The list of fully tested LDAP servers for each release of WebSphere Portal

Express is documented in the detailed system requirements for each release. WebSphere Portal Express support accepts problem reports for the appropriate WebSphere Portal Express releases using the tested directory servers. These problem reports receive high-priority attention. Features that are tested with these directories include relatively simple search and retrieval functions for user and group objects. Functions outside this scope, such as the Active Directory Global Catalog feature, are considered advanced features and have not been tested with WebSphere Portal Express. WebSphere Portal Express support encourages customers to work with their LDAP provider for additional support on these advanced features.

Untested and partially supported LDAP servers:
In general, WebSphere Portal Express support makes a best effort to support directory servers that have not been tested with WebSphere Portal Express. WebSphere Portal Express support accepts problem reports for the appropriate WebSphere Portal Express releases using untested directory servers. If WebSphere Portal Express support can re-create the reported problem using a tested LDAP server, staff will attempt to fix the problem. If the support team is not able to re-create the problem on a tested LDAP server, customers are referred to the LDAP provider for further assistance.

Support for External Security Managers (ESM)

ESM support spans two (2) categories:

Fully tested and supported ESM software:
The list of fully tested ESM software versions for each release of WebSphere Portal Express is documented in the detailed system requirements for each release. WebSphere Portal Express support accepts problem reports for the appropriate WebSphere Portal Express releases using the tested ESM servers. These problem reports receive high-priority attention. Features that are tested with these software products include authentication and authorization. Functions outside this scope, such as login customizations, referrals, impersonation, and step up authentication are considered advanced features and have not been tested with WebSphere Portal Express. WebSphere Portal Express support encourages customers to work with their ESM provider for additional support on these advanced features.

Untested and partially supported ESM servers:
In general, WebSphere Portal Express support makes a best effort to support ESM servers that have not been tested with WebSphere Portal Express when relying on the ESM for authentication only. WebSphere Portal Express support accepts problem reports for the appropriate WebSphere Portal Express releases using untested ESM Trust Association Interceptor (TAI) implementations. If WebSphere Portal Express support can re-create the reported problem using a tested ESM, staff will attempt to fix the problem. If the support team is not able to re-create the problem on a tested ESM, customers are referred to the ESM provider for further assistance.

Related information:


System requirements

User IDs and passwords

Understanding character limitations for user IDs and passwords is important because they are used throughout the system to provide access and secure content. The character limitations provided here apply to the IBM WebSphere Portal Express administrator, IBM WebSphere Application Server administrator, database administrator, LDAP server administrator, and user IDs. Database and LDAP servers can have more restrictive limitations than provided here. Therefore, check the database and LDAP server product documentation for restrictions. Failure to correctly define user IDs and passwords during the installation process can result in installation failure. In addition, your company might have more restrictive user ID and password requirements that you must also follow.

When a person signs up as a user or when an administrator enrolls a user, they must complete the user information form. On this form, do not enter characters that might not be supported. Regardless of what characters you are able to enter on the user information form, user ID and passwords are limited to the valid characters described here. You can specify other characters in the given name and surname fields. If your company policy is more restrictive, you can provide that information to your users in the enrollment form help or as inline help directly on the form.

Important: WebSphere Portal Express cannot create user IDs or passwords that contain spaces, although it fully supports any existing user IDs and passwords or those IDs created in the user repository that contain spaces.

Under normal circumstances a valid user ID and password can contain the following characters:

Note: The only supported characters in IBM i are lowercase characters, uppercase characters, numbers, and the underscore.

Lowercase characters {a-z}

Uppercase characters {A-Z}

Numbers {0-9}

Exclamation point {!}

Open parenthesis {(}

Close parenthesis {)}

Dash {-}; this character is not supported as the first character in the user ID or password

Period {.}; this character is not supported as the first character in the user ID or password

Underscore {_}; this character is the only supported special character in IBM i

Grave accent {`}

Tilde {~}

Commercial at {@}, this character is not supported when you create the WebSphere Portal Express and WebSphere Application Server administrator during installation.

Important: These characters are all ASCII characters. Non-ASCII characters are not allowed for user name or password.

Note: If you plan on using a non-ASCII-based encoding, ensure your Java virtual machine has the correct generic arguments specific for the non-ASCII-based

encoding. For example, for UTF-8 encoding, add the following two parameters to the Java virtual machine generic arguments for WebSphere Portal: **-Dfile.encoding=UTF-8** and **-Dclient.encoding.override=UTF-8**

Note: (Linux only) Some tasks might require you to enter the fully qualified user ID. If your fully qualified user ID contains a space; for example: `cn=wpsadmin,cn=users,l=SharedLDAP,c=US,ou=Lotus,o=Software Group,dc=ibm,dc=com`, you must place the fully qualified user ID in the properties file or into a parent properties file instead of as a flag on the command line. For example, create a parent properties file called `mysecurity.properties`, enter the fully qualified user ID, and then run the task: `./ConfigEngine.sh` *`task_name`* `-DparentProperties=/opt/mysecurity.properties`.

Note: (Windows only) Some tasks might require you to enter the fully qualified user ID. If your fully qualified user ID contains a space; for example: `cn=wpsadmin,cn=users,l=SharedLDAP,c=US,ou=Lotus,o=Software Group,dc=ibm,dc=com`, you must place quotations around the fully qualified user ID before you run the task; for example, `"cn=wpsadmin,cn=users,l=SharedLDAP,c=US,ou=Lotus,o=Software Group,dc=ibm,dc=com"`.

The following table contains a list of the required fields on the user information form and the supported characters.

Table 6. Valid characters and unsupported characters for user information

User information	Valid characters	Unsupported characters
User ID	**Note:** The only supported characters in IBM i are lowercase characters, uppercase characters, numbers, and the underscore. Lowercase characters {a-z} Uppercase characters {A-Z} Numbers {0-9} Exclamation point {!} Open parenthesis {(} Close parenthesis {)} Dash {-}; this character is not supported as the first character in the user ID or password Period {.}; this character is not supported as the first character in the user ID or password Underscore {_}; this character is the only supported special character in IBM i Grave accent {`} Tilde {~} Commercial at {@}, this character is not supported when you create the WebSphere Portal Express and WebSphere Application Server administrator during installation.	Only ASCII characters are allowed. **Other restrictions:** The user ID cannot contain spaces; for example, *`user name`*. **Note:** User IDs cannot be longer than 200 characters. If you enter any unsupported characters during the installation, you receive an error message that states which character is invalid. For example, "The special character [@] was found in the **administrative user ID** field. Enter the administrative user ID again." **Important:** You receive a different error message if you enter any unsupported characters when you create users through the **Manage users and groups** portlet.

Table 6. Valid characters and unsupported characters for user information (continued)

User information	Valid characters	Unsupported characters
Password / Confirm password	**Note:** The only supported characters in IBM i are lowercase characters, uppercase characters, numbers, and the underscore. Lowercase characters {a-z} Uppercase characters {A-Z} Numbers {0-9} Exclamation point {!} Open parenthesis {(} Close parenthesis {)} Dash {-}; this character is not supported as the first character in the user ID or password Period {.}; this character is not supported as the first character in the user ID or password Underscore {_}; this character is the only supported special character in IBM i Grave accent {`} Tilde {~} Commercial at {@}, this character is not supported when you create the WebSphere Portal Express and WebSphere Application Server administrator during installation.	Diacritics, such as the umlaut, and DBCS characters are not allowed. **Other restrictions:** The password cannot contain spaces; for example, *pass word*. **Note:** Passwords cannot be longer than 128 characters. **Attention:** Login or **ConfigEngine** tasks might fail if the password contains any unsupported characters, including DBCS characters. This action happens even if a user is successfully enrolled with a password that contains DBCS characters. If you enter any unsupported characters during the installation, you receive an error message that states which character is invalid. For example, "The special character [@] was found in the password field. Enter the password again."
Given name	All characters	n/a
Surname	All characters	n/a

Note: The previous characters are true if the **user.UNIQUEID.charset** parameter is set to `ascii`. If set to `unicode`, the standard Java Letter definition is used and all characters that are recognized as letter or digit by Java are allowed by default. See the **Puma Validation Service** section in the "Portal configuration services" link for information about further parameters that can be modified to affect the behavior of Portal's validation of users, groups, and passwords.

Web Content Manager environments

To use a Web Content Manager system, you need to deploy a set of Web Content Manager environments within your overall WebSphere Portal system. Reviewing the Web Content Manager environments help you understand what happens in each environment and how you might want to set up your physical severs. Web Content Manager is installed by using the WebSphere Portal installation user interface.

Each server or cluster in your web content system requires a separate data repository, but they would usually share LDAP. A Web Content Manager system can be deployed in isolation or in parallel with a WebSphere Portal system.

"Web content system overview"
The type of web content system you deploy is determined by the size of your web content system, the type of website, and the number of users that create content, or view your web content.

"Web content authoring environments" on page 111
An authoring environment is used to create and manage web content and is used by your content authors and website designers.

"Web content testing environments" on page 114
Testing environments can be simple or complex. A simple example is a stand-alone server where content and non-content items are tested before they are sent to the live site. A complex environment is a complete replica of your delivery environment. A more complex environment is used to test content, theme, and application changes and the performance of your delivery environment.

"Web content delivery environments" on page 116
A production delivery environment is the environment that hosts your website. If you have a IBM Web Content Manager only license, you can deliver your website without using WebSphere Portal features.

Web content system overview

The type of web content system you deploy is determined by the size of your web content system, the type of website, and the number of users that create content, or view your web content.

Web content system types

There are three main types of web content systems:

Single environment systems
This environment is where authoring and delivery occur within a single environment. This type of environment would be deployed by a small organization with a small website, such as an intranet. Authoring and delivering content within the same environment can be resource-intensive, so the type of environment you deploy needs to be robust enough to allow authoring and delivery to occur at the same time. For example, running clustered servers is a common solution for a single instance system.

Dual environment systems
This environment is where authoring and delivery are split into different environments. This model reduces the load on both authoring and delivery servers and also allows the authoring environment to be located behind a firewall. This type of system would be used with externally facing websites, or where you have many users authoring content or many users viewing a website.

Staged systems
This environment is where a staging environment is added between the authoring and delivery environments. The staging environment can be used for user acceptance testing (UAT) or to accumulate changes from your authoring environment before changes are syndicated to your delivery environment in a single batch. This system would be deployed to deliver large, complex sites with many content creators and you need to ensure that the website content is accurate, error-free and can run under load.

Environment types

Authoring environment

An authoring environment is used to create and manage web content. This environment is used by your content creators and website designers. An authoring system can consist of:

- An authoring server or cluster.
- Individual UAT servers where site and content updates can be tested before the content is syndicated to the delivery environment.

Staging environment

A staging environment can consist of:

- Individual holding servers where changes from your authoring environment can be accumulated before your changes are syndicated to your delivery environment in a single batch. Pairs of holding servers can be used to provide you with built-in redundancy.
- A complete replica of your delivery environment where UAT can occur to both review site and content updates, and to test the performance of your delivery environment.

Delivery environment

This environment is used by your website viewers. A delivery environment can consist of:

- Pre-rendered sites where a web content site is pre-rendered as a set of HTML files that are then used to deliver a static website.
- a WebSphere Portal server or cluster where content is delivered by a servlet. Servlet delivery is used to deliver websites that contain dynamic content, but do not include any WebSphere Portal content or applications.
- a WebSphere Portal server or cluster where content is delivered by either a local or remote web content viewer portlet. Web content viewer portlets are used to deliver websites that contain dynamic web content alongside other portlets or applications.
- A combination of the previous three.

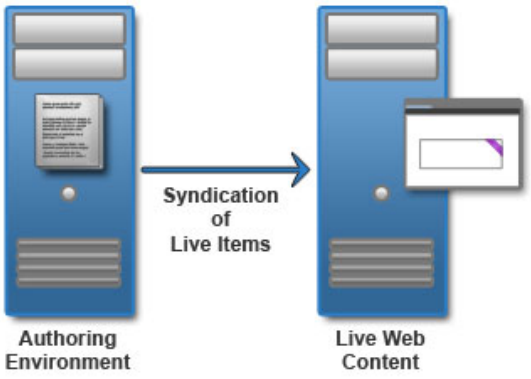
Web content authoring environments

An authoring environment is used to create and manage web content and is used by your content authors and website designers.

Most Web Content Manager sites need to support many content authors. A clustered server solution is the best solution for this scenario.

Standard authoring environment

A standard authoring environment consists of a single authoring cluster that syndicates directly to either a staging or delivery environment. The following topology depicts an authoring environment that is configured to syndicate live items to the live production environment. The production server hosts the live site that visitors browse. Live items include published and expired items. The authoring environment is depicted as a stand-alone server, however it might be clustered.



The following activities occur in the authoring environment:

- Create drafts
- Approve drafts
- Test changes
- Publish changes

Authoring environment with testing

Add a test environment to run user acceptance testing on your content management system and website. The test environment provides an extra layer of validation before content and design changes are pushed to the live website.

In the following topology, all items are syndicated from the authoring environment to the testing environment. After the content is tested, only live items are syndicated back to the authoring environment. Then, live items are syndicated to the production server.

In the diagram, the following activities happen on each server.

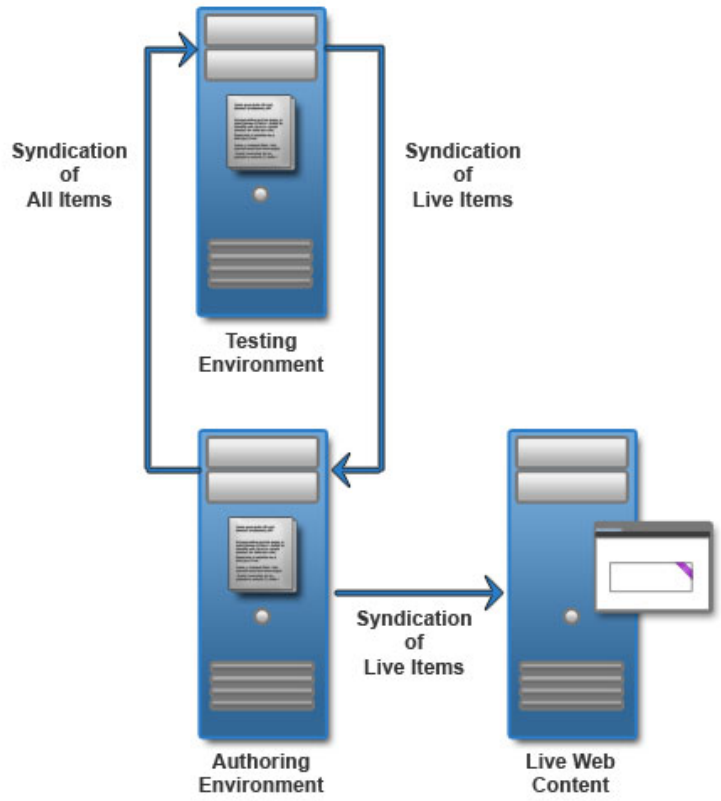
Authoring environment

- Create drafts
- Approve drafts
- Test changes
- Syndicate all items to test environment
- Syndicate live items to live site

Testing environment

- Publish changes
- Test changes
- Syndicate live items to authoring environment

In addition, theme changes might be pushed to testing environment to validate that the content elements and design elements integrate as expected. Design integration elements are not included in the topology diagram.



Decentralized authoring environments

If your content authors are at different locations, or you have different groups of content authors, consider deploying a set of decentralized authoring clusters. This scenario works best if the decentralized content authors work with separate content stored in different content libraries.

For example, if you have users that are in different locations, it might be more efficient to set up a local authoring environment at each location. Two-way syndication is used between all authoring environments with a centralized authoring environment. The centralized authoring environment provides an integrated view of all changes from the different authoring environments.

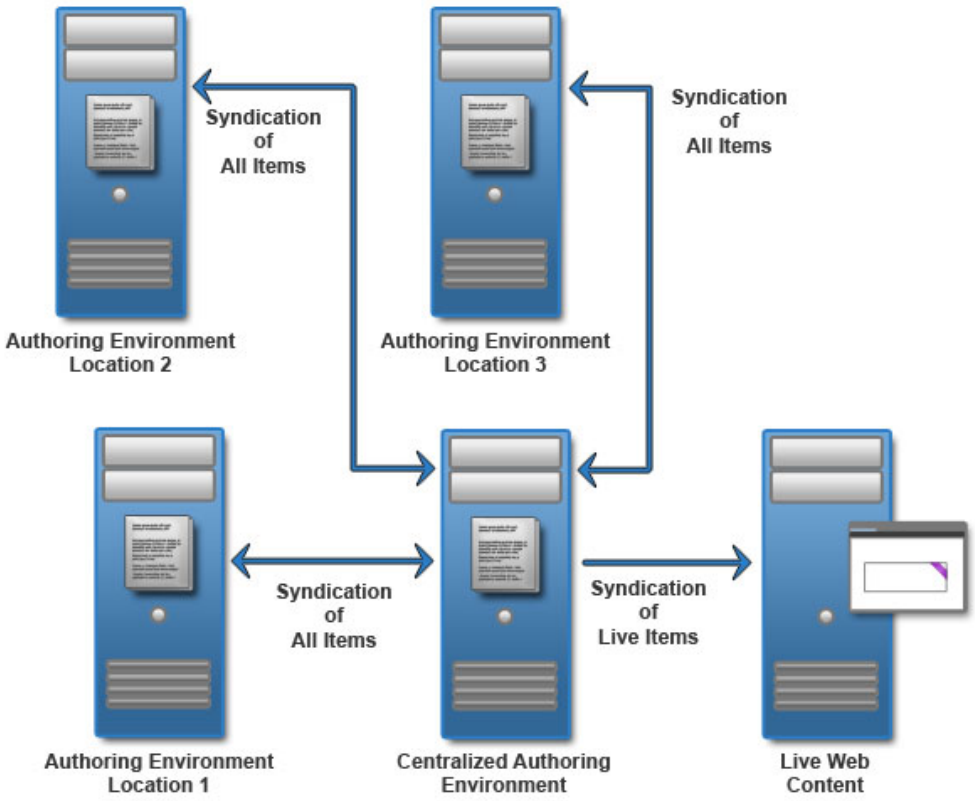
In the diagram, the following activities happen on each server.

Authoring environments 1, 2, and 3

- Create drafts
- Approve drafts
- Test changes
- Syndicate all items to central authoring environment

Centralized authoring environment

- Receive all items from authoring environments
- Syndicate all items to authoring environment
- Syndicate live items to testing environment (not shown)
- Syndicate live items to the live website



Decentralized authoring creates the risk of conflicting updates between authoring environments. To reduce the risk of conflicts, you can allocate different sites, or different sections of a site, to each authoring environment. You can also use different authoring environments for different user roles. For example, content authors might use a different authoring environment than presentation template designers.

Access to each decentralized authoring environment is controlled with a combination of authoring portlet access controls and item security settings. For example, only users that require access to the local authoring environment would be granted access to the local authoring portlet. Users would be given "Read" access to all items, but only "Edit" access to items they are required to update.

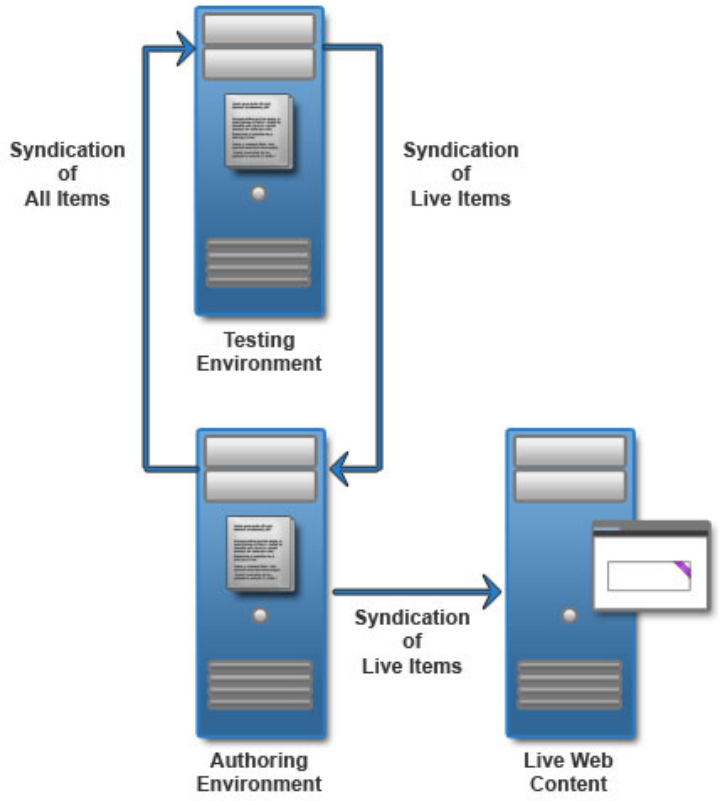
Web content testing environments

Testing environments can be simple or complex. A simple example is a stand-alone server where content and non-content items are tested before they are sent to the live site. A complex environment is a complete replica of your delivery environment. A more complex environment is used to test content, theme, and application changes and the performance of your delivery environment.

A single website might have multiple testing environments. Some testing environments focus on content and design integration, application and data integration, performance, and more. A web content test environment is a layer of validation before changes are sent to the live website. The test environment might be used to accumulate changes from your authoring environments before the changes are syndicated to the production delivery environment.

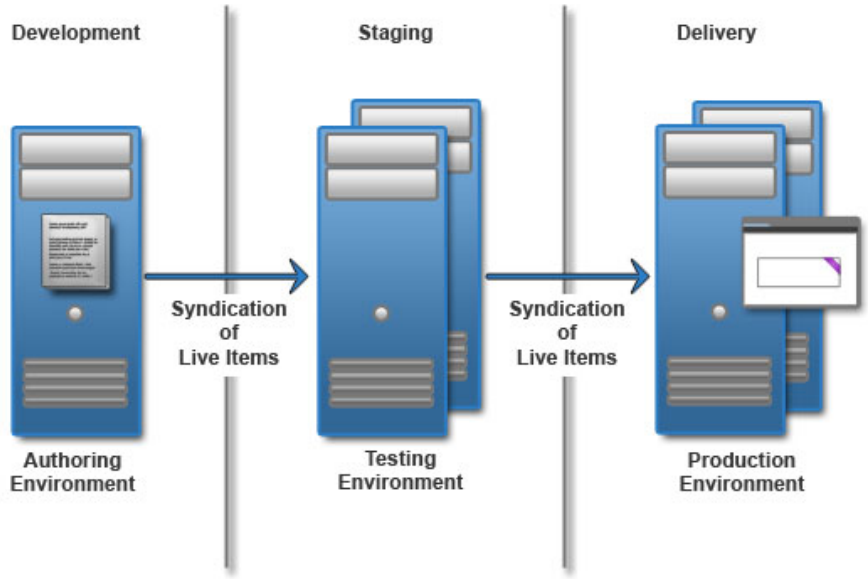
Site testing within an authoring environment

When testing within an authoring environment a testing server is paired with an authoring server. The testing server simulates the delivery environment and is used to test major changes to a website.



System testing within a staging environment

When testing within a staging environment, data from the authoring environment is syndicated to a staging environment. User acceptance testing happens in the staging environment. If all tests are passed, data is syndicated from staging to the delivery production environment.



Specific activities take place in each environment.

Authoring environment

- Create drafts
- Approve drafts
- Test changes
- Publish changes
- Syndicate live items to the staging environment

Testing environment

Test changes.

Syndicate live items to production environment.

Production environment

Deliver live website.

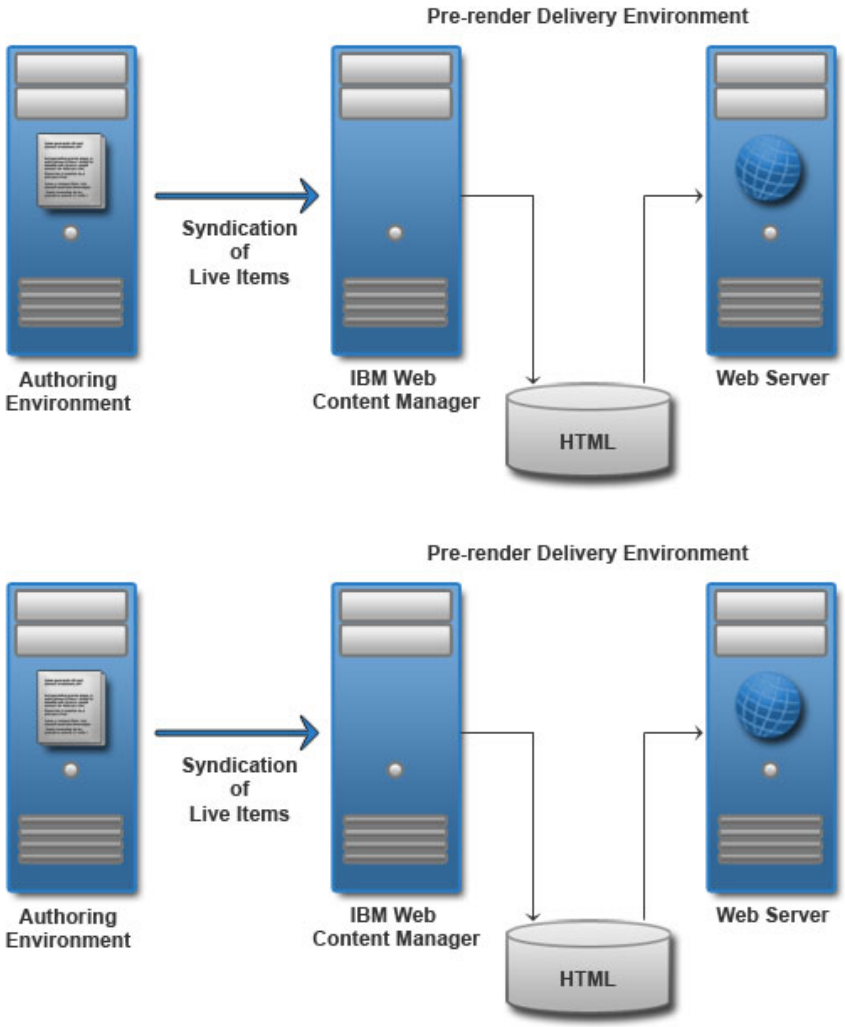
Web content delivery environments

A production delivery environment is the environment that hosts your website. If you have a IBM Web Content Manager only license, you can deliver your website without using WebSphere Portal features.

Pre-rendered delivery

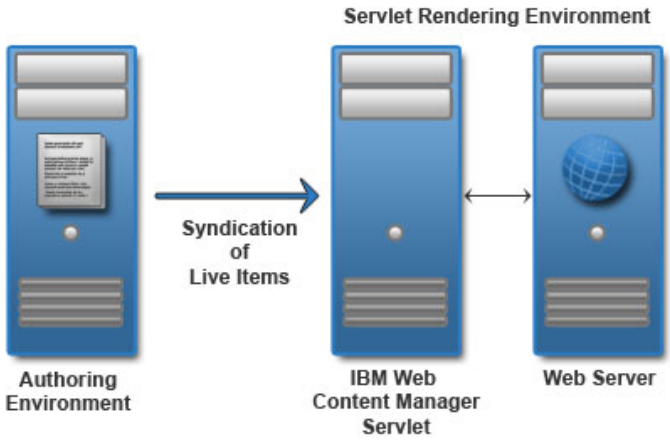
Deploy a pre-rendered site when you are not using any WebSphere Portal features, such as portlets, and your content is static and is only updated periodically. When you set up a pre-render delivery environment, your complete website is converted to static HTML files.

In a pre-rendered delivery environment, live items are syndicated from the authoring environment to the delivery environment. The content is converted into a set of static HTML files, which are then displayed to users through a web server.



Servlet delivery

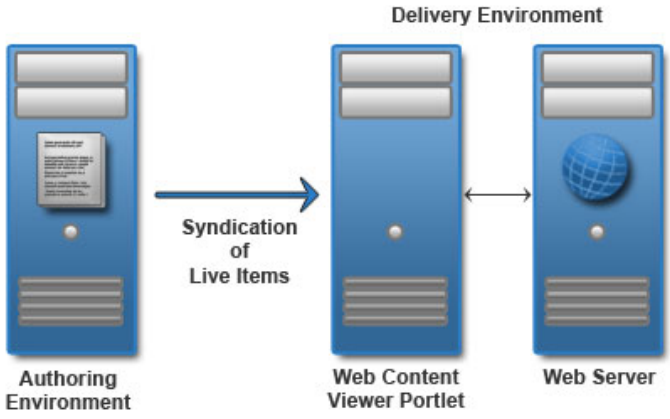
A servlet delivered website is used when you do not need to use any portlet-based features such as authoring tools. In a servlet delivery environment, content is syndicated from the authoring server to the delivery server. The Web Content Manager servlet displays the content. Site visitors access the site through a web server.



Portlet delivery

Web content viewers are portlets that display content from a web content library as part of a portal page. If your presentation is simple, a single web content viewer can be sufficient. However, you can also use multiple web content viewers to aggregate content from different libraries and provide a richer experience for site visitors. A local web content view portlet is used to display content within your web content delivery environment.

In a portlet delivery environment, content is syndicated from the authoring server to the delivery server. It is displayed to users through a web content viewer portlet that is deployed on a portal server. When a local web content viewer is used, the web content viewer portlet is deployed on the same server as Web Content Manager.

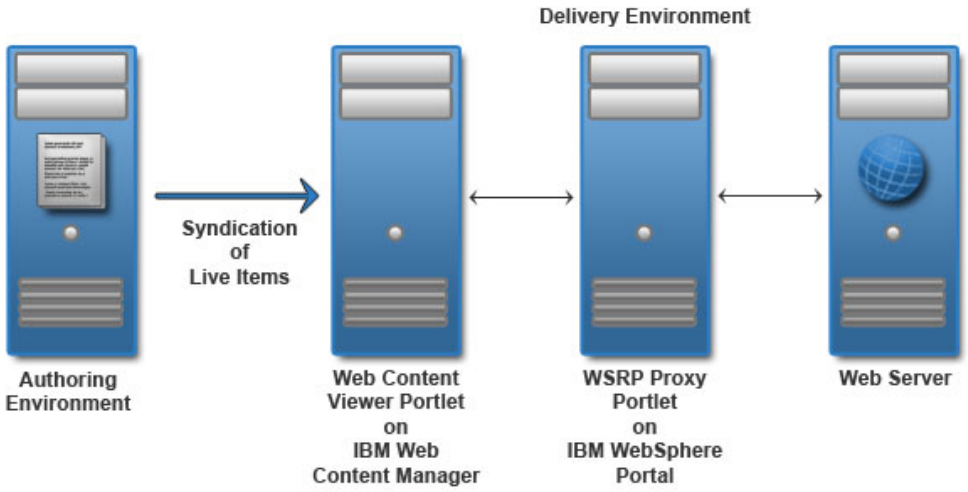


Remote portlet delivery

WSRP support in the web content viewer is used to display content on a remote WebSphere Portal server or cluster.

In a remote portlet delivery environment, content is syndicated from the authoring server to a Web Content Manager server in the delivery environment. The web content viewer portlet is deployed on the Web Content Manager server. It is configured to communicate with the WSRP proxy portlet that is installed on a portal server in the delivery environment. Users view web content by accessing the

proxy portlet on the remote portal server, typically through a web server.



Evaluation license

IBM WebSphere Portal Express comes with a special license that you can use to install and deploy the product for a 60-day evaluation at no charge. If you then purchase WebSphere Portal Express, with a simple utility you can convert the evaluation license to a full production license. Users who prefer to purchase and install WebSphere Portal Express directly, without the evaluation license, can do so without having to install or convert the evaluation license.

Note: This version is intended for new users who did not previously install WebSphere Portal Express. You cannot use the WebSphere Portal Express media or downloaded installation image to upgrade an existing installation.

Using the evaluation license

When you install WebSphere Portal Express under the terms of the 60-day evaluation license, individuals and teams can explore and use the site immediately.

Each time that you launch the site, the SystemOut log file records the number of days that remain in the evaluation license. The log file is in *wp_profile_root*`/logs/WebSphere_Portal`.

To continue use after the evaluation license expires, you need to purchase WebSphere Portal Express. The purchased production software includes the files that are needed to convert the previously installed evaluation license to a full production license. Custom templates and data files that users created during the evaluation period are fully compatible with the production license, so no additional migration is required.

For purchasing information, contact your local IBM representative or authorized IBM Business Partner.

Database

IBM WebSphere Portal Express includes an Apache Derby database that is configured and ready for immediate use. As a result, you have a running portal that is ready for exploration or portlet and theme development. But for a

production environment or any environment for Web Content Manager, you must use one of the other supported database management systems.

Apache Derby

Derby does not support clustered environments, enabling security in a database-only mode, or vertical cloned environments in which multiple application servers are configured on a single server.

The Derby database that is installed by default is not supported for use in a production environment.

Apache Derby and Web Content Manager

Use one of the other supported databases in a production environment or when you are developing presentation templates or authoring web content. The Derby database can be sufficient for non-production installations of WebSphere Portal Express, the performance of Derby with Web Content Manager is poor. A typical cause of performance issues is transaction timeouts. Although you can increase these timeouts, the resulting performance is prohibitively slow. You must use one of the other supported database management systems. They are better able to handle large amounts of data and can be tuned for performance.

CF07
Apache Derby and Site Builder

The use of Apache Derby with Site Builder is not supported on development or production environments.

Database transfer

Transfer data to another supported database before you use the portal extensively. Large amounts of data in the databases can cause the database transfer to fail if your Java heap size is not large enough. Do not postpone transferring data to another database management system. Waiting to transfer the database can cause errors to occur during the transfer process, such as not having adequate Java heap size.

Configuration Wizard

Use the wizard to either create scripts that you or your database administrator can use to create databases, create database user IDs, and configure database user ID privileges. The wizard collects information about your database management system, the database topology you want, the user IDs you require, and more. Then, it generated custom scripts and instructions.

"Database users" on page 121
There are two types of database users: database configuration users and database runtime users. Become familiar with the privileges required for each user type to work with the database domains of IBM WebSphere Portal Express and the commands for creating database configuration users and granting privileges.

"Database topologies" on page 123
Consider the database configuration options in relation to your IBM WebSphere Portal Express deployment scenario. The complexity of the network topology

increases as you scale from a proof-of-concept environment using IBM DB2 Universal Database™ Workgroup Server Edition to systems using vertical and horizontal cloning techniques.

"Portal database domains" on page 126
Sets of databases tables and schemas for portal resources are called *database domains*. Database domains classify and help you determine how to distribute portal data. There are six database domains: release, customization, community, JCR, feedback, and likeminds.

"JDBC type 2 and type 4 drivers" on page 128
The Configuration Wizard uses JDBC type 4 drivers by default. You can change the default selection in the Configuration Wizard.

Database users

There are two types of database users: database configuration users and database runtime users. Become familiar with the privileges required for each user type to work with the database domains of IBM WebSphere Portal Express and the commands for creating database configuration users and granting privileges.

Database configuration user

The database administration user that is typically created when a database management system (DBMS) is installed is the *database installation user* or the *database configuration user*. The database configuration user is not necessarily the user that is created by default when the database management system is installed. The default user might be used as the database configuration user. The database configuration user is used by WebSphere Portal Express for configuration tasks and creates the database structure that is needed by WebSphere Portal Express. For example, the database configuration user can create database tables and indexes, do database transfer, and often times has operating system privileges, depending on the database management system.

Database runtime user

The *database runtime user* has fewer privileges than the database configuration user. The runtime user has runtime access to the data source of a database and can do basic read and write operations on the data. Consider creating a dedicated runtime user for each database domain of WebSphere Portal Express. If you do not create runtime users, then WebSphere Portal Express uses the configuration user to connect to the databases at run time.

Privileges of database users

The following table identifies the minimum privileges that are needed to correct function by the two types of database users: configuration users and runtime users. The privileges that are listed pertain to all WebSphere Portal Express database domains.

Table 7. List of minimum privileges held by database runtime users for all database domains.

Permission within the database domain	Release	Community	Customization	JCR	Feedback	Likeminds
Access to the database	Yes	Yes	Yes	Yes	Yes	Yes
Read on all tables	Yes	Yes	Yes	Yes	Yes	Yes
Write on all tables	Yes	Yes	Yes	Yes	Yes	Yes
Update on all tables	Yes	Yes	Yes	Yes	Yes	Yes
Delete on all tables	Yes	Yes	Yes	Yes	Yes	Yes
Create tables	No	No	No	No	No	No
Create indexes	No	No	No	No	No	No
Use of sequences	No	No	No	No	Yes	No

Table 8. List of privileges held by database configuration users for all database domains.

Permission within the database domain	Release	Community	Customization	JCR	Feedback	Likeminds
Access to the database	Yes	Yes	Yes	Yes	Yes	Yes
Read on all tables	Yes	Yes	Yes	Yes	Yes	Yes
Write on all tables	Yes	Yes	Yes	Yes	Yes	Yes
Update on all tables	Yes	Yes	Yes	Yes	Yes	Yes
Delete on all tables	Yes	Yes	Yes	Yes	Yes	Yes
Quota on disk to create new objects	Yes	Yes	Yes	Yes	Yes	Yes
Create table spaces	Yes	Yes	Yes	Yes	Yes	Yes
Drop table spaces	Yes	Yes	Yes	Yes	Yes	Yes
Create tables	Yes	Yes	Yes	Yes	Yes	Yes

Table 8. List of privileges held by database configuration users for all database domains. (continued)

Permission within the database domain	Release	Community	Customization	JCR	Feedback	Likeminds
Alter tables	Yes	Yes	Yes	Yes	Yes	Yes
Drop tables	Yes	Yes	Yes	Yes	Yes	Yes
Create indexes	Yes	Yes	Yes	Yes	Yes	Yes
Drop indexes	Yes	Yes	Yes	Yes	Yes	Yes
Create triggers	Yes	Yes	Yes	No	Yes	Yes
Drop triggers	Yes	Yes	Yes	Yes	Yes	Yes
Create sequences	No	No	No	Yes	Yes	No
Use of sequences	No	No	No	Yes	Yes	No
Create types	No	No	No	Yes	No	No
Drop types	No	No	No	Yes	No	No
Create views	No	No	No	No	No	No
Drop views	No	No	No	Yes	No	No

Related information:

"Database Transfer: Granting privileges to database users for DB2 for i" on page 514

Configuration and runtime database users are granted a different set of privileges, depending on whether these users are schema owners or not. You can create a copy of the SQL scripts and edit this copy to manually grant permissions to configuration and runtime database users.

Database topologies

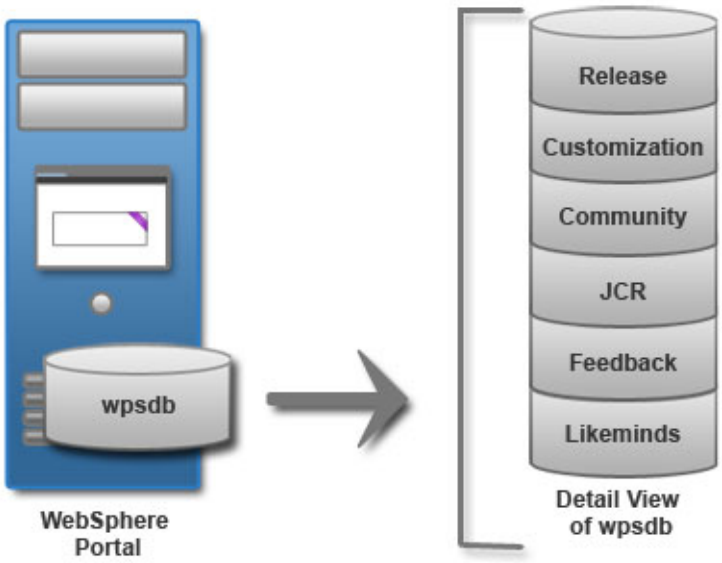
Consider the database configuration options in relation to your IBM WebSphere Portal Express deployment scenario. The complexity of the network topology increases as you scale from a proof-of-concept environment using IBM DB2 Universal Database Workgroup Server Edition to systems using vertical and horizontal cloning techniques.

WebSphere Portal Express data is separated into six portal database domains: release, customization, community, JCR, feedback, and likeminds. The portal database domains facilitate the flexibility that is required to meet different availability requirements. The database topology varies depending on the deployment scenario. A proof-of-concept or development environment has different database topology requirements than a production environment. Review the topologies to determine your portal deployment requirement.

Local database

For proof-of-concept, demonstrations, and development environments, you can use a local database. You can install the database management software on the same server as WebSphere Portal Express. When the database is on the same server as the portal, it is referred to as a *local* database. Using a local database can make administering your environment easier. However, this setup is best used for proof-of-concept deployments only. A local database competes for server resources with your portal.

In the topology diagram, all of the portal database domains are stored in one database, **wpsdb**.

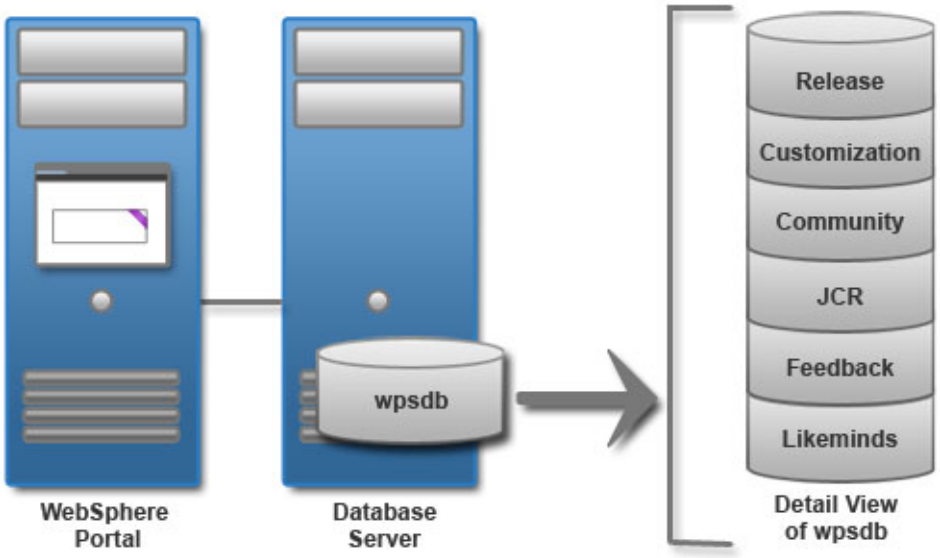


Remote database

For normal load balancing, you can use one or more remote databases. You can install the database management software on a different server from WebSphere Portal Express. When the database is on a different physical server than the portal, it is referred to as a *remote* database. Using a remote database can provide performance benefits, depending on the speed of the network.

When multiple lines of production are involved and each line of production is implemented as a cluster of servers, share portal database domains. Each database domain can be placed on a separate database for efficient maintenance. The release and JCR portal database domains cannot be shared.

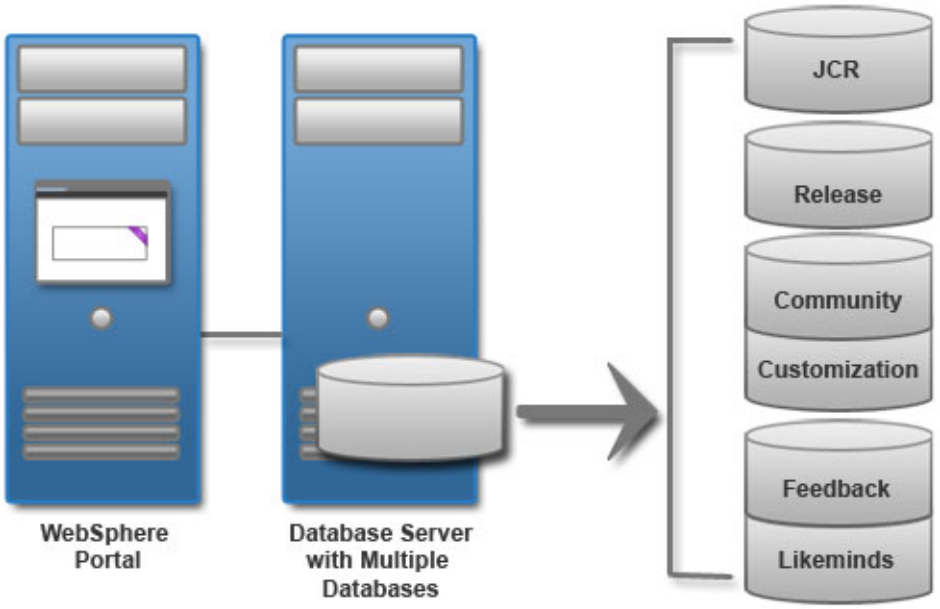
The topology diagram is similar to the local database topology. The main difference is that the database is on a different server than the portal.



High capacity and availability

For high capacity load balancing, use one or multiple remote databases. When you deploy the portal in a large-scale, high-demand environment, you can dedicate a server specifically for database transactions. As more users access the portal, the portal application becomes database intensive. Database activity can take up processor resources and disk I/O time. Separating the database from the server that the portal is running on increases its capacity.

In the topology diagram, there is a remote database server with four databases. The JCR and release portal database domains have unique databases. They cannot be shared. Also, the release portal database domain cannot be taken offline. Another database contains the customization and community portal database domains. Finally, there is another database to contain the feedback and likeminds portal database domains.



Portal database domains

Sets of databases tables and schemas for portal resources are called *database domains*. Database domains classify and help you determine how to distribute portal data. There are six database domains: release, customization, community, JCR, feedback, and likeminds.

The database domains categorize portal data into the following categories and subcategories to help you decide how to distribute portal data into different databases:

Release data (release and JCR)
Includes all portal content definitions, rules, and rights that are designed externally then brought into the portal by a staging process, such as page hierarchy, available portlets and themes, templates, credential slots, Personalization rules, and policies. These resources are typically not modified during production and need administrative rights to do so. Administrators typically create release data on an integration server and stage it to the production system. Release data is protected by access control and contains only data, not code. Release data includes two separate portal database domains: release and JCR.

The release portal database domain contains portal static site configuration, including access control, pages, and portlets.

The JCR portal database domain contains authored content, Personalization rules, and theme policy definitions.

Customization data (customization)
This data is associated with a particular user only and cannot be shared across users or user groups. Typical examples are portlet data or customized pages (implicitly derived pages). Because this data is scoped to a single user only, access control protection is simplified.

Community data (community)
This data includes all resources that are modified during production. Typically, users and groups are allowed to modify or delete shared resources. Community resources are protected by access control.

Configuration data
Configuration data is not store in a portal database domain. It is typically kept in property files. The property files are either protected by file system security or application server administration rights. This data defines the portal server setup, such as database connection, object factories, and deployment descriptors. This type of data typically is constant during the time a server node is running.

Feedback and likeminds portal database domains store data exclusively for the Feedback and LikeMinds applications.

Database schema names

The table includes the default names that are used in the Configuration Wizard. Replace these values with the values in your environment; schema names must be different when the database is shared.

All table spaces are approximately 2.8 GB by default. The size increases with the use of the Java Content Repository function.

For some database software, such as DB2, the database name cannot exceed 8 characters and can contain letters and numbers only.

Table 9. Space required for various databases

Application	Database name	Space required
WebSphere Portal Express Used for the portal (at a minimum) or to hold all data. Stores information about user customization, such as pages, and user profile and login information.	*reldb* *commdb* *custdb*	Depends on the number of users and portal objects, such as pages and portlets.
Personalization,Web Content Manager Contains documents, personalization rules, personalization campaigns, and document library configuration information.	*jcrdb*	Depends on the number and size of Personalization rules and campaigns, and the number and size of items and elements that are created in.Web Content Manager
Feedback Contains the information that is logged by your website for analysis of site activity and generating reports.	*fdbkdb*	Depends on the amount of traffic to the site. The amount of data that is logged per login-enabled page can vary.
LikeMinds Contains the recommendations that are displayed to users. The LikeMinds application analyzes the visitor's interactions with your website are and generates predictions.	*lmdb*	Depends on the amount of traffic to the site.

Database users

The table indicates types of objects that are owned by each user. The architecture allows each of the following users to exist in the same database. All table spaces are approximately 2.8 GB by default. The size increases with the use of Java Content Repository.

Application	Database user	Function
WebSphere Portal Express	*releaseusr* *communityusr* *customizationusr*	Core user who owns approximately 230 tables, which are used for WebSphere Portal Express core objects, which include tables that store the user customizations that are made to pages.
Java Content Repository	*jcr*	Java Content Repository user who owns approximately 100 tables. The number might be higher depending on usage.
Feedback	*feedback*	Feedback user who owns approximately 50 tables that are used for logging site and personalization usage.

Application	Database user	Function
LikeMinds	*likeminds*	LikeMinds user who owns approximately 15 tables that are used to hold the website usage analysis routines and recommendation text.

JDBC type 2 and type 4 drivers

The Configuration Wizard uses JDBC type 4 drivers by default. You can change the default selection in the Configuration Wizard.

During step 2, Customize Values, click **Advanced** to change the driver type.

When you use a JDBC type 2 connection, WebSphere Portal Express and DB2 Connect are installed on one system (the local system). The DB2 server is installed on a different system (the remote system).

When you use a JDBC type 4 connection, you do not need DB2 Connect. Instead, the DB2 Universal JDBC driver that is supplied with DB2 is copied to portal server. It is used within the Java virtual machine (JVM) of WebSphere Portal Express and connects directly to the remote DB2 server.

Depending on your database software, there might be more required configuration steps for using type 2 drivers. Review the installing database software topics.

Related tasks:

"Installing and preparing Oracle or Oracle RAC" on page 151
You can use Oracle or Oracle RAC as the database software.

"Installing and preparing DB2 for z/OS" on page 154
Use this information to install DB2 for z/OS® for use with IBM WebSphere Portal Express.

"Installing and preparing DB2" on page 149
Use this information to install DB2 or DB2 pureScale for use with IBM WebSphere Portal Express.

User registry considerations

A user registry or repository authenticates a user and retrieves information about users and groups to do security-related functions, including authentication and authorization.

User registries store user account information, such as user ID and password, that can be accessed during authentication. User repositories store user profiles and preference information. A user registry or repository is used to:

- Authenticate a user by using basic authentication, identity assertion, or client certificates
- Retrieve user and group information to do security-related administrative functions such as mapping users and groups to security roles

By default, IBM WebSphere Portal Express is installed with a federated repository with a built-in file repository. The federated repository allows you to add various user registries, realm support for Virtual Portals, and/or property extensions to create a single, working unit. The available user registries that you can add to the federated repository are LDAP user registries, database user registries, and custom user registries.

Remember: Using the built-in file repository is not recommended in a production environment. After you add another repository and choose the administrative users from that repository, you must remove the file repository.

Based on the federated repository, WebSphere Portal Express allows you to create a user base that can be federated over multiple repositories: LDAP, DB, and/or custom user registry. It also allows you to define additional attributes in a separate store if your corporate LDAP directory is read-only.

If you are using a federated repository, you must plan on where you want to store new users and groups. By default, new users and groups are stored in the default file repository. If you use use multiple LDAP user registries and database user registries, you must figure out which user registry you want to define as your default user registry where new users and groups are stored. After you add all user registries to your federated repository, you can run the `wp-set-entitytypes` task to set a specific user registry as the default location.

Remember: Before you combine multiple user registries, review the registries for the following limitations and correct any issues:

- Distinguished names must be unique for a realm over all registries. For example, if `uid=wpsadmin,o=yourco` exists in LDAP1, it must not exist in LDAP2, LDAP3, or DB1.
- The short name, for example wpsadmin, should be unique for a realm over all registries.
- The base distinguished names for all registries that are used within a realm must not overlap; for example, if LDAP1 is `c=us,o=yourco`, LDAP2 must not be `o=yourco`.
- Do not leave the base entry blank for any of the registries used within a realm.
- If IBM Domino is one of your user registries in a multiple registry configuration and shares a realm with another user registry, ensure that the groups are stored in a hierarchical format in the Domino Directory as opposed to the default flat-naming structure. For example, the flat-naming convention is `cn=`*`groupName`* and the hierarchical format is `cn=`*`groupName`*`,o=`*`root`*.
- The user must exist in a user registry and not within the property extension configuration; otherwise, the user cannot be a member of the realm.

"User registry options" on page 130
IBM WebSphere Portal Express provides various security configuration tasks. In the past, there was one task and you might not recover from errors. Also, you might not expand your user registry to meet your growing business needs. Now there are multiple tasks and you can fine-tune your system to meet your business needs.

"Virtual Member Manager integration" on page 132
IBM WebSphere Application Server includes the Virtual Member Manager (VMM), which IBM WebSphere Portal Express uses to access user and group information. VMM provides an interface that enables communication between WebSphere Portal Express and any repository, whether federated repositories or your own custom user registry.

"Realm support" on page 134
A realm is a collection of users or groups from one or more branches of your repository tree. Those branches can be part of a single repository, for example an LDAP user registry, or it can be a combination of multiple user registries. A realm is then mapped to a Virtual Portal to allow the realm's user population to log in to the Virtual Portal. This functionality allows you to define areas within WebSphere Portal Express that only a limited set of users can access.

"Property extension" on page 134
Use the property extension, formerly known as the lookaside database, to store extra user attributes into a database store without touching your backend user registry. You can use the property extension if your LDAP is read-only but you want users to specify an extra attribute such as Timezone. You can store this extra attribute in the database store. You can also add extra attributes for an application if you cannot change your repository schema. Property extension can be used with a federated repository or a custom user registry.

User registry options

IBM WebSphere Portal Express provides various security configuration tasks. In the past, there was one task and you might not recover from errors. Also, you might not expand your user registry to meet your growing business needs. Now there are multiple tasks and you can fine-tune your system to meet your business needs.

You have the following general security options to choose from:

Table 10. Security options with explanation

Security option	Explanation
Federated security	With this option, you can create Virtual Portals with multiple realms. You can also use multiple repositories (LDAP, database, custom), and you can add Application Groups to your system. This option is good if you must merge multiple LDAP servers into one cohesive structure. **Attention:** If you plan to enable the transient user feature, you must choose the federated user registry configuration. **Important:** You must take special care that there are no duplicate names between the various repositories. For example, if you installed the product with a Portal Administrator of `admin1`, then `admin1` must not exist in the corporate LDAP server.
Custom security	This option provides you with the ability to write a fully controlled WebSphere Security environment. There is a custom user registry and a custom member adapter for Virtual Member Manager (VMM). The abilities of this option depend on your implementation.

Federated security

WebSphere Portal Express is configured with a default federated repository with a built-in file repository. The federated repository offers you the richest number of options to meet your business needs. You can easily expand your business as your needs grow. For example, your company acquires a new business that has an existing LDAP user registry. You can add that LDAP server to your federated repository. Choose one of the following tasks to enable a production repository:

Table 11. Tasks to enable a production repository

Task	Description
Add a federated LDAP repository to the VMM configuration	Select this option to add an LDAP server to the federated repository. This task does not change the current security assignment. Therefore, the administrative user that is defined during installation is still active.
Add a federated database repository to the VMM configuration	Select this option to add a database to the federated repository. This task does not change the current security assignment. Therefore, the administrative user that is defined during installation is still active.
Add a federated custom user registry	Select this option to add a custom user registry that your company created to the federated repository. This task does not change the current security assignment. Therefore, the administrative user that is defined during installation is still active.

After you add your initial user registry, you can add more user registries to the repository to create a multiple user registry configuration. After you configure your repository, you must remove the default file-based repository. You do not have to remove the file-based repository in a development environment or if you are using IBM Connections. The following tasks are required to remove the default file-based repository:

Table 12. Tasks required to remove the default file-based repository

Task	Description
Change the user registry where users and groups are stored	This task changes the default repository where new users and groups are stored.
Change WebSphere Application Server administrator	This task changes the WebSphere Application Server administrator user ID and password.
Change WebSphere Portal Server administrator	This task changes the WebSphere Portal Express administrator user ID and password.
Delete a federated repository from the VMM configuration	This task deleted the default file-based repository from your configuration.

After you use your federated repository, you might need to manage your user registry. You can run any of the following optional tasks to fine-tune your federated repository:

Table 13. Optional tasks to manage the federated repository

Task	Description
Updating the federated LDAP user registry	Choose this option to update certain parameters such as your bind ID and password to fix issues with your LDAP user registry.
Updating the federated database user registry	Choose this option to update certain parameters such as the data source name, database URL, and database type to fix issues with your database user registry.

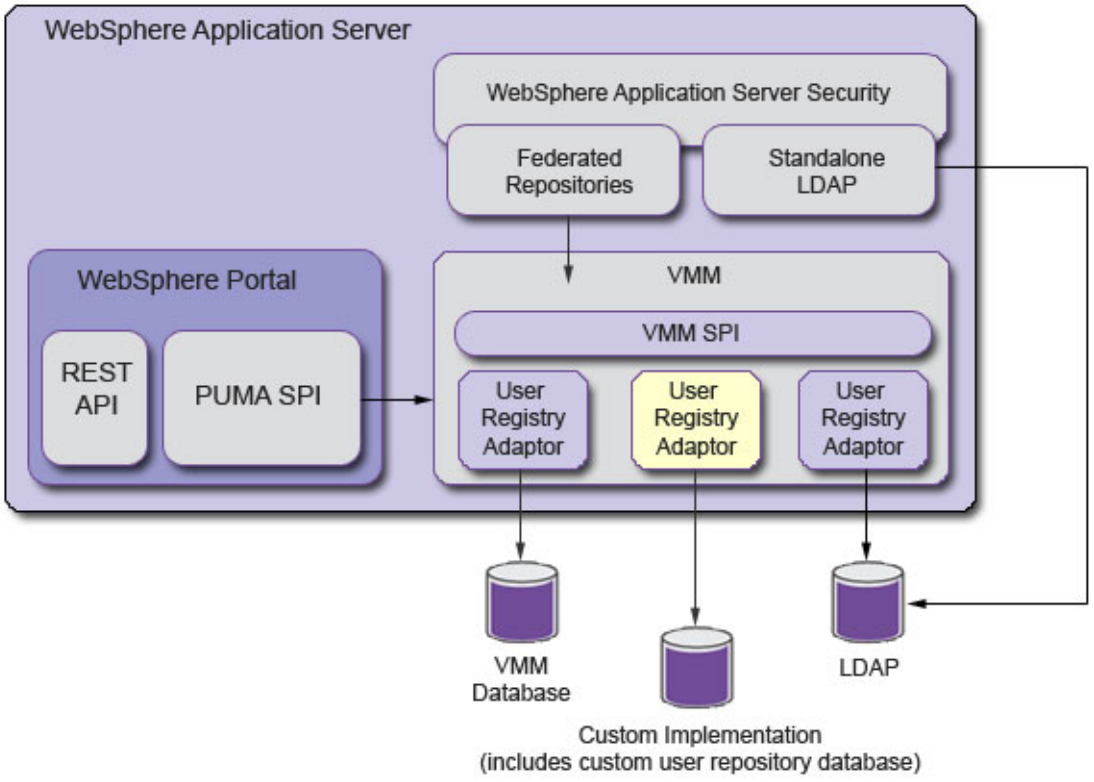
Table 13. Optional tasks to manage the federated repository (continued)

Task	Description
Create a realm	Choose this option to create a realm, which is a group of users from one or more user registries that form a coherent group within WebSphere Portal Express. Realms allow flexible user management with various configuration options. A realm must be mapped to a Virtual Portal to allow the defined users to log in to the Virtual Portal. In a federated repository, you can create multiple realms.

Virtual Member Manager integration

IBM WebSphere Application Server includes the Virtual Member Manager (VMM), which IBM WebSphere Portal Express uses to access user and group information. VMM provides an interface that enables communication between WebSphere Portal Express and any repository, whether federated repositories or your own custom user registry.

The Virtual Member Manager (VMM) is an abstract component within the WebSphere Application Server infrastructure. As the following diagram illustrates, WebSphere Portal Express uses the Portal User Management Architecture (PUMA) System Programming Interface (SPI) to retrieve and set attributes on user objects. PUMA passes these requests to VMM, which then passes the requests on to a corresponding registry adapter that connects VMM to the repository.



The preceding diagram includes the following components:

Federated repositories

An out-of-box implementation of the `UserRegistry` interface that supports multiple repositories. To communicate with the federated repositories, both WebSphere Application Server and WebSphere Portal Express dispatch all operations to VMM.

VMM SPI

VMM offers a Service Provider Interface (SPI), `wim.Repository`, that enables communication with repositories. WebSphere Application Server uses this SPI to connect to federated repositories. WebSphere Portal Express uses this SPI to connect to all repositories.

User registry adapter

An implementation of the VMM SPI that enables VMM to connect to a specific repository, whether an LDAP directory, database, files, or other repository. Registry adapters enable communication between WebSphere Portal Express and any repository.

Important: You must create a user registry adapter if you plan to use a custom user registry or repository that WebSphere Portal Express does not support out-of-box. To create a user registry adapter, implement the `wim.Repository` interface. Refer to the following topics in the WebSphere Application Server Information Center for information and instructions:

Repository SPI (System programming interfaces for virtual member manager adapters)

Sample custom adapters for federated repositories examples

Related tasks:

"Setting up custom user repositories" on page 1615
A custom user repository is any repository that WebSphere Portal Express does not support out-of-box. However, you can configure WebSphere Portal Express to support any type of repository in a federated or stand-alone user registry, whether an LDAP directory, database, file system, and so on. Setting up custom user repositories involves tasks such as defining additional repositories to the default federated user registry, creating a custom stand-alone user repository, and updating your user repository to reflect changes in your environment. Learn what steps are required to create and update custom user repositories and what specific interfaces you must implement to enable communication between WebSphere Portal Express and a repository.

Related information:

Webcast replay of WebSphere Portal WMM to VMM comparison

Setting up a custom user repository with Virtual Member Manager for IBM WebSphere Application Server and IBM WebSphere Portal

IBM WebSphere Developer Technical Journal: Expand your user registry options with a federated repository in WebSphere Application Server, Using the Virtual Member Manager

Virtual member manager APIs

Sample custom adapters for federated repositories examples

 Repository SPI

Realm support

A realm is a collection of users or groups from one or more branches of your repository tree. Those branches can be part of a single repository, for example an LDAP user registry, or it can be a combination of multiple user registries. A realm is then mapped to a Virtual Portal to allow the realm's user population to log in to the Virtual Portal. This functionality allows you to define areas within WebSphere Portal Express that only a limited set of users can access.

For example, if you are an international company with employees in Asia, Europe, USA, and Canada, you might have an application or information that applies only to a subset of these employees. You can create a subset of employees and create a Virtual Portal that contains the application or information for that realm. Users from one realm cannot access another realm unless they are also members of that realm. For example, the wpsadmin user cannot log in to a Virtual Portal unless the wpsadmin user is a member of the corresponding realm.

You can create a realm that combines users from your various user registries; for example, your realm can span three LDAP user registries and a database user registry: LDAP1, LDAP2, LDAP3, and DB1.

Remember: Before you combine multiple user registries, review the registries for the following limitations and correct any issues:

- Distinguished names must be unique for a realm over all registries. For example, if `uid=wpsadmin,o=yourco` exists in LDAP1, it must not exist in LDAP2, LDAP3, or DB1.
- The short name, for example wpsadmin, should be unique for a realm over all registries.
- The base distinguished names for all registries that are used within a realm must not overlap; for example, if LDAP1 is `c=us,o=yourco`, LDAP2 must not be `o=yourco`.
- Do not leave the base entry blank for any of the registries used within a realm.
- If IBM Domino is one of your user registries in a multiple registry configuration and shares a realm with another user registry, ensure that the groups are stored in a hierarchical format in the Domino Directory as opposed to the default flat-naming structure. For example, the flat-naming convention is `cn=`*`groupName`* and the hierarchical format is `cn=`*`groupName`*`,o=`*`root`*.
- The user must exist in a user registry and not within the property extension configuration; otherwise, the user cannot be a member of the realm.

Property extension

Use the property extension, formerly known as the lookaside database, to store extra user attributes into a database store without touching your backend user registry. You can use the property extension if your LDAP is read-only but you want users to specify an extra attribute such as Timezone. You can store this extra attribute in the database store. You can also add extra attributes for an application if you cannot change your repository schema. Property extension can be used with a federated repository or a custom user registry.

IBM Web Content Manager stores extra information for the following features:

- Web content user profiling
- Category selection trees

If this information cannot be stored in the main repository, for example the main repository is read-only, a property extension configuration is required.

WebSphere Portal Express high availability

IBM WebSphere Portal Express is licensed for use in a single-server configuration and might not be used in either a cloned configuration or a clustered configuration except when implementing idle standby for the purpose of failover.

In an idle standby configuration, a server is considered idle if it is used exclusively for administrative needs and to help a failover situation. WebSphere Portal Express is installed on the idle standby server, but it is not operational to service user transactions or to query workloads.

Implementing idle standby requires the purchase of a separate WebSphere Portal Express Idle Standby Part Number, in addition to licensing the primary server, regardless of whether your primary servers are currently licensed under the per User License Option or the per Processor Value Unit License Option.

Once you have an Idle Standby License Option, you can use WebSphere Portal Express in an idle standby configuration. To achieve failover, implement a primary node and a secondary node. The primary node can be configured to be active always; the secondary node becomes active only if the primary node fails.

Remember: For the deployment manager and each WebSphere Portal Express node to be in the cluster, verify that each system clock is set to within 5 minutes of the others or the `addNode` command fails.

The deployment scenario, Deploying WebSphere Portal Express for high availability using idle standby, in the WebSphere Portal Express wiki provides information about setting up an idle standby configuration.

Getting the software

There are different ways for you to get WebSphere Portal Express and Web Content Manager Version 8.5 software.

"Physical media"
WebSphere Portal Express is available on DVD. Installing from physical media is practical when installing on a limited number of servers.

"Live repository" on page 136
You can download and install WebSphere Portal Express directly from a live repository. All that you need is IBM Installation Manager and you can select the portal repository from the list.

"Electronic images" on page 138
Use Passport Advantage for downloading the electronic images of WebSphere Portal Express. Installing is simpler than ever before. IBM Installation Manager makes it easy. The following sections, formerly referred to as download documents, provide download information for the different WebSphere Portal Express and Web Content Manager offerings.

Physical media

WebSphere Portal Express is available on DVD. Installing from physical media is practical when installing on a limited number of servers.

Benefits:

This option has the following benefits and is best if performing a limited number of installations:

- No prerequisite steps required before installation
- Provides a seamless way to install IBM Installation Manager if it is not currently installed
- Provides an installation path if a file server is not available or is slower than the DVD

Limitations:

- Installing from the DVD on IBM i is not supported
- Silently installing from the DVD is not supported

Linux note: For security reasons, some Linux versions, such as Red Hat Enterprise Linux, prevent programs from running from automounted DVDs. Thus, not allowing to run the installer from the DVD. To fix this issue, add the `/dev/hdc /media/ auto pamconsole,fscontext=system_u:object_r:removable_t,exec,noauto,managed 0 0` line to the end of the `/etc/fstab` file to update file system table.

Live repository

You can download and install WebSphere Portal Express directly from a live repository. All that you need is IBM Installation Manager and you can select the portal repository from the list.

About this task

After you enter your credentials in IBM Installation Manager, you can see the software repositories that you are entitled to download and install. This option is ideal when you perform installations at a large scale.

Depending on your operating system, choose one of the following paths to install directly from a live repository:

- From the IBM Installation Manager graphical user interface, complete the following steps:
  1. Click **File** > **Preferences** > **Passport Advantage**.
  2. Select the **Connect to Passport Advantage** check box.
  3. Click **Apply**.
  4. Click **OK**.
  5. On the main Installation Manager panel, select **Install**.
  6. In the Password Required dialog box, enter your credentials.
  7. Click **OK**.
  8. Select the package that you want to download and install.
     - If you are installing Portal to an existing copy of WebSphere Application Server, then select only **IBM WebSphere Portal Server.** To complete this installation option, your existing copy of WebSphere Application Server must meet the following requirements:
       - WebSphere Application Server Version 8.5.5.2 or later
       - Java Version 7
       - No existing Portal profile

– If you are not installing Portal to an existing copy of WebSphere Application Server, then select the following packages:
  - **IBM WebSphere Application Server Network Deployment**
  - **IBM WebSphere Portal Server**
  - **IBM WebSphere SDK Java Technology Edition**

    Note: The **IBM WebSphere SDK Java Technology Edition** option is required for a WebSphere Portal Express installation even though it might be marked as "Optional."

  If you have a WebSphere Portal Express Enable license, you must select both the **IBM WebSphere Portal Express** and **IBM WebSphere Portal Enable** packages.

Tip: If you have a WebSphere Portal Express Enable license, you must select both the **IBM WebSphere Portal Express** and **IBM WebSphere Portal Enable** packages.

- From the IBM Installation Manager console user interface, complete the following steps:
  1. Enter `P` to go to the **Preferences** menu.
  2. Enter `6` to go to the **Passport Advantage** menu.
  3. Enter `1` to connect to Passport Advantage.
  4. Enter `P` to enter your user name and password.
  5. Enter `0` for OK.
  6. Enter `A` to apply your changes and return to the **Preferences** menu.
  7. Enter `R` to return to the **Main** menu.
  8. Enter `1` to see all of the repositories that you are entitled to download and install.
  9. Enter the number of the package that you want to download and install.
     – If you are installing Portal to an existing copy of WebSphere Application Server, then select only **IBM WebSphere Portal Server.** To complete this installation option, your existing copy of WebSphere Application Server must meet the following requirements:
       - WebSphere Application Server Version 8.5.5.2 or later
       - Java Version 7
       - No existing Portal profile
     – If you are not installing Portal to an existing copy of WebSphere Application Server, then select the following packages:
       - **IBM WebSphere Application Server Network Deployment**
       - **IBM WebSphere Portal Server**
       - **IBM WebSphere SDK Java Technology Edition**

         Note: The **IBM WebSphere SDK Java Technology Edition** option is required for a WebSphere Portal Express installation even though it might be marked as "Optional."

     If you have a WebSphere Portal Express Enable license, you must select both the **IBM WebSphere Portal Express** and **IBM WebSphere Portal Enable** packages.
  10. Enter `1` to install the package that you selected.

Related tasks:

Running the installation program
Use the IBM Installation Manager program to install the software. IBM Installation Manager provides multiple user interfaces. Select the one that you are more comfortable with. You can use the graphical user interface or the console to create a response file for automated installations.

Electronic images

Use Passport Advantage for downloading the electronic images of WebSphere Portal Express. Installing is simpler than ever before. IBM Installation Manager makes it easy. The following sections, formerly referred to as download documents, provide download information for the different WebSphere Portal Express and Web Content Manager offerings.

"Getting WebSphere Portal Express software"
This section describes how to download and assemble IBM WebSphere Portal Express Version 8.5 components by using the Passport Advantage Online website.

Getting WebSphere Portal Express software

This section describes how to download and assemble IBM WebSphere Portal Express Version 8.5 components by using the Passport Advantage Online website.

Downloadable eAssembly images

The IBM WebSphere Portal Express Version 8.5 product consists of multiple offerings and each offering includes multiple downloadable electronic images that are packaged by platform. Each platform package is called an eAssembly and each eAssembly contains all of the electronic images that are required for that offering to run on that platform. The following table shows the eAssembly numbers for each of the IBM WebSphere Portal Express Version 8.5 offerings.

Important: When you expand the eAssembly files, ensure that you expand the files into the same parent directory to create a complete installable image. For example, if you expand into a parent directory that is called *download_directory*, the following directory structure results:

download_directory/Setup
download_directory/IBMJAVA7
download_directory/WAS8552
download_directory/WP85_Express

Failure to expand all files in the same parent directory results in an installation failure.

The following directories might also exist but are not required to be expanded in the same parent directory:

download_directory/qsg_850_portal_wcm
download_directory/WP85_RemoteSearch

Required software

Table 14. List of eAssembly images for IBM WebSphere Portal and Web Content Manager V8.5 Multiplatform Multilingual Quick Start Guide

Part number	Description
CIYE7ML	IBM WebSphere Portal and Web Content Manager V8.5 Multiplatform Multilingual Quick Start Guide

Table 15. List of eAssembly images for IBM WebSphere Portal Express V8.5 and IBM WebSphere Application Server Setup V8.5.5.2

Part number	Description
CIZ5VML	IBM WebSphere Portal Extend Setup V8.5 Multiplatform Multilingual IMPORTANT: All parts of this image (Setup, Install, WAS ND, SDK) must be extracted into the same directory to create an installable image.
CIZ5WML	IBM WebSphere Portal Extend Install V8.5 Multiplatform Multilingual IMPORTANT: All parts of this image (Setup, Install, WAS ND, SDK) must be extracted into the same directory to create an installable image.
CIYW0ML	IBM WebSphere Application Server Network Deployment V8.5.5.2 Multiplatform Multilingual IMPORTANT: All parts of this image (Setup, Install, WAS ND, SDK) must be extracted into the same directory to create an installable image.
CIYW1ML	IBM WebSphere SDK Java Technology Edition V7.0.6.1 Multiplatform Multilingual IMPORTANT: All parts of this image (Setup, Install, WAS ND, SDK) must be extracted into the same directory to create an installable image.
CIYW2ML	IBM WebSphere Portal V8.5 Remote Search and Document Conversion Services Multiplatform Multilingual

Optional software

Table 16. List of eAssembly images for IBM WebSphere Application Server Network Deployment V8.5.5 Supplements and Edge Components

Part number	Description
CIK1VML	IBM WebSphere Application Server V8.5.5 Supplements (1 of 3) for Multiplatform Multilingual
CIK1WML	IBM WebSphere Application Server V8.5.5 Supplements (2 of 3) for Multiplatform Multilingual
CIK1XML	IBM WebSphere Application Server V8.5.5 Supplements (3 of 3) for Multiplatform Multilingual
CIK2NML	IBM WebSphere Edge Components: Load Balancer for IPv4 and IPv6 (for WebSphere Application Server Network Deployment V8.5.5) Multiplatform Multilingual
CIL5DML	IBM WebSphere Edge Components: Load Balancer for IPv4 (for WebSphere Application Server Network Deployment V8.5.5) Multiplatform, Multilingual
CIL5EML	IBM WebSphere Edge Components: Caching Proxy (for WebSphere Application Server Network Deployment V8.5.5)Multiplatform, Multilingual

Table 17. List of eAssembly images for IBM DB2 Workgroup Server Edition 10.5

Part number	Description
CIXU9ML	IBM® DB2® Workgroup Server Edition - Restricted Use 10.5 for Linux® on AMD64 and Intel® EM64T systems (x64)
CIWN3ML	IBM® DB2® Workgroup Server Edition - Restricted Use 10.5 for Linux® on POWER® (System i® and System p®) systems
CIWN4ML	IBM® DB2® Workgroup Server Edition - Restricted Use 10.5 for Linux® on z Systems®
CIWN5ML	IBM® DB2® Workgroup Server Edition - Restricted Use 10.5 for AIX®
CIWN6ML	IBM® DB2® Workgroup Server Edition - Restricted Use 10.5 for HP-UX on HP Integrity Itanium-based systems
CIWN7ML	IBM® DB2® Workgroup Server Edition - Restricted Use 10.5 for Solaris on UltraSPARC systems
CIWN8ML	IBM® DB2® Workgroup Server Edition - Restricted Use 10.5 for Solaris on x64 systems
CIWN9ML	IBM® DB2® Workgroup Server Edition - Restricted Use 10.5 for Windows® on AMD64 and Intel® EM64T systems (x64)
CIWM5ML	IBM® DB2® National Language Pack 10.5 for Linux® on AMD64 and Intel® EM64T systems (x64)
CIWM6ML	IBM® DB2® National Language Pack 10.5 for Linux® on POWER® (System i® and System p®) systems
CIWM7ML	IBM® DB2® National Language Pack 10.5 for Linux® on z Systems®
CIWM8ML	IBM® DB2® National Language Pack 10.5 for AIX®
CIWM9ML	IBM® DB2® National Language Pack 10.5 for HP-UX on HP Integrity Itanium-based systems
CIWN0ML	IBM® DB2® National Language Pack 10.5 for Solaris on UltraSPARC systems
CIWN1ML	IBM® DB2® National Language Pack 10.5 for Solaris on x64 systems

Table 18. List of eAssembly images for IBM Tivoli Directory Integrator V7.1.1

Part number	Description
CZUE5ML	IBM Tivoli Directory Integrator V7.1.1 Quick Start Guide
CZUF0ML	IBM Tivoli Directory Integrator Identity Edition V7.1.1 for Windows - x86
CZUF7ML	IBM Tivoli Directory Integrator Identity Edition V7.1.1 for Windows - x86-64
CZUF1ML	IBM Tivoli Directory Integrator Identity Edition V7.1.1 for AIX
CI772ML	IBM Tivoli Directory Integrator Identity Edition V7.1.1 for AIX - PPC64
CZUF2ML	IBM Tivoli Directory Integrator Identity Edition V7.1.1 for Linux Linux - x86
CZUF3ML	IBM Tivoli Directory Integrator Identity Edition V7.1.1 for Linux Linux - x86-64
CZUE9ML	IBM Tivoli Directory Integrator Identity Edition V7.1.1 for Linux - zSeries & s/390
CZUF5ML	IBM Tivoli Directory Integrator Identity Edition V7.1.1 for Linux - POWER (i/p Series)
CZUF4ML	IBM Tivoli Directory Integrator Identity Edition V7.1.1 for Solaris - SPARC
CZUE6ML	IBM Tivoli Directory Integrator Identity Edition V7.1.1 for Solaris -Opteron
CZUE7ML	IBM Tivoli Directory Integrator Identity Edition V7.1.1 for HP-UX - Integrity

Table 19. List of eAssembly images for IBM Security Directory Server V6.3.1

Part number	Description
CIS0HML	IBM Security Directory Server V6.3.1 win-base.zip (with ENT, Only IM based Installer, supports 32/64 bit)
CIS0JML	IBM Security Directory Server V6.3.1 win-IM.zip (contains ibm_IM_32bit, ibm_IM_64bit)
CIS0KML	IBM Security Directory Server V6.3.1 win-jdk.zip (contains ibm_jdk_32bit, ibm_jdk_64bit)
CIS0LML	IBM Security Directory Server V6.3.1 win-db2.zip (contains ibm_db2_32bit, ibm_db2_64bit)
CIS0MML	IBM Security Directory Server V6.3.1 win-ewas.zip (contains ibm_ewas_32bit, ibm_ewas_64bit)
CIS0NML	IBM Security Directory Server V6.3.1 win-gskit.zip (contains ibm_gskit_32bit, ibm_gskit_64bit)
CIS0PML	IBM Security Directory Server V6.3.1 win.iso (with ENT, IM based Installer, pre-req installables)
CIS0RML	IBM Security Directory Server V6.3.1 aix-ppc64-base.tar (with ENT, IM based Installer)
CIS0TML	IBM Security Directory Server V6.3.1 aix-ppc64-IM.tar
CIS0UML	IBM Security Directory Server V6.3.1 aix-ppc64-jdk.tar
CIS0VML	IBM Security Directory Server V6.3.1 aix-ppc64-db2.tar
CIS0WML	IBM Security Directory Server V6.3.1 aix-ppc64-ewas.tar
CIS0XML	IBM Security Directory Server V6.3.1 aix-ppc64-gskit.tar
CIS0YML	IBM Security Directory Server V6.3.1 aix-ppc64.iso (with ENT, IM based Installer, pre-req installables)
CIS10ML	IBM Security Directory Server V6.3.1 linux-x86-64-base.tar (with ENT, IM based Installer)
CIS12ML	IBM Security Directory Server V6.3.1 linux-x86-64-IM.tar
CIS13ML	IBM Security Directory Server V6.3.1 linux-x86-64-jdk.tar
CIS14ML	IBM Security Directory Server V6.3.1 linux-x86-64-db2.tar
CIS15ML	IBM Security Directory Server V6.3.1 linux-x86-64-ewas.tar
CIS16ML	IBM Security Directory Server V6.3.1 linux-x86-64-gskit.tar
CIS17ML	IBM Security Directory Server V6.3.1 linux-x86-64.iso (with ENT, IM based Installer, pre-req installables)
CIS1HML	IBM Security Directory Server V6.3.1 linux-x86-base.tar (with ENT, Native Installer)
CIS1JML	IBM Security Directory Server V6.3.1 linux-x86-jdk.tar
CIS1KML	IBM Security Directory Server V6.3.1 linux-x86-db2.tar
CIS1LML	IBM Security Directory Server V6.3.1 linux-x86-ewas.tar
CIS1MML	IBM Security Directory Server V6.3.1 linux-x86-gskit.tar
CIS1NML	IBM Security Directory Server V6.3.1 linux-x86.iso (with ENT, native installer, pre-req installables)
CIS19ML	IBM Security Directory Server V6.3.1 solaris-x86-64-base.tar (with ENT, Native Installer)
CIS1BML	IBM Security Directory Server V6.3.1 solaris-x86-64-jdk.tar
CIS1CML	IBM Security Directory Server V6.3.1 solaris-x86-64-db2.tar
CIS1DML	IBM Security Directory Server V6.3.1 solaris-x86-64-ewas.tar
CIS1EML	IBM Security Directory Server V6.3.1 solaris-x86-64-gskit.tar
CIS1FML	IBM Security Directory Server V6.3.1solaris-x86-64.iso (with ENT, native installer, pre-req installables)
CIS1QML	IBM Security Directory Server V6.3.1 solaris-sparc-base.tar (with ENT, Native Installer)

Table 19. List of eAssembly images for IBM Security Directory Server V6.3.1 (continued)

Part number	Description
CIS1SML	IBM Security Directory Server V6.3.1 solaris-sparc-jdk.tar
CIS1TML	IBM Security Directory Server V6.3.1 solaris-sparc-db2.tar
CIS1UML	IBM Security Directory Server V6.3.1 solaris-sparc-ewas.tar
CIS1VML	IBM Security Directory Server V6.3.1 solaris-sparc-gskit.tar
CIS1WML	IBM Security Directory Server V6.3.1 solaris-sparc.iso (with ENT, native installer, pre-req installables)
CIS1ZML	IBM Security Directory Server V6.3.1 linux-ppc64-base.tar (with ENT, Native Installer)
CIS21ML	IBM Security Directory Server V6.3.1 linux-ppc64-jdk.tar
CIS22ML	IBM Security Directory Server V6.3.1 linux-ppc64-db2.tar
CIS23ML	IBM Security Directory Server V6.3.1 linux-ppc64-ewas.tar
CIS24ML	IBM Security Directory Server V6.3.1 linux-ppc64-gskit.tar
CIS26ML	IBM Security Directory Server V6.3.1 linux-ppc64.iso (with ENT, native installer, pre-req installables)
CIS28ML	IBM Security Directory Server V6.3.1 linux-s390x-base.tar (with ENT, Native Installer)
CIS2AML	IBM Security Directory Server V6.3.1 linux-s390x-jdk.tar
CIS2BML	IBM Security Directory Server V6.3.1 linux-s390x-db2.tar
CIS2CML	IBM Security Directory Server V6.3.1 linux-s390x-ewas.tar
CIS2DML	IBM Security Directory Server V6.3.1 linux-s390x-gskit.tar
CIS2EML	IBM Security Directory Server V6.3.1 linux-s390x.iso (with ENT, native installer, pre-req installables)

Table 20. List of eAssembly images for IBM Web Experience Factory V8.5

Part number	Description
CIYW5ML	IBM Web Experience Factory V8.5 Multiplatform Multilingual QuickStart
CIYW6ML	IBM Web Experience Factory V8.5 Multiplatform Multilingual

Table 21. List of eAssembly images for IBM Connections V4.5

Part number	Description
CIHC4ML	IBM Connections V4.5 Quick Start Guide for AIX, Windows, Linux, IBMi Multilingual
CIHC5ML	IBM Connections V4.5 for Windows Multilingual
CII8MML	IBM Connections V4.5 for AIX Multilingual
CIHC6ML	IBM Connections V4.5 for Linux Multilingual
CIHC7ML	IBM Connections V4.5 Linux for z Systems Multilingual
CIHC8ML	IBM Connections V4.5 Wizard for Windows Multilingual
CIHC9ML	IBM Connections V4.5 Wizard for Linux, AIX Multilingual

Table 22. List of eAssembly images for IBM Connections Content Manager 4.5

Part number	Description
CIHC4ML	IBM Connections V4.5 Quick Start Guide for AIX, Windows, Linux, IBMi Multilingual
CIQ52ML	IBM FileNet Content Engine V5.2 AIX Multilingual
CIQ55ML	IBM FileNet Content Engine V5.2 Linux Multilingual
CIQ57ML	IBM FileNet Content Engine V5.2 Windows Multilingual
CIQ58ML	IBM FileNet Content Engine V5.2 Linux on z Systems Multilingual
CIQ5DEN	IBM FileNet Content Engine Client V5.2 Linux English
CIQ5FEN	IBM FileNet Content Engine Client V5.2 Windows English
CIQ5GEN	IBM FileNet Content Engine Client V5.2 Linux on z Systems English
CIQ5AEN	IBM FileNet Collaboration Services V2.0 Win
CIG7ZML	IBM FileNet Collaboration Services V2.0 Win
CIG80ML	IBM FileNet Collaboration Services V2.0 AIX
CIG84ML	IBM FileNet Collaboration Services V2.0 Linux
CIG8CML	IBM FileNet Collaboration Services V2.0 Linux on z Systems

Table 23. List of eAssembly images for IBM Sametime Limited Use 9.0

Part number	Description
CIQ43ML	IBM Sametime V9.0 Multiplatform Multilingual Quickstart Guide
CITN1ML	IBM Sametime Connect Client Limited Use V9.0 Windows, x86 Linux, Mac Multilingual
CITN2ML	IBM Sametime Community Server Limited Use V9.0 Windows Multilingual
CITN3ML	IBM Sametime Community Server Limited Use V9.0 AIX, Linux Multilingual
CITN4ML	IBM Sametime Community Server Limited Use V9.0 IBM i Multilingual
CIQ48ML	IBM Sametime Proxy Server V9.0 Windows, x86 Linux, AIX, IBMi Multilingual
CIQ4CML	IBM Sametime System Console Server V9.0 Windows, x86 Linux, AIX, IBMi Multilingual
CIQ6JML	IBM WebSphere V8.5.5.0 iFixes for IBM Sametime V9.0 Windows, x86 Linux, AIX, IBMi Multilingual
CIB8QML	Quick Start Guide - IBM Notes and Domino V9.0 Multilingual
CIB8REN	IBM Notes and Domino V9.0 Release Notes English
CIB9AEN	IBM Notes, Domino Designer and Admin V9.0 for Windows XP,Vista and Windows 7 32 Bit English
CIBM0EN	IBM Domino Server V9.0 32 Bit for Windows English
CIBM3EN	IBM Domino Server V9.0 32 bit for AIX English
CIBM4EN	IBM Domino Server V9.0 32 bit for Linux for xSeries English
CIBM7EN	IBM Domino Server V9.0 for IBM i English
CINK4CS	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Czech
CINK5HU	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Hungarian
CINK6RU	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Russian
CINK7EL	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Greek
CINK8TR	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Turkish
CINK9AR	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Arabic
CINL0PT	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Portuguese
CINL1PL	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Polish
CINL2TH	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Thai
CINL3SI	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Slovenian
CINL4SK	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Slovakian
CILI1FI	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Finnish
CILI2NO	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Norwegian
CILI3SV	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Swedish
CILI4DA	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Danish
CILI5NL	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Dutch
CIJ7CCA	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Catalan
CIJ7DDE	IBM Notes, Domino Designer and Admin 9.0 for Windows XP, Windows 7 and Windows 8 German
CIJ7EKO	IBM Notes, Domino Designer and Admin 9.0 for Windows XP, Windows 7 and Windows 8 Korean
CIJ7FSC	IBM Notes, Domino Designer and Admin 9.0 for Windows XP, Windows 7 and Windows 8 Simplified Chinese
CIJ7GES	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Spanish
CIJ7HFR	IBM Notes, Domino Designer and Admin 9.0 for Windows XP, Windows 7 and Windows 8 French
CIJ7ITC	IBM Notes, Domino Designer and Admin 9.0 for Windows XP, Windows 7 and Windows 8 Traditional Chinese
CIJ7JJA	IBM Notes, Domino Designer and Admin 9.0 for Windows XP, Windows 7 and Windows 8 Japanese
CIJ7KIT	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Italian
CIJ7LBP	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Brazilian Portuguese
CILS9FI	IBM Domino 9.0 Language Pack for Windows 2008/2012 and IBM Power Systems (Multi O/S Install)Finnish
CILT4FI	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install Finnish
CILT0NO	IBM Domino 9.0 Language Pack for Windows 2008/2012 and IBM Power Systems (Multi O/S Install)Norwegian
CILT5NO	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install Norwegian

Table 23. List of eAssembly images for IBM Sametime Limited Use 9.0 (continued)

Part number	Description
CILT1SV	IBM Domino 9.0 Language Pack for Windows 2008/2012 and IBM Power Systems (Multi O/S Install)Swedish
CILT6SV	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install Swedish
CILT2DA	IBM Domino 9.0 Language Pack for Windows 2008/2012 and IBM Power Systems (Multi O/S Install)Danish
CILT7DA	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install Danish
CILT3NL	IBM Domino 9.0 Language Pack for Windows 2008/2012 and IBM Power Systems (Multi O/S Install)Dutch
CIJ9RJA	IBM Domino Server V9.0 Language Pack for Windows 2008, Windows 2012 and IBM Power Systems Multi O/S Install Japanese
CIJA9JA	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install Japanese
CIJ9TEN	IBM Domino Server V9.0 Language Pack for Windows 2008, Windows 2012 and IBM Power Systems Multi O/S Install English
CIJB1EN	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install English
CIJ9LKO	IBM Domino Server V9.0 Language Pack for Windows 2008, Windows 2012 and IBM Power Systems Multi O/S Install Korean
CIJA4KO	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install Korean
CINQ9CS	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Czech
CINS0CS	IBM Domino Server V9.0 Language Pack for Linux on xSeries Czech
CINS7TH	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Thai
CINS9TH	IBM Domino Server V9.0 Language Pack for AIX Thai
CIJ9MSC	IBM Domino Server V9.0 Language Pack for Windows 2008, Windows 2012 and IBM Power Systems Multi O/S Install Simplified Chinese
CIJA5SC	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install Simplified Chinese
CIJ9QTC	IBM Domino Server V9.0 Language Pack for Windows 2008, Windows 2012 and IBM Power Systems Multi O/S Install Traditional Chinese
CIJA8TC	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install Traditional Chinese
CIJ9JCA	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Catalan
CIJ9KDE	IBM Domino Server V9.0 Language Pack for Windows 2008, Windows 2012 and IBM Power Systems Multi O/S Install German
CIJA3DE	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install German
CIJ9SIT	IBM Domino Server V9.0 Language Pack for Windows 2008, Windows 2012 and IBM Power Systems Multi O/S Install Italian
CIJB0IT	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install Italian
CIJ9PFR	IBM Domino Server V9.0 Language Pack for Windows 2008, Windows 2012 and IBM Power Systems Multi O/S Install French
CIJA7FR	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install French
CIJ9NES	IBM Domino Server V9.0 Language Pack for Windows 2008, Windows 2012 and IBM Power Systems Multi O/S Install Spanish
CIJA6ES	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install Spanish
CIJ9UBP	IBM Domino Server V9.0 Language Pack for Windows 2008, Windows 2012 and IBM Power Systems Multi O/S Install Brazilian Portuguese
CIJB2BP	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install Brazilian Portuguese
CINQ8AR	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Arabic

Table 23. List of eAssembly images for IBM Sametime Limited Use 9.0 (continued)

Part number	Description
CINR9AR	IBM Domino Server V9.0 Language Pack for Linux on xSeries Arabic
CINR0EL	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Greek
CINS1EL	IBM Domino Server V9.0 Language Pack for Linux on xSeries Greek
CINR1IW	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Hebrew
CINS2IW	IBM Domino Server V9.0 Language Pack for Linux on xSeries Hebrew
CINR2HU	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Hungarian
CINT0HU	IBM Domino Server V9.0 Language Pack for AIX and Linux on xSeries Hungarian
CINR7SK	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Slovakian
CINR8SI	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Slovenian
CINS8KZ	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Kazakh
CINR3PT	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Portuguese
CINR4PL	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Polish
CINT1PL	IBM Domino Server V9.0 Language Pack for AIX and Linux on xSeries Polish
CINR5RU	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Russian
CINS4RU	IBM Domino Server V9.0 Language Pack for Linux on xSeries Russian
CINR6TR	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Turkish
CINS5TR	IBM Domino Server V9.0 Language Pack for Linux on xSeries Turkish
CINS3PT	IBM Domino Server V9.0 Language Pack for Linux on xSeries Portuguese
CIK2NML	IBM WebSphere Edge Components: Load Balancer for IPv4 and IPv6 (for WebSphere Application Server Network Deployment V8.5.5) Multiplatform Multilingual
CIL5DML	IBM WebSphere Edge Components: Load Balancer for IPv4 (for WebSphere Application Server Network Deployment V8.5.5) Multiplatform, Multilingual
CIL5EML	IBM WebSphere Edge Components: Caching Proxy (for WebSphere Application Server Network Deployment V8.5.5)Multiplatform, Multilingual
CIK2HML	IBM WebSphere Application Server Network Deployment V8.5.5 (1 of 3) for Multiplatform Multilingual
CIK2IML	IBM WebSphere Application Server Network Deployment V8.5.5 (2 of 3) for Multiplatform Multilingual
CIK2JML	IBM WebSphere Application Server Network Deployment V8.5.5 (3 of 3) for Multiplatform Multilingual
CIK2GML	IBM Installation Manager V1.6.2 for Linux x86_64 (required to install WebSphere Application Server V8.5.5), Multilingual
CIL0DML	IBM Installation Manager V1.6.2 for Windows x86_64 (required to install WebSphere Application Server V8.5.5), Multilingual
CIK1YML	IBM Installation Manager V1.6.2 (required Install WebSphere Application Server V8.5.5) for AIX PowerPC Multilingual
CIK23ML	IBM Installation Manager V1.6.2 (required Install WebSphere Application Server V8.5.5) for IBM i Multilingual
CIK2LML	IBM DMZ Secure Proxy Server (1 of 2) (WebSphere Application Server Network Deployment V8.5.5) Multiplatform Multilingual
CIK2MML	IBM DMZ Secure Proxy Server (2 of 2) (WebSphere Application Server Network Deployment V8.5.5) Multiplatform Multilingual
CI6TQML	IBM® DB2® 10.1 - Limited Use for Linux® on AMD64 and Intel® EM64T
CI6TTML	IBM® DB2® 10.1 - Limited Use for 64-bit AIX®
CI6TYML	IBM® DB2® 10.1 - Limited Use for Windows® on AMD64 and Intel® EM64T systems (x64)
CI71VML	IBM® DB2® 10.1, National Language Pack for Linux® on AMD64 and Intel® EM64T systems (x64)
CI71QML	IBM® DB2® 10.1, National Language Pack for AIX®

Table 23. List of eAssembly images for IBM Sametime Limited Use 9.0 (continued)

Part number	Description
CI6VIML	IBM® DB2® Net Search Extender for Windows® on AMD64 and Intel® EM64T systems (x64)

Table 24. List of eAssembly images for IBM Connections Content Manager 4.5

Part number	Description
CIHC4ML	IBM Connections V4.5 Quick Start Guide for AIX, Windows, Linux, IBMi Multilingual
CIQ52ML	IBM FileNet Content Engine V5.2 AIX Multilingual
CIQ55ML	IBM FileNet Content Engine V5.2 Linux Multilingual
CIQ57ML	IBM FileNet Content Engine V5.2 Windows Multilingual
CIQ58ML	IBM FileNet Content Engine V5.2 Linux on z Systems Multilingual
CIQ5DEN	IBM FileNet Content Engine Client V5.2 Linux English
CIQ5FEN	IBM FileNet Content Engine Client V5.2 Windows English
CIQ5GEN	IBM FileNet Content Engine Client V5.2 Linux on z Systems English
CIQ5AEN	IBM FileNet Collaboration Services V2.0 Win
CIG7ZML	IBM FileNet Collaboration Services V2.0 Win
CIG80ML	IBM FileNet Collaboration Services V2.0 AIX
CIG84ML	IBM FileNet Collaboration Services V2.0 Linux
CIG8CML	IBM FileNet Collaboration Services V2.0 Linux on z Systems

Table 25. List of eAssembly images for IBM Sametime Limited Use 9.0

Part number	Description
CIQ43ML	IBM Sametime V9.0 Multiplatform Multilingual Quickstart Guide
CITN1ML	IBM Sametime Connect Client Limited Use V9.0 Windows, x86 Linux, Mac Multilingual
CITN2ML	IBM Sametime Community Server Limited Use V9.0 Windows Multilingual
CITN3ML	IBM Sametime Community Server Limited Use V9.0 AIX, Linux Multilingual
CITN4ML	IBM Sametime Community Server Limited Use V9.0 IBM i Multilingual
CIQ48ML	IBM Sametime Proxy Server V9.0 Windows, x86 Linux, AIX, IBMi Multilingual
CIQ4CML	IBM Sametime System Console Server V9.0 Windows, x86 Linux, AIX, IBMi Multilingual
CIQ6JML	IBM WebSphere V8.5.5.0 iFixes for IBM Sametime V9.0 Windows, x86 Linux, AIX, IBMi Multilingual
CIB8QML	Quick Start Guide - IBM Notes and Domino V9.0 Multilingual
CIB8REN	IBM Notes and Domino V9.0 Release Notes English
CIB9AEN	IBM Notes, Domino Designer and Admin V9.0 for Windows XP,Vista and Windows 7 32 Bit English
CIBM0EN	IBM Domino Server V9.0 32 Bit for Windows English
CIBM3EN	IBM Domino Server V9.0 32 bit for AIX English
CIBM4EN	IBM Domino Server V9.0 32 bit for Linux for xSeries English
CIBM7EN	IBM Domino Server V9.0 for IBM i English
CINK4CS	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Czech
CINK5HU	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Hungarian
CINK6RU	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Russian
CINK7EL	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Greek
CINK8TR	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Turkish
CINK9AR	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Arabic
CINL0PT	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Portuguese
CINL1PL	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Polish
CINL2TH	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Thai
CINL3SI	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Slovenian
CINL4SK	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Slovakian
CILI1FI	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Finnish
CILI2NO	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Norwegian
CILI3SV	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Swedish
CILI4DA	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Danish

Table 25. List of eAssembly images for IBM Sametime Limited Use 9.0 (continued)

Part number	Description
CILI5NL	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Dutch
CIJ7CCA	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Catalan
CIJ7DDE	IBM Notes, Domino Designer and Admin 9.0 for Windows XP, Windows 7 and Windows 8 German
CIJ7EKO	IBM Notes, Domino Designer and Admin 9.0 for Windows XP, Windows 7 and Windows 8 Korean
CIJ7FSC	IBM Notes, Domino Designer and Admin 9.0 for Windows XP, Windows 7 and Windows 8 Simplified Chinese
CIJ7GES	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Spanish
CIJ7HFR	IBM Notes, Domino Designer and Admin 9.0 for Windows XP, Windows 7 and Windows 8 French
CIJ7ITC	IBM Notes, Domino Designer and Admin 9.0 for Windows XP, Windows 7 and Windows 8 Traditional Chinese
CIJ7JJA	IBM Notes, Domino Designer and Admin 9.0 for Windows XP, Windows 7 and Windows 8 Japanese
CIJ7KIT	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Italian
CIJ7LBP	IBM Notes and Domino Designer 9.0 for Windows XP, Windows 7 and Windows 8 Brazilian Portuguese
CILS9FI	IBM Domino 9.0 Language Pack for Windows 2008/2012 and IBM Power Systems (Multi O/S Install)Finnish
CILT4FI	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install Finnish
CILT0NO	IBM Domino 9.0 Language Pack for Windows 2008/2012 and IBM Power Systems (Multi O/S Install)Norwegian
CILT5NO	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install Norwegian
CILT1SV	IBM Domino 9.0 Language Pack for Windows 2008/2012 and IBM Power Systems (Multi O/S Install)Swedish
CILT6SV	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install Swedish
CILT2DA	IBM Domino 9.0 Language Pack for Windows 2008/2012 and IBM Power Systems (Multi O/S Install)Danish
CILT7DA	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install Danish
CILT3NL	IBM Domino 9.0 Language Pack for Windows 2008/2012 and IBM Power Systems (Multi O/S Install)Dutch
CIJ9RJA	IBM Domino Server V9.0 Language Pack for Windows 2008, Windows 2012 and IBM Power Systems Multi O/S Install Japanese
CIJA9JA	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install Japanese
CIJ9TEN	IBM Domino Server V9.0 Language Pack for Windows 2008, Windows 2012 and IBM Power Systems Multi O/S Install English
CIJB1EN	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install English
CIJ9LKO	IBM Domino Server V9.0 Language Pack for Windows 2008, Windows 2012 and IBM Power Systems Multi O/S Install Korean
CIJA4KO	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install Korean
CINQ9CS	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Czech
CINS0CS	IBM Domino Server V9.0 Language Pack for Linux on xSeries Czech
CINS7TH	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Thai
CINS9TH	IBM Domino Server V9.0 Language Pack for AIX Thai
CIJ9MSC	IBM Domino Server V9.0 Language Pack for Windows 2008, Windows 2012 and IBM Power Systems Multi O/S Install Simplified Chinese
CIJA5SC	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install Simplified Chinese
CIJ9QTC	IBM Domino Server V9.0 Language Pack for Windows 2008, Windows 2012 and IBM Power Systems Multi O/S Install Traditional Chinese
CIJA8TC	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install Traditional Chinese

Table 25. List of eAssembly images for IBM Sametime Limited Use 9.0 (continued)

Part number	Description
CIJ9JCA	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Catalan
CIJ9KDE	IBM Domino Server V9.0 Language Pack for Windows 2008, Windows 2012 and IBM Power Systems Multi O/S Install German
CIJA3DE	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install German
CIJ9SIT	IBM Domino Server V9.0 Language Pack for Windows 2008, Windows 2012 and IBM Power Systems Multi O/S Install Italian
CIJB0IT	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install Italian
CIJ9PFR	IBM Domino Server V9.0 Language Pack for Windows 2008, Windows 2012 and IBM Power Systems Multi O/S Install French
CIJA7FR	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install French
CIJ9NES	IBM Domino Server V9.0 Language Pack for Windows 2008, Windows 2012 and IBM Power Systems Multi O/S Install Spanish
CIJA6ES	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install Spanish
CIJ9UBP	IBM Domino Server V9.0 Language Pack for Windows 2008, Windows 2012 and IBM Power Systems Multi O/S Install Brazilian Portuguese
CIJB2BP	IBM Domino Server V9.0 Language Pack for AIX, Linux on xSeries and Linux on zSeries Multi O/S Install Brazilian Portuguese
CINQ8AR	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Arabic
CINR9AR	IBM Domino Server V9.0 Language Pack for Linux on xSeries Arabic
CINR0EL	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Greek
CINS1EL	IBM Domino Server V9.0 Language Pack for Linux on xSeries Greek
CINR1IW	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Hebrew
CINS2IW	IBM Domino Server V9.0 Language Pack for Linux on xSeries Hebrew
CINR2HU	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Hungarian
CINT0HU	IBM Domino Server V9.0 Language Pack for AIX and Linux on xSeries Hungarian
CINR7SK	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Slovakian
CINR8SI	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Slovenian
CINS8KZ	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Kazakh
CINR3PT	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Portuguese
CINR4PL	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Polish
CINT1PL	IBM Domino Server V9.0 Language Pack for AIX and Linux on xSeries Polish
CINR5RU	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Russian
CINS4RU	IBM Domino Server V9.0 Language Pack for Linux on xSeries Russian
CINR6TR	IBM Domino Server V9.0 Language Pack for Windows 2008 and Windows 2012 Multi O/S Install Turkish
CINS5TR	IBM Domino Server V9.0 Language Pack for Linux on xSeries Turkish
CINS3PT	IBM Domino Server V9.0 Language Pack for Linux on xSeries Portuguese
CIK2NML	IBM WebSphere Edge Components: Load Balancer for IPv4 and IPv6 (for WebSphere Application Server Network Deployment V8.5.5) Multiplatform Multilingual
CIL5DML	IBM WebSphere Edge Components: Load Balancer for IPv4 (for WebSphere Application Server Network Deployment V8.5.5) Multiplatform, Multilingual
CIL5EML	IBM WebSphere Edge Components: Caching Proxy (for WebSphere Application Server Network Deployment V8.5.5)Multiplatform, Multilingual

Table 25. List of eAssembly images for IBM Sametime Limited Use 9.0 (continued)

Part number	Description
CIK2HML	IBM WebSphere Application Server Network Deployment V8.5.5 (1 of 3) for Multiplatform Multilingual
CIK2IML	IBM WebSphere Application Server Network Deployment V8.5.5 (2 of 3) for Multiplatform Multilingual
CIK2JML	IBM WebSphere Application Server Network Deployment V8.5.5 (3 of 3) for Multiplatform Multilingual
CIK2GML	IBM Installation Manager V1.6.2 for Linux x86_64 (required to install WebSphere Application Server V8.5.5), Multilingual
CIL0DML	IBM Installation Manager V1.6.2 for Windows x86_64 (required to install WebSphere Application Server V8.5.5), Multilingual
CIK1YML	IBM Installation Manager V1.6.2 (required Install WebSphere Application Server V8.5.5) for AIX PowerPC Multilingual
CIK23ML	IBM Installation Manager V1.6.2 (required Install WebSphere Application Server V8.5.5) for IBM i Multilingual
CIK2LML	IBM DMZ Secure Proxy Server (1 of 2) (WebSphere Application Server Network Deployment V8.5.5) Multiplatform Multilingual
CIK2MML	IBM DMZ Secure Proxy Server (2 of 2) (WebSphere Application Server Network Deployment V8.5.5) Multiplatform Multilingual
CI6TQML	IBM® DB2® 10.1 - Limited Use for Linux® on AMD64 and Intel® EM64T
CI6TTML	IBM® DB2® 10.1 - Limited Use for 64-bit AIX®
CI6TYML	IBM® DB2® 10.1 - Limited Use for Windows® on AMD64 and Intel® EM64T systems (x64)
CI71VML	IBM® DB2® 10.1, National Language Pack for Linux® on AMD64 and Intel® EM64T systems (x64)
CI71QML	IBM® DB2® 10.1, National Language Pack for AIX®
CI6VIML	IBM® DB2® Net Search Extender for Windows® on AMD64 and Intel® EM64T systems (x64)

Related information:

WebSphere Portal detailed system requirements

IBM Passport Advantage

Installing and preparing the prerequisite software

Before you install the digital experience software, make sure that the prerequisite software is installed and configured. Depending on your environment, you might already have the prerequisites, such as a database server and user registry. Verify that the prerequisite software is the correct version, has the required fix packs applied, and is configured to work with the digital experience software.

"Installing and preparing the database software" on page 149
You might already have a database software installed. You still need to make sure that the installed database software is a supported version and has the required fix packs applied. In addition, you might need to configure the supported database software to work with the digital experience software.

"Preparing the user registry software" on page 158
You probably already have a user registry that is deployed and configured for your network. Before you install and deploy the digital experience software, make sure that your user registry software is a support version. In addition, you might need to configure the user registry to work with the digital experience software.

"Preparing a remote web server" on page 166
Install and configure the web server plug-in. The IBM WebSphere Application Server provides the plug-in. Configure the web server to communicate with IBM WebSphere Portal Express.

Related tasks:

"Planning to install WebSphere Portal Express" on page 101
Before you install IBM WebSphere Portal Express in a production environment, you need to assess your hardware and software needs, possible database configurations, security options, and LDAP server options. Skipping this important step can lead to unexpected results and costly delays.

Installing and preparing the database software

You might already have a database software installed. You still need to make sure that the installed database software is a supported version and has the required fix packs applied. In addition, you might need to configure the supported database software to work with the digital experience software.

"Installing and preparing DB2"
Use this information to install DB2 or DB2 pureScale for use with IBM WebSphere Portal Express.

"Installing and preparing Oracle or Oracle RAC" on page 151
You can use Oracle or Oracle RAC as the database software.

"Installing and preparing SQL Server" on page 152
Use this information to install and configure SQL Server for use with WebSphere Portal Express.

"Installing and preparing DB2 for z/OS" on page 154
Use this information to install DB2 for z/OS for use with IBM WebSphere Portal Express.

"Preparing IBM DB2 for i" on page 157
DB2 is integrated with IBM i. However, the databases and users that are required for WebSphere Portal Express must be created.

Installing and preparing DB2

Use this information to install DB2 or DB2 pureScale for use with IBM WebSphere Portal Express.

Before you begin

- Review the database considerations.
- Ensure the database that you plan to use is supported by this version of WebSphere Portal Express. Refer to the list of supported databases in the WebSphere Portal Express detailed system requirements.
- Set up the operating system with updated kernel parameters according to the DB2 Quick Beginnings guide at DB2 Technical Support.
- When you install DB2 with the DB2 installation program, it automatically creates a DB2 administrative user with the correct operating system rights.
- Ensure that you have enough disk space for the DB2 instance home directory to be able to create the required databases.
- A DB2 instance supports a limited number (NUMDB) of concurrently active databases. Increase this value if the DB2 instance maintains databases for WebSphere Portal Express and other applications. The NUMDB value depends on how many databases are concurrently used on the DB2 instance. Examples of concurrent usage include two portals that access the same DB2 instance or transferring data from multiple portals that use the same DB2 instance. To change the default to 30, enter the following command at the database prompt: `UPDATE DATABASE MANAGER CONFIGURATION USING NUMDB 30`. A message displays that confirms a successful completion of the update.
- WebSphere Portal Express supports DB2 JDBC Type 2 (CLI-based) and Type 4 (JCC) drivers.

About this task

All DB2 instructions apply to DB2 pureScale except where specifically noted.

Procedure

1. To install DB2 or the DB2 client and the required fix pack, follow the instructions that are provided with the DB2 documentation.
2. If DB2 is installed on another system than WebSphere Portal Express, copy the driver JAR files from the DB2 server to the Portal server. The typical location for these files on the DB2 server is in the `db2_home/java` directory. Place these driver files within the *wp_profile_root* directory, for example:

   wp_profile_root`/PortalServer/dbdrivers/db2jcc4.jar`

   wp_profile_root`/PortalServer/dbdrivers/db2jcc_license_cu.jar`
3. Ensure that the DB2 instance port was added to the services file during the DB2 installation.

   Linux :

   a. Get the value for the TCP/IP service name (SVCENAME). Open a shell and log in as the instance owner. Enter the following command:

      ```
      db2 "get dbm cfg"|grep (SVCENAME)
      ```

      A typical value for the SVCENAME is `(SVCENAME) = db2c_db2inst1`

   b. Get the DB2 port number by using the SVCENAME value. Enter the following command:

      ```
      echo /etc/services | grep your_SVCENAME_value
      ```

      A typical output for the port number:

      your_SVCENAME_value 50000`/tcp`

      In this example, 50000 is the port number.

   Windows:

   a. Get the value for the TCP/IP service name (SVCENAME). Open a DB2 Command Window and enter the following command:

      ```
      db2 get dbm cfg | findstr (SVCENAME)
      ```

      A typical value for the SVCENAME is `(SVCENAME) = db2c_DB2`

   b. Get the DB2 port number by using the SVCENAME value. Enter the following command:

      ```
      type %SystemRoot%\system32\drivers\etc\services | findstr db2c_DB2
      ```

      A typical output for the port number:

      your_SVCENAME_value 50000`/tcp`

      In this example, 50000 is the port number.

What to do next

Use the Configuration Wizard to set up and configure the database to work with WebSphere Portal Express. You can use the wizard to create custom scripts that you or your database administrator can use to configure the database. You can also use the wizard to automatically set up and configure the database. The wizard creates instructions and scripts that are based on your selections and provided data.

When you use the wizard and provide information about the database for your environment, be aware of the following considerations:

- The value for the database name, database server node, or schema name must be unique.
- When the DB2 Universal JDBC driver (type 4 mode) is used, connect to the database directly. Do not connect to an alias database (gateway), instead specify the real database name in the JDBC connection URL (*dbdomain*.DbUrl) and in the database name property (*dbdomain*.DbName).
- The Configuration Wizard's default configuration uses the instance (*db2inst1*) that is created by the installation program. Using separate databases can improve scalability and performance. The wizard allows you to change the default behavior. You can select whether each of the portal database domains exist in one or many instances.
- Your database name (*dbdomain*.DbName) cannot exceed eight characters.
- You cannot use the **Database Transfer** option in the configuration wizard to assign custom table spaces on your database server. You can perform manual steps to assign custom table spaces. Go to "Assigning custom table spaces" on page 555 for more information.

(Optional) After you transfer your data to DB2, run a configuration task to enable support for high availability recovery (HADR) and rollforward recovery.

Related tasks:

"DB2: Enabling support for high availability recovery and rollforward recovery " on page 560
Optional: To prevent data loss on DB2, modify the JCR schema to support High Availability Disaster Recovery and rollforward recovery.

Installing and preparing Oracle or Oracle RAC

You can use Oracle or Oracle RAC as the database software.

Before you begin

- Review the database considerations.
- Ensure the database that you plan to use is supported by this version of WebSphere Portal Express. Refer to the list of supported databases in the WebSphere Portal Express detailed system requirements.
- For Linux and Oracle OCI Type-2 JDBC driver: The Oracle OCI Type-2 JDBC driver with the full oracle client works with WebSphere Portal Express. However, to successfully complete the database transfer task, the Oracle Thin Type-4 JDBC is required for Linux. After successfully completing the database transfer task using the thin driver, if needed, you can specify and then return to using the Oracle OCI Type-2 full client.

Procedure

1. Install the Oracle client and any required fix packs. Follow the instructions that are provided with the Oracle documentation.
2. Install Oracle JDBC Type 4 drivers or Oracle JDBC OCI Type 2 drivers as appropriate.

   Refer to the Oracle or Oracle RAC product documentation for installation instructions.

Oracle RAC

3. Ensure the Oracle CRS (Cluster Ready Service) and Oracle RAC databases are installed and configured on the primary and secondary nodes.
4. Start global services daemon (GSD), oracle listeners, and agents in both RAC nodes. Run the following commands:

```
$ gsdctl start
$ lsnrctl start
$ agentctl start
```

5. The default tablespace size for Oracle RAC may need to be set to 1024MB with `autoextend` turned on for database transfer to be successful.

Update environment variables

6. `LIBPATH=/u01/app/oracle/product/11.2.0/client_1;export LIBPATH`
7. JDBC OCI (Linux and Solaris): Set the environment variable on the IBM WebSphere Portal Express server to point to the directory where you downloaded and extracted the compressed Oracle client files.

   `LD_LIBRARY_PATH=/u01/app/oracle/product/11.2.0/client_1;export LD_LIBRARY_PATH`
8. Restart server1 to ensure that the Configuration Wizard uses the updated environment variables. Go to *AppServer_home*`/profiles/cw_profile/bin` and stop the server:
   - Linux : `./stopServer.sh server1 -username` *username* `-password` *password*
   - IBM i: `stopServer server1 -username` *username* `-password` *password*
   - Windows: `stopServer.bat server1 -username` *username* `-password` *password*

   Then, start the server:
   - Linux : `./startServer.sh server1`
   - IBM i: `startServer server1`
   - Windows: `startServer.bat server1`

What to do next

Use the Configuration Wizard to set up and configure the database to work with WebSphere Portal Express. You can use the wizard to create custom scripts that you or your database administrator can use to configure the database. You can also use the wizard to automatically set up and configure the database. The wizard creates instructions and scripts that are based on your selections and provided data.

Important: To successfully complete the database transfer task, the Oracle Thin Type-4 JDBC is required for Linux.

When you use the wizard, you provide information about the database for your environment.

Note: Before you enter your database name (*dbdomain*`.DbName`) in the Configuration Wizard, check your database documentation for restrictions on character length.

You cannot use the **Database Transfer** option in the configuration wizard to assign custom table spaces on your database server. You can perform manual steps to assign custom table spaces. Go to "Assigning custom table spaces" on page 555 for more information.

Installing and preparing SQL Server

Use this information to install and configure SQL Server for use with WebSphere Portal Express.

Before you begin

- Review the database considerations.

- Ensure the database that you plan to use is supported by this version of WebSphere Portal Express. Refer to the list of supported databases in the WebSphere Portal Express detailed system requirements.
- Download a JDBC 4.0 compliant driver from Microsoft

Procedure

1. Install SQL Server and all required patches.
2. In the SQL Server Setup, choose the following:
   a. In the Setup Role panel, choose **SQL Server Feature Installation**.
   b. In the Features Selection panel, choose at least **Database Engine Services**.
   c. In the Database Engine Configuration, select **Mixed Mode (SQL Server Authentication and Windows authentication)**.

      Important:
      Mixed Mode authentication allows either a Windows user or an SQL Server user to log in to the SQL Server. However, WebSphere Portal Express requires the user to be an SQL Server user.
3. Complete the installation. Use the SQL Server documentation as a guide.
4. Install the JDBC driver. Use Microsoft SQL Server JDBC drivers and enable XA connections:
   a. Download and install the Microsoft SQL Server JDBC driver; see Microsoft Download Center for information.
   b. Copy the `sqljdbc_xa.dll` file from the `xa` subdirectory to the following directory of the SQL Server installation:
      - Microsoft SQL Server 2012: `C:\Program Files\Microsoft SQL Server\MSSQL11.MSSQLSERVER\MSSQL\Binn`
      - CF05 Microsoft SQL Server 2014: `C:\Program Files\Microsoft SQL Server\MSSQL12.MSSQLSERVER\MSSQL\Binn`
   c. Start the database server.
   d. Ensure that the Distributed Transaction Coordinator is started. Verify this status in the list of services in the Computer Management console.
   e. Start the Microsoft SQL Server Management Studio and connect to the local database engine as the system administrator, `sa`.
   f. Select **File** > **Open** > **File** and select `xa_install.sql` from the subdirectory of the downloaded and extracted JDBC driver.
   g. Run the script by Select **Query** > **Execute** to run the script.

      Note: Ignore any warnings that say that stored procedures cannot be found.
   h. To grant permission to a specific user to participate in distributed transactions with the JDBC driver, add the user to the SqlJDBCXAUser role.
5. Complete the following steps to enable XA Transactions in Windows Component Services:
   a. Choose **Component Services** of the **Administrative Tools**.
   b. Expand the tree view to locate the computer where you want to turn on support for XA transactions. For example, My Computer.
   c. Display the menu for the computer name and click **Properties**.
   d. Click **Options** and tune the Transaction Timeout. The minimum requirement is 180 seconds.
   e. Click **OK** to save your changes.

f. Expand **Component Services,** > **Computers,** > **My Computer,** > **Distributed Transaction Coordinator.**

g. Right-click **Local DTC** and then select **Properties**.

h. Click the **Security tab** on the Local DTC Properties dialog box.

i. Select the **Enable XA Transactions** check box, and then click **OK**. This action causes a MS DTC service restart.

6. Start SQL Server.
7. Connect at least one user to the SQL Server instance. A user can be granted permission to use several schema names, so a single user for each instance is sufficient.

What to do next

Use the Configuration Wizard to set up and configure the database to work with WebSphere Portal Express. You can use the wizard to create custom scripts that you or your database administrator can use to configure the database. You can also use the wizard to automatically set up and configure the database. The wizard creates instructions and scripts that are based on your selections and provided data.

When you use the wizard, you provide information about the database for your environment.

Note: Before you enter your database name (*dbdomain*.DbName) in the Configuration Wizard, check your database documentation for restrictions on character length.

You cannot use the **Database Transfer** option in the configuration wizard to assign custom table spaces on your database server. You can perform manual steps to assign custom table spaces. Go to "Assigning custom table spaces" on page 555 for more information.

Installing and preparing DB2 for z/OS

Use this information to install DB2 for z/OS for use with IBM WebSphere Portal Express.

Before you begin

- Review the database considerations.
- Ensure the database that you plan to use is supported by this version of WebSphere Portal Express. Refer to the list of supported databases in the WebSphere Portal Express detailed system requirements.
- The DB2 subsystem must be on a supported z/OS operating system.
- Ensure that Java Database Connectivity requirements are met. Consult the following references:
  - *DB2 Universal Database for OS/390 and z/OS: Application Programming Guide and Reference for Java*
  - The IBM Redbooks publication, DB2 for z/OS and OS/390: Ready for Java SG24-6435-00.
- If the current version of WebSphere Portal Express and an earlier version coexist with the same DB2 for z/OS subsystem, the database user IDs for the current version must be different from the earlier version to avoid conflicts during

installation. If the two versions of WebSphere Portal Express connect to two different DB2 for z/OS subsystems, using the same user ID does not cause conflict.

Procedure

1. To use DB2 for z/OS as the database software for WebSphere Portal Express, you must have DB2 for z/OS installed on your z/OS system. Refer to the DB2 for z/OS product documentation for instructions.
2. Read the Planning for DB2 for z/OS topic before you configure DB2 for z/OS as the WebSphere Portal Express database.
3. Check the buffer pool allocations for your system and define the buffer pools as appropriate for your installation and define a large enough size.

   ```
   -db2 display bufferpool(bp2)
   -db2 alter bufferpool(bp2) vpsize(15000)
   ```

   a. Repeat for extra buffer pools as needed. For example:

      bp3

      bp4

      bp5

      bp32k1

      bp32k2
4. Update the BP8K0 catalog buffer pool to 35,000 buffers before you transfer the database. Change this value according to your environment. The SYSIBM.SYSDATABASE table is in this buffer pool and is used extensively by DB2 for z/OS for the database transfer.
5. Change the Common Service Area (CSA) setting to `3500,350000`. Read the appropriate DB2 for z/OS topic for complete information about calculating and setting CSA:
   - DB2 for z/OS Version 10.1, Common service area
   - DB2 for z/OS Version 11.1, Common service area storage requirements
6. During database transfer from Derby to DB2 for z/OS, a supporting low-order byte table space is created for the database tables that store documents. The PRIQTY and SECQTY for the table space are assigned with the default values. If you plan to store many documents, use an automatic class selection (ACS) routine to allocate the DB2 for z/OS data sets with a primary and secondary space allocation of at least 10 cylinders. Or, specify a large enough value for **PRIQTY** and **SEQTY** in the DB2 DSNTIJUZ member. The table spaces can be identified by their name, having a structure like JCRDB.S*xxxxxxx*, where *xxxxxxx* is a system-assigned combination of seven numbers and characters.

   a. Also, in member DSNTIJUZ, update the following parameters and then verify DSNTIJUZ runs successfully.

      edmdbdc = *204800*

      edmpool=*409600*

      edmstmtc=*204800*

      rrulock=*no*

      cachedyn=*yes* (prepared, dynamic SQL statements are cached)

      dbacrvw=*yes* (to allow database administrators to create Views)

      +++tbsbpxml=BP16K1 (to explicitly create an XML table space. This value can be changed to any valid 16 K buffer pool where the Administrative User has the USE privilege.)

7. Ensure that the job DSNTIJSG ran to create the objects that are needed for the DB2 JDBC and ODBC metadata methods. See the DB2 Installation Guide Enabling stored procedures and tables for JDBC and ODBC support.
8. Ensure that job DSNTIJMS runs successfully (re-execute binds).
9. Ensure that job DSNTIJEX runs successfully.
10. Because large objects are stored in columns that can become large, logging changes to these columns requires a huge amount of log space. For this reason, large object (LOB) logging is disabled by default for table spaces that contain such data. With LOB logging disabled, you can recover full backups, but not incremental backups that can be used for point in time recovery. To recover point in time backups, you must enable LOB logging. For detailed instructions, see technote 1306637, *Managing LOB logging in DB2 for z/OS*.

LikeMinds

11. If you intend to run the LikeMinds sample, increase the **NUMLKTS** and **NUMLKUS** parameters: Ten times the default is sufficient, more depending on your usage of the sample. For example, if **NUMLKTS**=*1000* and **NUMLKUS**=*10000* are the installation default values, then update these values to **NUMLKTS**=*10000* and **NUMLKUS**=*100000*.
12.

Type 2 driver configuration

13. If you are planning to use Type 2 driver with DB2 for z/OS Version 9.1.2, ensure that DB2 for z/OS APAR PK58105 is installed.
14. If you are using the older DB2 Type 2 JDBC driver, enable extended shared memory usage with the following commands:

    ```
    export EXTSHM=ON
    db2set DB2ENVLIST=EXTSHM
    db2start
    ```

    For permanent changes, add the environment variable to the `profile.env` file:

    ```
    DB2ENVLIST='EXTSHM'
    ```

    in `/home/db2inst/sqllib/userprofile` add:

    ```
    export EXTSHM=ON
    ```

    Note: The shell must be reopened before you restart DB2 for z/OS.
15. If you are using the older DB2 Type 2 JDBC driver, configure your DB2 Connect client with the following commands:

    ```
    db2 update dbm cfg using tp_mon_name WAS
    db2 update dbm cfg using spm_name hostname
    ```

    where *hostname* is the host name for the server where the DB2 Connect client is installed (same as portal server).\
16. If you are using the older DB2 Type 2 JDBC driver, install the DB2 Connect client on the WebSphere Portal Express server to connect to the remote database.

What to do next

Use the Configuration Wizard to set up and configure the database to work with WebSphere Portal Express. You can use the wizard to create custom scripts that you or your database administrator can use to configure the database. You can also use the wizard to automatically set up and configure the database. The wizard creates instructions and scripts that are based on your selections and provided data.

When you use the Configuration wizard and provide information about your database, consider the following points:

•
•
•

- Before you enter your database name (*dbdomain*.DbName) in the Configuration Wizard, check your database documentation for restrictions on character length.
- You cannot use the **Database Transfer** option in the configuration wizard to assign custom table spaces on your database server. You can perform manual steps to assign custom table spaces. Go to "Assigning custom table spaces" on page 555 for more information.

Preparing IBM DB2 for i

DB2 is integrated with IBM i. However, the databases and users that are required for WebSphere Portal Express must be created.

Before you begin

- Ensure the database that you plan to use is supported by this version of WebSphere Portal Express. Refer to the list of supported databases in the WebSphere Portal Express detailed system requirements.

What to do next

Use the Configuration Wizard to set up and configure the database to work with WebSphere Portal Express. You can use the wizard to create custom scripts that you or your database administrator can use to configure the database. You can also use the wizard to automatically set up and configure the database. The wizard creates instructions and scripts that are based on your selections and provided data.

When you use the wizard and provide information about the database for your environment, be aware of the following considerations:

- If you choose to use one database to store all WebSphere Portal Express, Member Manager, and content publishing information, only one user profile is required. Additional user profiles are necessary only if using multiple IBM i systems or separate databases are required.
- Before you enter your database name (*dbdomain*.DbName) in the Configuration Wizard, check your database documentation for restrictions on character length.
- The Configuration Wizard uses JDBC type 4 drivers by default. The examples in the Configuration Wizard apply only to JDBC type 4 drivers. You can change the default selection in the Configuration Wizard to JDBC type 2 during step 2, Customize Values. If you select JDBC type 2, apply these examples to your configuration:

Table 26. Example values for JDBC type 2 driver

Field label in Configuration Wizard	Example value
Database name	*LOCAL/wpsdb
Database URL	jdbc:db2:*LOCAL/wpsdb
DB2 for IBM i library	/QIBM/ProdData/OS400/Java400/ext/db2_classes16.jar
DB2 for IBM i driver class name	com.ibm.db2.jdbc.app.DB2Driver

Preparing the user registry software

You probably already have a user registry that is deployed and configured for your network. Before you install and deploy the digital experience software, make sure that your user registry software is a support version. In addition, you might need to configure the user registry to work with the digital experience software.

"Preparing an Active Directory-Lightweight-Directory-Services on Windows"
If you plan to use an Active Directory-Lightweight-Directory-Services as an LDAP user registry, you must install and set up the server so that it communicates with IBM WebSphere Portal Express.

"Preparing a Domino Directory server" on page 159
If you plan to use a Domino Directory as an LDAP user registry, you must install and set up the server so that it communicates with IBM WebSphere Portal Express.

"Preparing a IBM Directory Server" on page 161
If you plan to use a IBM Directory Server as an LDAP user registry, you must install and set up the server so that it communicates with IBM WebSphere Portal Express.

"Preparing an Active Directory server" on page 162
If you plan to use Active Directory as an LDAP user registry, you must install and set up the server so that it can communicate with IBM WebSphere Portal Express.

"Preparing a Novell eDirectory" on page 163
If you plan to use a Novell eDirectory as an LDAP user registry, you must install and set up the server so that it communicates with IBM WebSphere Portal Express.

"Preparing a SecureWay Security Server" on page 164
If you plan to use a SecureWay Security Server as an LDAP user registry, you must install and set up the server so that it communicates with IBM WebSphere Portal Express.

"Preparing an Oracle Directory Server" on page 165
If you plan to use an Oracle Directory Server as an LDAP user registry, you must install and set up the server so that it communicates with IBM WebSphere Portal Express. Oracle Directory Server was formerly known as Sun Java System Directory Server.

Preparing an Active Directory-Lightweight-Directory-Services on Windows

If you plan to use an Active Directory-Lightweight-Directory-Services as an LDAP user registry, you must install and set up the server so that it communicates with IBM WebSphere Portal Express.

Procedure

1. Download and install Active Directory Application Mode.
2. Complete the following steps as a guide to create the WebSphere Portal Express administrative user:
   a. Create a user with the Windows administrative tools.

      Note: There is a 20 character limitation for the user account name.
   b. Set the password for the new user.
   c. Activate the new user with the Windows administrative tools. Set the **msDS-UserAccountDisabled** attribute to `false`.

Preparing a Domino Directory server

If you plan to use a Domino Directory as an LDAP user registry, you must install and set up the server so that it communicates with IBM WebSphere Portal Express.

Procedure

1. Complete the following steps to install the Domino Directory:
   a. Go to Domino documentation.
   b. Select the appropriate version tab for your product.
   c. Click the **Download/View online** link for the Lotus Domino Information Center.
   d. Click **Domino Administrator Help** > **Installation** > **Installing and setting up Domino servers** > **Server installation** > **Installing Domino** and complete this task.
   e. Click **Domino Administrator Help** > **Installation** > **Installing and setting up Domino servers** > **The Domino server setup program** and complete this task.
2. Complete the following steps to install the Domino Directory:
   a. Go to Domino documentation.
   b. Select the appropriate version tab for your product.
   c. Click the **Download/View online** link for Installing and Managing Domino for System i.
   d. Complete the tasks under **Chapter 3 Installing Domino on your system**.
   e. Complete the tasks under **Chapter 6 Setting up a First Domino server**.
   f. Complete the tasks under **Chapter 8 Setting up an Additional Domino server**.
3. Complete the following steps as a guide to create the WebSphere Portal Express administrative user:
   a. Go to the **People** view of the Domino Directory and then click **Add Person**.
   b. Enter the following values in the **New Person** form to create the LDAP bind user. The following example uses `wpsbind` to represent the LDAP bind user:

      Last Name
      `wpsbind`

      User name
      `wpsbind/`*`DominoDomain`*, where *DominoDomain* is your Domino Internet domain.

      `wpsbind`

      Note: Make sure that you enter two values in the **User Name** field, where the first value includes the Domino domain.

      Short name/UserID
      `wpsbind`

      Internet password
      `wpsbind`
   c. Click **Save and Close** to save the new person record for `wpsbind` and return to the **People** view.
   d. Click **Add Person** and enter the following values in the **New Person** form to create the Portal administration user. The following example uses `wpsadmin` to represent the Portal administration user:

Last Name
wpsadmin, where wpsadmin is the user ID for the WebSphere Portal Express Administrator.

User name
wpsadmin/*DominoDomain*, where *DominoDomain* is your Domino Internet domain.

```
wpsadmin
```

Note: Make sure that you enter two values in the **User Name** field, where the first value includes the Domino domain.

Short name/UserID
wpsadmin

Internet password
wpsadmin

e. Click **Save and Close** to save the new person record for wpsadmin and return to the **People** view.

f. Go to the **Groups** view and click **Add Group**.

g. Enter the following values in the **New Group** form on the Basic tab:

Group name
wpsadmins

Note: If your Domino LDAP shares a realm with another user registry, you must use the hierarchical naming convention for the group names. Enter wpsadmins/DominoDomain to avoid unexpected results during WebSphere Portal Express run time.

Group type
Multi-purpose

Members
wpsbind/DominoDomain

wpsadmin/DominoDomain

Note: You can add more administrator users.

h. Click **Save and Close** to save the wpsadmins group with the wpsbind and wpsadmin users as members.

4. Complete the following steps to update the access control list for the Domino Directory:

a. Open the names.nsf file in the Domino Administrator or Lotus Notes client.

b. Click **File** > **Application** > **Access Control** from the main menu to open the access control list for the file.

c. In the **Access Control List** > **Basics** panel, ensure that the wpsadmins group has either Author or Editor access.

d. Add the following **Role Types** to the wpsadmins group:

GroupCreator

GroupModifier

UserCreator

UserModifier

e. Click **OK**.

Preparing a IBM Directory Server

If you plan to use a IBM Directory Server as an LDAP user registry, you must install and set up the server so that it communicates with IBM WebSphere Portal Express.

Procedure

1. Install IBM Directory Server. Refer to the IBM Directory Server Installation and Configuration Guide for instructions.

   Restriction: Users or groups must not contain a Turkish uppercase dotted I or lowercase dotted i in the distinguished name. It prevents correct retrieval of that user or group.
2. Complete the following steps with the IBM Directory Server web administration tool to create the WebSphere Portal Express administrative user:
   a. Optional: Complete the following steps to create a directory suffix:
      1) Click the **Server Administration** folder in the directory server console navigation.
      2) Click the **Manage Server Properties** folder under the Server Administration folder and then select **Suffixes** on the main page.
      3) Type the **Base DN** name for the suffix; for example: `dc=yourcompany,dc=com`.
      4) Click **Add**.
      5) Click **OK** to save your changes.
   b. Open the appropriate LDIF file in the *PortalServer_root*`/installer/wp.iim/ldif` directory, with a text editor:

      Use the `PortalUsers.ldif` file as a working example and adapt appropriately to work with your LDAP server.

      Use the `ContentUsers.ldif` file for the IBM Content Manager group and user ID if you configured IBM Content Manager.
   c. Replace every `dc=yourco,dc=com` with your suffix.
   d. Replace any prefixes and suffixes that are unique to your LDAP server.
   e. You can specify user names other than `wpsadmin` and `wpsbind`. For security reasons, specify nontrivial passwords for these administrator accounts.
   f. Save your changes.
   g. Complete the instructions that are provided with your directory server to import the LDIF file.
3. Complete the following steps to create the WebSphere Portal Express administrative user:
   a. Open the appropriate LDIF file in the *PortalServer_root*`/installer/wp.iim/ldif` directory, with a text editor:

      Use the `PortalUsers.ldif` file as a working example and adapt appropriately to work with your LDAP server.

      Use the `ContentUsers.ldif` file for the IBM Content Manager group and user ID if you configured IBM Content Manager.
   b. Replace every `dc=yourco,dc=com` with your suffix.
   c. Replace any prefixes and suffixes that are unique to your LDAP server.
   d. You can specify user names other than `wpsadmin` and `wpsbind`. For security reasons, specify nontrivial passwords for these administrator accounts.

e. Optional: If you use IBM Security Access Manager Version 5.1, set the **objectclasses** to accessGroup. If you use Security Access Manager Version 6, set the **objectclasses** to groupOfNames.

f. Save your changes.

g. Complete the instructions that are provided with your directory server to import the LDIF file.

Preparing an Active Directory server

If you plan to use Active Directory as an LDAP user registry, you must install and set up the server so that it can communicate with IBM WebSphere Portal Express.

Procedure

1. Complete the following steps to install and configure Active Directory:
   a. Install Windows Server version 2008 or 2012, which includes Active Directory. Refer to http://www.microsoft.com/windows2000/technologies/directory/ad/default.asp for information.
   b. Install the necessary Service Packs.
   c. Use the Windows Server documentation to install Internet Information Services (IIS). Use IIS to export server certificates. It must be installed before you install Certificate Services.
   d. Use the Windows Server documentation to install Certificate Services if you plan on using Active Directory over SSL.
2. Complete the following steps as a guide to create the WebSphere Portal Express administrative user:
   a. Create a user with the Windows administrative tools.

      Note: There is a 20 character limitation for the user account name.
   b. Set the password for the new user.
   c. Activate the new user with the Windows administrative tools. Set the **msDS-UserAccountDisabled** attribute to false.
3. Complete the following steps to enable SSL for Active Directory; this step sets passwords during sign-up and user creation:
   a. Install an Enterprise certificate authority on a Windows Domain Controller. It installs a certificate on a server or a third-party certificate on the Domain Controller.
   b. Click **Start** > **All Programs** > **Administrative Tools** > **Active Directory Users and Computer**.
   c. In the Active Directory Users and Computers window, right-click on your domain name and select **Properties**.
   d. In the Domain Properties dialog box, select the **Group Policy** tab.
   e. Select the **Default Domain Policy** group policy and then click **Edit**.
   f. Select **Windows Settings** under Computer Configuration.
   g. Select **Security Settings** and then select **Public Key Policies**.
   h. Select **Automatic Certificate Request Settings**.
   i. Use the wizard to add a policy for Domain Controllers.

      Note: When these requirements are complete, all domain controllers request a certificate and support LDAP over SSL with port 636.

Preparing a Novell eDirectory

If you plan to use a Novell eDirectory as an LDAP user registry, you must install and set up the server so that it communicates with IBM WebSphere Portal Express.

Procedure

1. Install Novell eDirectory. Refer to eDirectory for information.
2. Complete the following steps with the web administration tool to create the WebSphere Portal Express administrative user:
   a. Optional: Complete the following steps to create a directory suffix:
      1) Click the **Server Administration** folder in the directory server console navigation.
      2) Click the **Manage Server Properties** folder under the Server Administration folder and then select **Suffixes** on the main page.
      3) Type the **Base DN** name for the suffix; for example: `dc=yourcompany,dc=com`.
      4) Click **Add**.
      5) Click **OK** to save your changes.
   b. Open the appropriate LDIF file in the *PortalServer_root*`/installer/wp.iim/ldif` directory, with a text editor:

      Use the `PortalUsers.ldif` file as a working example and adapt appropriately to work with your LDAP server.

      Use the `ContentUsers.ldif` file for the IBM Content Manager group and user ID if you configured IBM Content Manager.
   c. Replace every `dc=yourco,dc=com` with your suffix.
   d. Replace any prefixes and suffixes that are unique to your LDAP server.
   e. You can specify user names other than `wpsadmin` and `wpsbind`. For security reasons, specify nontrivial passwords for these administrator accounts.
   f. Save your changes.
   g. Complete the instructions that are provided with your directory server to import the LDIF file.
3. Complete the following steps to create the WebSphere Portal Express administrative user:
   a. Open the appropriate LDIF file in the *PortalServer_root*`/installer/wp.iim/ldif` directory, with a text editor:

      Use the `PortalUsers.ldif` file as a working example and adapt appropriately to work with your LDAP server.

      Use the `ContentUsers.ldif` file for the IBM Content Manager group and user ID if you configured IBM Content Manager.
   b. Replace every `dc=yourco,dc=com` with your suffix.
   c. Replace any prefixes and suffixes that are unique to your LDAP server.
   d. You can specify user names other than `wpsadmin` and `wpsbind`. For security reasons, specify nontrivial passwords for these administrator accounts.
   e. Optional: If you use IBM Security Access Manager Version 5.1, set the **`objectclasses`** to `accessGroup`. If you use Security Access Manager Version 6, set the **`objectclasses`** to `groupOfNames`.
   f. Save your changes.
   g. Complete the instructions that are provided with your directory server to import the LDIF file.

Preparing a SecureWay Security Server

If you plan to use a SecureWay Security Server as an LDAP user registry, you must install and set up the server so that it communicates with IBM WebSphere Portal Express.

Procedure

1. Install SecureWay Security Server. Refer to IBM SecureWay Security Server for z/OS and OS/390® for information.
2. Complete the following steps with the web administration tool to create the WebSphere Portal Express administrative user:
   a. Optional: Complete the following steps to create a directory suffix:
      1) Click the **Server Administration** folder in the directory server console navigation.
      2) Click the **Manage Server Properties** folder under the Server Administration folder and then select **Suffixes** on the main page.
      3) Type the **Base DN** name for the suffix; for example: `dc=yourcompany,dc=com`.
      4) Click **Add**.
      5) Click **OK** to save your changes.
   b. Open the appropriate LDIF file in the *PortalServer_root*`/installer/wp.iim/ldif` directory, with a text editor:

      Use the `PortalUsers.ldif` file as a working example and adapted appropriately to work with your LDAP server.

      Use the `ContentUsers.ldif` file for the IBM Content Manager group and user ID if you configured IBM Content Manager.
   c. Replace every `dc=yourco,dc=com` with your suffix.
   d. Replace any prefixes and suffixes that are unique to your LDAP server.
   e. You can specify user names other than `wpsadmin` and `wpsbind`. For security reasons, specify nontrivial passwords for these administrator accounts.
   f. Save your changes.
   g. Complete the instructions that are provided with your directory server to import the LDIF file.
3. Complete the following steps to create the WebSphere Portal Express administrative user:
   a. Optional: Complete the following steps to create a directory suffix:
      1) Go to IBM System i and IBM i Information Center, select the appropriate documentation version and go to **Networking** > **TCP/IP applications, protocols, and services** > **IBM Directory Server for iSeries (LDAP)** > **Administering Directory Server** > **General administration tasks** > **Adding and Removing Directory Server suffixes** for information.
      2) Stop and restart the LDAP server.
   b. Open the appropriate LDIF file in the *PortalServer_root*`/installer/wp.iim/ldif` directory, with a text editor:

      Use the `PortalUsers.ldif` file as a working example and adapted appropriately to work with your LDAP server.

      Use the `ContentUsers.ldif` file for the IBM Content Manager group and user ID if you configured IBM Content Manager.
   c. Replace every `dc=yourco,dc=com` with your suffix.
   d. Replace any prefixes and suffixes that are unique to your LDAP server.

e. You can specify user names other than `wpsadmin` and `wpsbind`. For security reasons, specify nontrivial passwords for these administrator accounts.

f. Optional: If you use IBM Security Access Manager Version 5.1, set the **`objectclasses`** to `accessGroup`. If you use Security Access Manager Version 6, set the **`objectclasses`** to par.

g. Save your changes.

h. Complete the instructions that are provided with your directory server to import the LDIF file.

Preparing an Oracle Directory Server

If you plan to use an Oracle Directory Server as an LDAP user registry, you must install and set up the server so that it communicates with IBM WebSphere Portal Express. Oracle Directory Server was formerly known as Sun Java System Directory Server.

Procedure

1. Install Oracle Directory Server. Refer to Oracle software for information.
2. Complete the following steps with the web administration tool to create the WebSphere Portal Express administrative user:
   a. Optional: Complete the following steps to create a directory suffix:
      1) Click the **Server Administration** folder in the directory server console navigation.
      2) Click the **Manage Server Properties** folder under the Server Administration folder and then select **Suffixes** on the main page.
      3) Type the **Base DN** name for the suffix; for example: `dc=yourcompany,dc=com`.
      4) Click **Add**.
      5) Click **OK** to save your changes.
   b. Open the appropriate LDIF file in the *PortalServer_root*`/installer/wp.iim/ldif` directory, with a text editor:

      Use the `PortalUsers.ldif` file as a working example and adapt appropriately to work with your LDAP server.

      Use the `ContentUsers.ldif` file for the IBM Content Manager group and user ID if you configured IBM Content Manager.
   c. Replace every `dc=yourco,dc=com` with your suffix.
   d. Replace any prefixes and suffixes that are unique to your LDAP server.
   e. You can specify user names other than `wpsadmin` and `wpsbind`. For security reasons, specify nontrivial passwords for these administrator accounts.
   f. Save your changes.
   g. Complete the instructions that are provided with your directory server to import the LDIF file.
3. Complete the following steps to create the WebSphere Portal Express administrative user:
   a. Open the appropriate LDIF file in the *PortalServer_root*`/installer/wp.iim/ldif` directory, with a text editor:

      Use the `PortalUsers.ldif` file as a working example and adapt appropriately to work with your LDAP server.

      Use the `ContentUsers.ldif` file for the IBM Content Manager group and user ID if you configured IBM Content Manager.
   b. Replace every `dc=yourco,dc=com` with your suffix.

c. Replace any prefixes and suffixes that are unique to your LDAP server.
d. You can specify user names other than `wpsadmin` and `wpsbind`. For security reasons, specify nontrivial passwords for these administrator accounts.
e. Optional: If you use IBM Security Access Manager Version 5.1, set the **`objectclasses`** to `accessGroup`. If you use Security Access Manager Version 6, set the **`objectclasses`** to `groupOfNames`.
f. Save your changes.
g. Complete the instructions that are provided with your directory server to import the LDIF file.

Preparing a remote web server

Install and configure the web server plug-in. The IBM WebSphere Application Server provides the plug-in. Configure the web server to communicate with IBM WebSphere Portal Express.

Procedure

1. Install and configure the web server. Refer to the web server documentation for information.

   Dynamic cluster: If you are creating a dynamic cluster environment, install and configure an OnDemand Router (ODR). Go to Creating and configuring ODRs for information.
2. If you are using Microsoft Internet Information Server, update the **UrlSegmentMaxLength** Registry key. Change it to a value of `0` to eliminate potential problems in a WebSphere Portal Express environment with the default IIS limitation on the length of URL path segments. Update the **AllowRestrictedChars** Registry key to a value of 1 to accept hex-escaped characters in request URLs that decode to the `U+0000 - U+001F` and `U+007F - U+009F` ranges.

   Refer to Configuring Microsoft Internet Information Services (IIS) for information.

   Note: Refer to Http.sys registry settings for IIS for information.
3. If you are using IBM Domino, edit the `NOTES.INI` file on the web server. Set the **`HTTPEnableConnectorHeaders`** and **`HTTPAllowDecodedUrlPercent`** parameters to 1. Also, if you are using WebDAV, enable it in the Domino web server administrative console.
4. If you are using IBM HTTP Server or Apache Server, edit the `httpd.conf` file on the web server. Set the **`AllowEncodedSlashes`** directive to `On`. Add the directive to the root level as a global directive.

Table 27. Links to HTTP and Apache server documentation

HTTP server type	Documentation link
Read the appropriate HTTP Server documentation	IBM HTTP Server
Read the appropriate Apache Server documentation	**`AllowEncodedSlashes`** directives

5. Stop the web server.
6. Install and configure the web server plug-in on the system where the web server is located. Use the plug-ins installation wizard that is provided with WebSphere Application Server. Refer to the following topic for information:

- IBM i: LinuxWindows: Selecting a Web server topology diagram and road map

Important: Depending on how you use the web server, you must adjust the **ServerIOTimeout** parameter. It defines how long the plug-in must wait for a response from the application. The minimum value is `60` but you must increase this value if you are retrieving data from a database. To update this value, locate and open your `plugin-cfg.xml` file and set **ServerIOTimeout** to an appropriate value.

7. If you are using an Oracle iPlanet web server, some portlets require that you disable the **unix-uri-clean** or **nt-uri-clean** directives. Edit the `obj.conf` file to enable or disable these directives. Refer to the Oracle iPlanet web server documentation to determine the appropriate setting for your environment.

   Note: If you are using Oracle iPlanet web server Version 7, you must disable **uri-clean**.
8. Web 2.0 REST features in portal might require an enabled PUT and DELETE method. If your web server has these methods disabled, complete one of the following options:
   - Enable HTTP tunneling to simulate PUT and DELETE requests, which means that POST requests are used instead. See the "Switch for tunneling of HTTP methods" link for information.
   - Follow the instructions for your web server to enable PUT and DELETE requests.
9. Start the web server.
10. Optional: Complete the following steps if you plan to use the web application bridge feature:
    a. Log on to the WebSphere Integrated Solutions Console.
    b. Go to **Applications** > **Application Types** > **WebSphere enterprise applications**.
    c. Find and click the **wp.vwat.servlet.ear** application link.
    d. Under the **Web Module Properties** heading, click **Context Root For Web Modules**.
    e. Change the context root to /. This step can create name conflicts. Add a rewrite rule to avoid these conflicts. For more information read *Apache Module mod_rewrite* and *Providing short vanity URLs*.
    f. Click **OK**.
    g. Click **Save** to save your changes to the master configuration.
    h. Stop and restart the **wp.vwat.servlet.ear** application.
11. Optional: If you want to use the short version of vanity URLs, add a rewrite rule to your web server. For more information, read *Providing short vanity URLs*.

Installing the digital experience software

IBM's Exceptional Digital Experience is designed to help create, manage, simplify, and integrate your processes into an engaging online experience. IBM WebSphere Portal and IBM Web Content Manager are a part of the Exceptional Digital Experience. The product documentation uses digital experience software as a shorthand for IBM WebSphere Portal and IBM Web Content Manager.

About this task

Select a roadmap to guide you through the installation and deployment process. The installation process uses IBM Installation Manager to install the digital experience. Select your operating system to get started. After you complete the installation, you have a functional portal, unless you selected to install only the binaries for a migration path. Then, use the Configuration Wizard to complete the deployment.

"IBM i: Installing WebSphere Portal Express and Web Content Manager"
Installing the digital experience software includes preparing your operating system and using IBM Installation Manager to install IBM WebSphere Portal Express and Web Content Manager. Then, use the Configuration Wizard to complete the deployment configuration.

"Linux: Installing WebSphere Portal Express and Web Content Manager" on page 181
Installing the digital experience software includes preparing your operating system and using IBM Installation Manager to install IBM WebSphere Portal Express and Web Content Manager. Then, use the Configuration Wizard to complete the deployment configuration.

"Windows: Installing WebSphere Portal Express and Web Content Manager" on page 196
Installing the digital experience software includes preparing your operating system and using IBM Installation Manager to install IBM WebSphere Portal Express and Web Content Manager. Then, use the Configuration Wizard to complete the deployment configuration.

Related concepts:

Chapter 3, "Roadmaps," on page 69
Review the roadmaps to understand the common deployment patterns that are supported by the configuration wizard.

Related tasks:

"Planning to install WebSphere Portal Express" on page 101
Before you install IBM WebSphere Portal Express in a production environment, you need to assess your hardware and software needs, possible database configurations, security options, and LDAP server options. Skipping this important step can lead to unexpected results and costly delays.

Related reference:

"Getting the software" on page 135
There are different ways for you to get WebSphere Portal Express and Web Content Manager Version 8.5 software.

IBM i: Installing WebSphere Portal Express and Web Content Manager

Installing the digital experience software includes preparing your operating system and using IBM Installation Manager to install IBM WebSphere Portal Express and Web Content Manager. Then, use the Configuration Wizard to complete the deployment configuration.

Before you begin

Roadmaps for installation and deployment are included in this product documentation to guide you through the process. Roadmaps provide a high-level overview of the process from beginning to end. Select the configuration that is closest to the configuration that you need.

1. “IBM i: Preparing your operating system”
   Prepare the operating system to ensure a successful installation.
2. “IBM i: Preparing the Installation Manager”
   The IBM Installation Manager is used to install installation packages such as IBM WebSphere Portal Express and IBM WebSphere Application Server.
3. “IBM i: Running the installation program” on page 170
   Use the IBM Installation Manager program to install the software. IBM Installation Manager provides multiple user interfaces. Select the one that you are more comfortable with. You can use the graphical user interface or the console to create a response file for automated installations.
4. “IBM i: Converting the evaluation license” on page 175
   If you initially installed IBM WebSphere Portal Express using the evaluation license and then purchased the product, you can convert the installed evaluation license to a production license. You can do this at any time, before or after the evaluation license expires.
5. “IBM i: Next steps” on page 176
   The configuration process has changed. Use the Configuration Wizard to set up your integration with prerequisites, clusters, and more.
6. “IBM i: Upgrading the SDK” on page 178
   Starting with IBM WebSphere Portal Express Combined Cumulative Fix 05, you can change your SDK Java Technology Edition from version 7.0 to version 7.1.
7. “IBM i: Rendering documents on IBM i” on page 179
   To enable document preview function for IBM Web Content Manager and the Common Mail portlet, you must set up an HTML rendering server to work with WebSphere Portal Express. Because IBM i does not contain native graphics support, you must install extra fonts to run the document conversion that is required by these functions. Document conversion enables WebSphere Portal Express to convert documents that are produced by commonly used office programs into web pages so that they can be viewed and searched by users online. The additional fonts include an HTML rendering server that is known as X virtual frame buffer for the X server.

IBM i: Preparing your operating system

Prepare the operating system to ensure a successful installation.

Procedure

1. Make sure that you have a CD drive
2. Make sure that the server is in an unrestricted state
3. If you do not have one, create a valid IBM i user ID and password
4. Create a user profile with a user type (user class) of `*SECOFR` (other than QSECOFR) to install and configure WebSphere Portal Express

IBM i: Preparing the Installation Manager

The IBM Installation Manager is used to install installation packages such as IBM WebSphere Portal Express and IBM WebSphere Application Server.

Procedure

If you have an existing Installation Manager, start it and go to **File** > **Preferences**. Then, click **Updates**. Click the **Search for Installation Manager updates** check box. This box enables the Installation Manager to search for updates the next time you run an installation or update.
If you do not have an existing Installation Manager, then complete the following procedure:

1. Start all servers and applications that require a port number to avoid port conflicts when installing WebSphere Portal Express.
2. Type **ping** `yourserver.yourcompany.com` on a command line to verify that your fully qualified host name is properly configured.
3. Type **ping** `localhost` on a command line to verify that your network is properly configured.
4. If you are installing on a server with a firewall, antivirus, screen saver, or desktop search engine that is enabled, disable them before you install. If you do not disable them and the installation program detects them, a warning message displays during the installation.
5. Run the following task from the `IIM` directory: `installc -acceptLicense`

   Note: If you have an existing Installation Manager, search for updates to the latest supported version.

IBM i: Running the installation program

Use the IBM Installation Manager program to install the software. IBM Installation Manager provides multiple user interfaces. Select the one that you are more comfortable with. You can use the graphical user interface or the console to create a response file for automated installations.

"IBM i: Installing with the console"
Use the IBM Installation Manager to install IBM WebSphere Portal Express with an existing IBM WebSphere Application Server.

"IBM i: Creating a response file" on page 172
After IBM Installation Manager is installed, you can use it to record a response file that is based on your environment. Record a response file on the same operating system you plan for the installation. If you have multiple operating systems, you must record a response file for each operating system. Use a response file to automate your installation on multiple servers.

"IBM i: Installing with the response file" on page 174
Use either a sample response file or a response file that you created to install WebSphere Application Server and WebSphere Portal Express. You can create a response file with the graphical or console mode interface.

Related tasks:

"IBM i: Preparing your operating system" on page 169
Prepare the operating system to ensure a successful installation.

IBM i: Installing with the console:

Use the IBM Installation Manager to install IBM WebSphere Portal Express with an existing IBM WebSphere Application Server.

Before you begin

A working installation of IBM WebSphere Application Server is required. Ensure that it is not used by another copy of WebSphere Portal Express.

About this task

The installation program verifies the operating system and its prerequisites, available disk space, and any required software prerequisites before installation. You cannot install two instances of the server at the same time, even if you are installing to different directories. You must install each server completely before you install the next one.

Procedure

1. Start all servers and applications that require a port number to avoid port conflicts when you install WebSphere Portal Express. If you are installing multiple copies of IBM WebSphere Portal Express on your server, start the existing Configuration Wizard servers.
2. Open a command prompt and change to the `InstallationManager_root/eclipse/tools` directory.
3. Run the command to start the IBM Installation Manager in console mode: `imcl -c`
4. Complete the following steps to add the repositories:
   a. Enter `P` to go to the **Preferences** menu.
   b. Enter `1` to go to the **Repositories** menu.
   c. Enter `D` to add repositories.
   d. Type the path for your WebSphere Portal Express repository file.
   e. Enter `A` to apply your repositories and return to the **Preferences** menu.
   f. Enter `R` to return to the **Main** menu.
5. Enter `1` to install the software packages.
6. Select the software packages that you want to install.
   - If you are installing Portal to an existing copy of WebSphere Application Server, then select only **IBM WebSphere Portal Server.** To complete this installation option, your existing copy of WebSphere Application Server must meet the following requirements:
     – WebSphere Application Server Version 8.5.5.2 or later
     – Java Version 7
     – No existing Portal profile
   - If you are not installing Portal to an existing copy of WebSphere Application Server, then select the following packages:
     – **IBM WebSphere Application Server Network Deployment**
     – **IBM WebSphere Portal Server**
     – **IBM WebSphere SDK Java Technology Edition**

     Note: The **IBM WebSphere SDK Java Technology Edition** option is required for a WebSphere Portal Express installation even though it might be marked as "Optional."

   Tip: If you have a WebSphere Portal Express Enable license, you must select both the **IBM WebSphere Portal Express** and **IBM WebSphere Portal Enable** packages.
7. Choose one of the following options:
   - Enter `1` to choose version 8.5.0.0 for installation.
   - Enter `2` to show all available versions of the package.
8. Enter `N`.
9. Enter `A` to accept the license agreement.
10. Enter `N`.
11. Choose one of the following options for translation packages:
    - Enter `N` to select the default English package only.
    - Enter the number for the translation package you want to install.

12. On the **Incompatible package group** menu, choose one of the following options:
    - Enter `M` to change the installation directory. Then, enter the new installation directory.
    - Enter `N` to keep the existing directory.
13. Enter the number for the WebSphere Application Server root directory to use as the existing WebSphere Application Server.
14. On the **IBM Installation Manager** > **Install** > **Licenses** > **Location** > **Features** menu, enter `1` to select the **Portal Server Profile** feature. Then, enter `N` to continue.

    Note: Ensure that **Portal Server Profile** is selected to create a profile that contains the Portal application server and the product binary files. Clear this option if you need a binary only installation for migration.
15. Enter the configuration wizard administrator user ID and password.
16. If you selected the **Portal Server Profile** feature, enter the information for the following prompts:
    - **Enter the host name**
    - **Enter the node name**
    - **Enter the cell name**
    - **Enter an administrator user ID for the portal server**
    - **Enter an administrator user password for the portal server**
    - **Confirm administrator user password for the portal server**
17. Enter `N`.
18. Review the summary information.
19. Choose one of the following options:
    - Enter `G` to generate a response file.
    - Enter `I` to install WebSphere Portal Express.
20. When the installation is complete, enter `F` to return to the main installation menu.
21. Access the Configuration Wizard. Go to `http://`*`your_server`*`:`*`10200`*`/ibm/wizard`.
22. Log in to the Configuration Wizard with the administrative ID for the configuration wizard profile, `cw_profile`.

    Note: The wizard user interface might not be available in all languages. If the language is not currently supported, you might see the English version. For details on the supported languages for all of the WebSphere Portal Express user interfaces, see “Supported languages” on page 3622.

What to do next

Before you access WebSphere Application Server, configure the software license agreement to set the usage limit from the Proof of Entitlement (POE) or invoice. Go to Configuring software license information for information.

IBM i: Creating a response file:

After IBM Installation Manager is installed, you can use it to record a response file that is based on your environment. Record a response file on the same operating system you plan for the installation. If you have multiple operating systems, you

must record a response file for each operating system. Use a response file to automate your installation on multiple servers.

About this task

Procedure

1. Start all servers and applications that require a port number to avoid port conflicts when you install WebSphere Portal Express. If you are installing multiple copies of IBM WebSphere Portal Express on your server, start the existing Configuration Wizard servers.
2. Go to the `InstallationManager_root/eclipse/tools` directory.
3. Run the following task to start the recording: `imcl -c`
4. Complete the following steps to add the repositories:
   a. Enter `P` to go to the **Preferences** menu.
   b. Enter `1` to go to the **Repositories** menu.
   c. Enter `D` to add repositories.
   d. Type the path for your WebSphere Portal Express repository file.
   e. Enter `A` to apply your repositories and return to the **Preferences** menu.
   f. Enter `R` to return to the **Main** menu.
5. Enter `1` to install the software packages.
6. Select the software packages that you want to install.
   - If you are installing Portal to an existing copy of WebSphere Application Server, then select only **IBM WebSphere Portal Server.** To complete this installation option, your existing copy of WebSphere Application Server must meet the following requirements:
     – WebSphere Application Server Version 8.5.5.2 or later
     – Java Version 7
     – No existing Portal profile
   - If you are not installing Portal to an existing copy of WebSphere Application Server, then select the following packages:
     – **IBM WebSphere Application Server Network Deployment**
     – **IBM WebSphere Portal Server**
     – **IBM WebSphere SDK Java Technology Edition**

     Note: The **IBM WebSphere SDK Java Technology Edition** option is required for a WebSphere Portal Express installation even though it might be marked as "Optional."

   Tip: If you have a WebSphere Portal Express Enable license, you must select both the **IBM WebSphere Portal Express** and **IBM WebSphere Portal Enable** packages.
7. Choose one of the following options:
   - Enter `1` to choose version 8.5.0.0 for installation.
   - Enter `2` to show all available versions of the package.
8. Enter `N`.
9. Enter `A` to accept the license agreement.
10. Enter `N`.
11. Choose one of the following options for translation packages:
    - Enter `N` to select the default English package only.

- Enter the number for the translation package you want to install.

12. On the **Incompatible package group** menu, choose one of the following options:
    - Enter `M` to change the installation directory. Then, enter the new installation directory.
    - Enter `N` to keep the existing directory.
13. Enter the number for the WebSphere Application Server root directory to use as the existing WebSphere Application Server.
14. Enter the configuration wizard administrator user ID and password.
15. If you selected the **Portal Server Profile** feature, enter the information for the following prompts:
    - **Enter the host name**
    - **Enter the node name**
    - **Enter the cell name**
    - **Enter an administrator user ID for the portal server**
    - **Enter an administrator user password for the portal server**
    - **Confirm administrator user password for the portal server**
16. Enter `N`.
17. Review the summary information.
18. Enter `G` to generate a response file.
19. If you plan to install on a different computer, copy the response file to the response file directory on that computer.

IBM i: Installing with the response file:

Use either a sample response file or a response file that you created to install WebSphere Application Server and WebSphere Portal Express. You can create a response file with the graphical or console mode interface.

About this task

Locate the sample response file in the *setup_root*`/responsefiles` directory. Modify the file with your environment values.

The installation program verifies the operating system and its prerequisites, available disk space, and any required software prerequisites before installation. You cannot install two instances of the server at the same time, even if you are installing to different directories. You must install each server completely before you install the next one.

Procedure

1. If the repository URL requires authentication, use the IBM Installation Manager command-line tool to create a secure **Storage File**.

   The secure **Storage File** stores the credentials that are required for the repositories. The IBM Installation Manager command-line tool **imutilsc** is available from the installation tools directory. The following information is an example of the **imutilsc** key ring file command:

   ```
   imutilsc saveCredential -url repository_URL -userName credential_userName
    -userPassword password -secureStorageFile storage_file [ -masterPasswordFile master_password_file ]
   [ -preferences com.ibm.cic.common.core.preferences.ssl.nonsecureMode=true|false ]
   ```

```
 [ -proxyHost proxy_host -proxyPort proxy_port
  [ -proxyUsername proxy_username -proxyUserPassword proxyuser_password ]
  [ -useSocks ] ]
 [ -verbose ]
```

Tip: If you install on a different computer, copy the secure **Storage File** to that computer.

2. Go to the `InstallationManager_root/eclipse/tools` directory.
3. Run the following task to install WebSphere Portal Express and IBM WebSphere Application Server:

   Tip: Add the **-secureStorageFile** *pathtosecureStorageFile* **-masterPasswordFile** *pathtomasterPasswordFile* parameters to the **imcl** command if you are using a secure **Storage File** to store credentials.
   `imcl -acceptLicense input pathtoresponse.xml -log dirpath/logfilename`

What to do next

Before you access WebSphere Application Server, configure the software license agreement to set the usage limit from the Proof of Entitlement (POE) or invoice. Go to Configuring software license information for information.

Attention: When you install WebSphere Application Server and WebSphere Portal Express together, you might find the following message in the `SystemErr.log` file:
AppServer_root`/properties/version/installed.xml (No such file or directory)`

This message is part of the installation process and is not repeated after the installation is complete. Therefore, the message can be ignored.

IBM i: Converting the evaluation license

If you initially installed IBM WebSphere Portal Express using the evaluation license and then purchased the product, you can convert the installed evaluation license to a production license. You can do this at any time, before or after the evaluation license expires.

Before you begin

Review the evaluation license.

About this task

You do not need to reinstall the program files to convert the evaluation license. However, the license conversion utility must have access to the purchased production software. You cannot use the evaluation software to convert the evaluation license.

Complete the following steps to convert the evaluation license:

Procedure

1. Run the `stopServer` *WebSphere_Portal* `-username` *admin_userid* `-password` *admin_password* task from the *wp_profile_root*`/bin` directory.
2. Run the `ConfigEngine.sh install-license -Dlicense=`*media_filepath* task from the *wp_profile_root*`/ConfigEngine` directory.

   Tip: *media_filepath* is the location of the `PortalExpress` subdirectory on either the WebSphere Portal Express CD or the downloaded installation image, for

example */QOPT/disk_volume_label/*PortalExpress where *QOPT* is the disk mount point or /wpdownload/PortalExpress for the installation image.

When the task completes, a message confirms that the product is now licensed for use.

3. Run the startServer *WebSphere_Portal* task.

Related tasks:

"IBM i: Preparing your operating system" on page 169
Prepare the operating system to ensure a successful installation.

"IBM i: Preparing the Installation Manager" on page 169
The IBM Installation Manager is used to install installation packages such as IBM WebSphere Portal Express and IBM WebSphere Application Server.

IBM i: Next steps

The configuration process has changed. Use the Configuration Wizard to set up your integration with prerequisites, clusters, and more.

Verify that the installation was successful

Unless you selected to install only the binary files, you can log in to the portal and verify that your installation was successful. Access WebSphere Portal Express with the http://*yourserver*:*yourport*/wps/portal format.

If you are not sure what your port number is, use the **list-server-ports** command to determine the port number. Change to the . Run the **list-server-ports** task to generate the WebSphere_Portal_PortMatrix.txt file. For example: ConfigEngine.sh list-server-ports -DWasPassword=*password*.

Change to the . Run the **list-server-ports** task to generate the server1_PortMatrix.txt file. For example: ConfigEngine.sh list-server-ports -DWasPassword=*password*.

Modify the uri.home.substitution custom property in the Resource Environment Provider

In the **WP ConfigService**, set the **uri.home.substitution** custom property to true to avoid functional errors that might occur in certain unusual use cases. For more information on **uri.home.substitution**, see *Configuration Service* in the related links.

Select a roadmap

If you have not already selected a roadmap to guide you through the installation and deployment process, look at the available roadmaps. The roadmaps are based on typical environments, such as development environment, test environment, and more. They provide a high-level overview of the installation and deployment process. Each roadmap includes a topology diagram, usage recommendations, and instructions. Roadmaps are available for both new deployments and migrations scenarios.

"Roadmaps for installation and deployment" on page 69

"Roadmaps for migration" on page 92

Run the Configuration Wizard to finish the deployment

Then, start the Configuration Wizard and select the option that is defined in your selected roadmap. Use the wizard to run the configuration steps in real time or to generate scripts. The wizard generates instructions and scripts specific to your environment and the data that you entered. You can save the selection and data that you entered as an XML file. Then, you can load the XML file during a subsequent wizard session to set up a similar configuration on a different server.

If you select to run the configuration in real time, you still have the opportunity to download scripts for selected steps. For example, if only the database administrator can databases, then you can download the database creation script and give it to your database administrator to run.

Default URLs

During the configuration process, you might need to following URLs to access different administration user interfaces.

Use the following default URLs to access IBM WebSphere Portal Express, the WebSphere Integrated Solutions Console, and the Configuration Wizard:

IBM WebSphere Portal Express

`http://localhost:10039/wps/portal`

`https://localhost:10042/wps/portal`

WebSphere Integrated Solutions Console

`http://localhost:10038/ibm/console`

`https://localhost:10041/ibm/console`

Configuration Wizard

`http://localhost:10200/ibm/wizard`

`https://localhost:10202/ibm/wizard`

If you had any processes from other software in the default port range when the installation started, you might have different port numbers than the defaults.

Related concepts:

Chapter 3, “Roadmaps,” on page 69
Review the roadmaps to understand the common deployment patterns that are supported by the configuration wizard.

“Configuration Wizard” on page 232
Use the Configuration Wizard to set up stand-alone servers and new deployments, create clusters, migrate and update to new versions, and add new capabilities to existing deployments.

Related tasks:

“IBM i: Preparing your operating system” on page 169
Prepare the operating system to ensure a successful installation.

“IBM i: Preparing the Installation Manager” on page 169
The IBM Installation Manager is used to install installation packages such as IBM WebSphere Portal Express and IBM WebSphere Application Server.

"IBM i: Running the installation program" on page 170
Use the IBM Installation Manager program to install the software. IBM Installation Manager provides multiple user interfaces. Select the one that you are more comfortable with. You can use the graphical user interface or the console to create a response file for automated installations.

Related reference:

"Configuration Service" on page 297
The portal Configuration Service is responsible for collecting the most essential configuration data of the IBM WebSphere Portal Express engine.

IBM i: Upgrading the SDK

Starting with IBM WebSphere Portal Express Combined Cumulative Fix 05, you can change your SDK Java Technology Edition from version 7.0 to version 7.1.

About this task

IBM i note: SDK version 7.1 is an optional prerequisite for WebSphere Application Server on IBM i and has a separate, non-Installation Manager based installation.

Procedure

1. Install one of the following PTF Groups or higher:
   - V7R1: **PTF Group SF99572 level 15** or higher
   - V7R2: **PTF Group SF99716 level 4** or higher
2. Read the following information before you run the **managesdk** task:
   - For stand-alone environments: Stop the profile server (node) before you run the **managesdk** command.
   - For clustered environments:
     - If the profile is a federated node of a deployment manager, ensure that the deployment manager is running before you run the **managesdk** command to update the profile.
     - Stop all the nodes.
     - Ensure that the node agent for each node is started.
     - When you enable the SDK for a node, run the **managesdk** command from the `/bin` directory to which the node belongs. You can also run the command from the `/bin` directory of the profile that contains the node that you want to update.
     - A connection to the deployment manager must exist with a supported connector protocol in the following order of preference:
       - SOAP
       - Inter-Process Communications (IPC)
       - Remote Method Invocation (RMI)

       If the SOAP protocol is enabled, the **managesdk** command uses the SOAP protocol. If the SOAP protocol is not enabled but the IPC protocol is enabled, the command uses the IPC protocol. If the SOAP and IPC protocol are not enabled, then the command uses the RMI protocol.
     - You must provide the administrative user name and password with the **managesdk** command for each profile that contains a federated node or deployment manager node in a cell with security enabled. If you do not specify the **-user** and **-password** parameters, the **managesdk** command might fail or stop processing.

– When you enable the SDK for a deployment manager, only the deployment manager server is enabled. None of the managed nodes of the deployment manager are enabled to use the specific SDK.

3. Open a command prompt and change to the *AppServer_root*/bin directory.
4. Run the following command to enable all existing profiles to use the new SDK version:

   Go to managesdk command for information about the **managesdk** commands.

   - IBM i: `managesdk -enableProfileAll -sdkname 1.7.1_64 -enableServers`
5. Run the following commands to make the new SDK version the new default:
   - IBM i: `managesdk -setCommandDefault -sdkname 1.7.1_64`

     `managesdk -setNewProfileDefault -sdkname 1.7.1_64`
6. Repeat these steps on each node in your environment.

Related tasks:

"IBM i: Preparing your operating system" on page 169
Prepare the operating system to ensure a successful installation.

"IBM i: Preparing the Installation Manager" on page 169
The IBM Installation Manager is used to install installation packages such as IBM WebSphere Portal Express and IBM WebSphere Application Server.

"IBM i: Running the installation program" on page 170
Use the IBM Installation Manager program to install the software. IBM Installation Manager provides multiple user interfaces. Select the one that you are more comfortable with. You can use the graphical user interface or the console to create a response file for automated installations.

"IBM i: Converting the evaluation license" on page 175
If you initially installed IBM WebSphere Portal Express using the evaluation license and then purchased the product, you can convert the installed evaluation license to a production license. You can do this at any time, before or after the evaluation license expires.

IBM i: Rendering documents on IBM i

To enable document preview function for IBM Web Content Manager and the Common Mail portlet, you must set up an HTML rendering server to work with WebSphere Portal Express. Because IBM i does not contain native graphics support, you must install extra fonts to run the document conversion that is required by these functions. Document conversion enables WebSphere Portal Express to convert documents that are produced by commonly used office programs into web pages so that they can be viewed and searched by users online. The additional fonts include an HTML rendering server that is known as X virtual frame buffer for the X server.

About this task

Before you begin, you must have:

A WebSphere Portal Express profile that runs on your IBM i system; record the name of your WebSphere Portal Express profile for future reference.

OS/400 - Additional Fonts (5770SS1, Option 43).

After you set up your HTML rendering server, you must also enable the Document Conversion Services.

Complete the following tasks on each WebSphere Portal Express node to render documents on IBM i:

Note: Use the HTML rendering server that you associate with your WebSphere Portal Express profile only for WebSphere Portal Express. Using the HTML rendering server with other applications might cause problems.

"IBM i: Configuring an HTML rendering server on IBM i"
After you have installed OS/400 - Additional Fonts (5770SS1, Option 43), on your IBM i system, an HTML rendering server (X virtual frame buffer for X server) is present. You must select a display number for the HTML rendering server.

"IBM i: Associating an HTML rendering server with WebSphere Portal Express on IBM i" on page 181
After you select a display number for the HTML rendering server (X virtual frame buffer for X server), you must associate this server with the installed IBM WebSphere Portal Express profile.

Related concepts:

"IBM i: Next steps" on page 176
The configuration process has changed. Use the Configuration Wizard to set up your integration with prerequisites, clusters, and more.

Related tasks:

"IBM i: Preparing your operating system" on page 169
Prepare the operating system to ensure a successful installation.

"IBM i: Preparing the Installation Manager" on page 169
The IBM Installation Manager is used to install installation packages such as IBM WebSphere Portal Express and IBM WebSphere Application Server.

"IBM i: Running the installation program" on page 170
Use the IBM Installation Manager program to install the software. IBM Installation Manager provides multiple user interfaces. Select the one that you are more comfortable with. You can use the graphical user interface or the console to create a response file for automated installations.

"IBM i: Converting the evaluation license" on page 175
If you initially installed IBM WebSphere Portal Express using the evaluation license and then purchased the product, you can convert the installed evaluation license to a production license. You can do this at any time, before or after the evaluation license expires.

IBM i: Configuring an HTML rendering server on IBM i:

After you have installed OS/400 - Additional Fonts (5770SS1, Option 43), on your IBM i system, an HTML rendering server (X virtual frame buffer for X server) is present. You must select a display number for the HTML rendering server.

Procedure

1. If the QShell Interpreter is running, perform the following commands on the IBM i command line to stop it:

   ```
   QSH
   ```

   Press **F3**
2. Type `CALL QP2TERM` on the command line to start the OS/400 Portable Application Solutions Environment (OS/400 PASE) console.
3. Type `ps gaxuw | grep Xvnc ; ps gaxuw | grep vfb` on the command line to list all active HTML rendering servers:

   Note: If other rendering servers are already active, you might see output such as this (the number after the colon is the display number already in use):

```
v2kea554 40571 0.0  0.0 12484 0 - A Jul 13 4:08
/QOpenSys/QIBM/ProdData/DeveloperTools/vnc/Xvnc:6 -desktop X -httpd
```

4. Select any number from 1 to 99 that is not in use.
5. Press **F3** to return to the command line interface.
6. Type `SBMJOB CMD(CALL PGM(QP2SHELL) PARM('/usr/bin/X11/X' '-vfb' ':N')) JOB(XVFB) JOBQ(QSYSNOMAX) ALWMLTTHD(*YES)` on the command line to start the HTML rendering server, where *N* is the display number.
7. Verify that the HTML rendering server is started by repeating the prior steps to start PASE and list the active servers, confirming that an HTML rendering server with your display number is in the list.

What to do next

After you render your documents, you must enable the document conversion services. Go to Chapter 6, "Document Conversion Services," on page 727 for information.

IBM i: Associating an HTML rendering server with WebSphere Portal Express on IBM i:

After you select a display number for the HTML rendering server (X virtual frame buffer for X server), you must associate this server with the installed IBM WebSphere Portal Express profile.

Procedure

1. Log on to the WebSphere Integrated Solutions Console.
2. Go to **Servers** > **Server Types** > **WebSphere application servers** > **WebSphere_Portal** > **Server Infrastructure** > **Java and Process Management** > **Process definition** > **Environment Entries**.
3. Click **New**.
4. In the **Name** field type `DISPLAY`.
5. In the **Value** field type *host_name*`:n`, where *host_name* is the TCP/IP host name of your system and *n* is the display number of the HTML rendering server. (Example: `mysystem.xland.company.com:1`).
6. Click **OK**.
7. Save your changes to the master WebSphere Application Server configuration file.
8. Restart the server.

What to do next

After you render your documents, you must enable the document conversion services. Go to Chapter 6, "Document Conversion Services," on page 727 for information.

Linux: Installing WebSphere Portal Express and Web Content Manager

Installing the digital experience software includes preparing your operating system and using IBM Installation Manager to install IBM WebSphere Portal Express and Web Content Manager. Then, use the Configuration Wizard to complete the deployment configuration.

Before you begin

Roadmaps for installation and deployment are included in this product documentation to guide you through the process. Roadmaps provide a high-level overview of the process from beginning to end. Select the configuration that is closest to the configuration that you need.

1. "Linux: Preparing your operating system"
   Prepare the operating system to ensure a successful installation.
2. "Linux: Preparing the Installation Manager" on page 183
   The IBM Installation Manager is used to install installation packages such as IBM WebSphere Portal Express and IBM WebSphere Application Server.
3. "Linux: Running the installation program" on page 184
   Use the IBM Installation Manager program to install the software. IBM Installation Manager provides multiple user interfaces. Select the one that you are more comfortable with. You can use the graphical user interface or the console to create a response file for automated installations.
4. "Linux: Converting the evaluation license" on page 192
   If you initially installed IBM WebSphere Portal Express using the evaluation license and then purchased the product, you can convert the installed evaluation license to a production license. You can do this at any time, before or after the evaluation license expires.
5. "Linux: Next steps" on page 193
   The configuration process has changed. Use the Configuration Wizard to set up your integration with prerequisites, clusters, and more.
6. "Linux: Upgrading the SDK" on page 195
   Starting with IBM WebSphere Portal Express Combined Cumulative Fix 05, you can change your SDK Java Technology Edition from version 7.0 to version 7.1.

Linux: Preparing your operating system

Prepare the operating system to ensure a successful installation.

Procedure

1. If you are using IBM GPFS file sharing, set the file system inodes limit to 25000 or higher. For example, run the **mmchfs /dev/gpfs1nsd -F 250000** command. */dev/gpfs1nsd* is the IBM WebSphere Portal Express installation file system.
2. Set the file descriptor limit to `10240`.

   ```
   ulimit -n 10240
   ```

3. **Web Content Manager only:** Complete the following steps to remove any file size limits: Use the **ulimit -f** command to set the maximum size of files that can be created. Set this value to at least the size of the largest file you would upload to the content server. The **ulimit -f unlimited** command removes any limit on file size.
   a. Open a command prompt.
   b. Run the `ulimit -a` command to get a list of all the **ulimit** settings.
   c. Find the **-f** parameter. If the value is `unlimited`, no further action is required.
   d. If the **-f** parameter is not `unlimited`, choose one of the following options:
      - Remove the file size limit permanently. For information about how to remove the file size limit see your system documentation.
      - Run the `ulimit -f unlimited` command for each session.

4. Install and configure X server on Linux (such as X Window System or GNOME) to use the graphical user interface the installation program provides. For information about adding packages, read your system documentation.

   Note: If you plan to install with a response file, X server is not required.

Linux: Preparing the Installation Manager

The IBM Installation Manager is used to install installation packages such as IBM WebSphere Portal Express and IBM WebSphere Application Server.

Procedure

If you have an existing Installation Manager, start it and go to **File** > **Preferences**. Then, click **Updates**. Click the **Search for Installation Manager updates** check box. This box enables the Installation Manager to search for updates the next time you run an installation or update.
If you do not have an existing Installation Manager, then complete the following procedure:

1. Start all servers and applications that require a port number to avoid port conflicts when installing WebSphere Portal Express.
2. Type **ping** `yourserver.yourcompany.com` on a command line to verify that your fully qualified host name is properly configured.
3. Type **ping** `localhost` on a command line to verify that your network is properly configured.
4. If you are installing on a server with a firewall, antivirus, screen saver, or desktop search engine that is enabled, disable them before you install. If you do not disable them and the installation program detects them, a warning message displays during the installation.
5. Optional: Complete the following steps to install as a non-root user:
   a. Log in to the operating system as the root user.
   b. Open a command line.
   c. Use the appropriate system commands to create the following items:

      Non-root user, including password

      Group

      Directory, used when you install IBM Installation Manager and WebSphere Portal Express
   d. Set the user profile for the number of open files. Set the value to **ulimit -n 10240**.
   e. Add the non-root user to the new group.
   f. Run the following task to change the owner of the directory to the non-root user:

      `chown user:group /directory`
   g. Run the following task to change permissions for the directory:

      `chmod 755 /directory`
   h. Log in as the non-root user.
   i. Use the following command to install IBM Installation Manager:

      `./userinst`

      Set the IBM Installation Manager installation location to `/directory/IBM/InstallationManager`.
6. Run the following task from the `IIM` directory: `./install`

7. Run the following task from the Portal Setup disk if you want to start the launchpad to complete all necessary installation steps: `./setup.sh`

   Launchpad tip: On the Launchpad: Install Portal panel, you have the following options:

   Install IBM WebSphere Portal from media
   Choose this option if you are installing directly from the DVD.

   Install IBM WebSphere Portal from network
   Choose this option if you downloaded the electronic images.

   Install IBM Installation Manager only
   Choose this option if you want to install multiple IBM products, or if you want to install WebSphere Portal Express directly from IBM Passport Advantage, and a previous version of Installation Manager is not already installed on your system.

   Restriction: If you have a previously installed 32-bit version of Installation Manager, then you cannot proceed with this option. You must manually install the 64-bit version of Installation Manager that is specific to your operating system using the Setup disc before you proceed with the WebSphere Portal Express installation.

   Tip: Add the `LaunchPadLocale` *`language_code`* to the **`setup`** task to change the display to your user locale or to another language.

   Note: If the language is not currently supported for the user interface, you might see the English version. For details on supported languages and the language codes for all of the WebSphere Portal Express user interfaces, see "Supported languages" on page 3622.

   After you complete all necessary steps from the launchpad, verify that your installation was successful.

Linux: Running the installation program

Use the IBM Installation Manager program to install the software. IBM Installation Manager provides multiple user interfaces. Select the one that you are more comfortable with. You can use the graphical user interface or the console to create a response file for automated installations.

"Linux: Installing with the graphical user interface" on page 185
Use the IBM Installation Manager to install IBM WebSphere Portal Express, IBM WebSphere Application Server, and the Apache Derby database.

"Linux: Installing with the console" on page 187
Use the IBM Installation Manager to install IBM WebSphere Portal Express with an existing IBM WebSphere Application Server.

"Linux: Creating a response file" on page 189
After IBM Installation Manager is installed, you can use it to record a response file that is based on your environment. Record a response file on the same operating system you plan for the installation. If you have multiple operating systems, you must record a response file for each operating system. Use a response file to automate your installation on multiple servers.

"Linux: Installing with the response file" on page 191
Use either a sample response file or a response file that you created to install WebSphere Application Server and WebSphere Portal Express. You can create a response file with the graphical or console mode interface.

Related tasks:

"Linux: Preparing your operating system" on page 182
Prepare the operating system to ensure a successful installation.

Linux: Installing with the graphical user interface:

Use the IBM Installation Manager to install IBM WebSphere Portal Express, IBM WebSphere Application Server, and the Apache Derby database.

About this task

The installation program verifies the operating system and its prerequisites, available disk space, and any required software prerequisites before installation. You cannot install two instances of the server at the same time, even if you are installing to different directories. You must install each server completely before you install the next one.

Procedure

1. Start all servers and applications that require a port number to avoid port conflicts when you install WebSphere Portal Express. If you are installing multiple copies of IBM WebSphere Portal Express on your server, start the existing Configuration Wizard servers.
2. If necessary, start the Installation Manager.
3. After you install or upgrade the Installation Manager, complete the following steps to add the repositories where the installation media exists:
   a. Open the IBM Installation Manager and go to **File** > **Preferences** > **Repositories**.
   b. Select **Add Repositories**.
   c. Select **Browse** and go to the *Portal-install-eimage*/Setup/repository.config file and then click **OK**.
   d. Ensure that all required repositories are checked. Then, click **Test Connections** to ensure that the IBM Installation Manager can successfully access the directory where the service repositories are stored.
   e. Select **Apply**.
   f. Select **OK**.
4. On the main IBM Installation Manager panel, select **Install**.
5. Optional: If you are installing Portal to an existing copy of WebSphere Application Server, then select only **IBM WebSphere Portal Server** on the Install Packages screen. To complete this installation option, your existing copy of WebSphere Application Server must meet the following requirements:
   - WebSphere Application Server Version 8.5.5.2 or later
   - Java Version 7
   - No existing Portal profile

   Skip the following step, if you choose to install to an existing copy of WebSphere Application Server.
6. On Install Packages: Select packages to install, select the following packages, and then click **Next**:
   - **IBM WebSphere Application Server Network Deployment**
   - **IBM WebSphere Portal Server**
   - **IBM WebSphere SDK Java Technology Edition**

Note: The **IBM WebSphere SDK Java Technology Edition** option is required for a WebSphere Portal Express installation even though it might be marked as "Optional" on the Select packages to install screen.

Tip: If you have a WebSphere Portal Express Enable license, you must select both the **IBM WebSphere Portal Express** and **IBM WebSphere Portal Enable** packages.

7. On Install Packages: Select the fixes to install screen, select any required fixes. Then, click **Next**.
8. Accept the license agreement and then click **Next**.
9. Enter the directory where you want to store shared resources and then click **Next**.
10. Complete the following steps on the Install Packages: Installation Directory panel:

    Remember: The installation directory that you specify must NOT contain any files or the following characters: `~ ! @ # $ % ^ & * ( ) + { } | < > ? ` = [ ] ; ' , . "` and spaces.

    a. Select the WebSphere Application Server Package Group Name and then enter the installation directory path.
    b. Select the WebSphere Portal Express Package Group Name and then enter the installation directory path.
    c. Click **Next**.
11. Select the translations to install and then click **Next**.
12. Optional: If you chose to install Portal to an existing copy of WebSphere Application Server, select the available copy of WebSphere Application Server, and click **Next**.

    Note: Skip this step, if you are not installing Portal to an existing copy of WebSphere Application Server.
13. On Install Packages: Select the features to install, expand the WebSphere Application Server and WebSphere Portal Express packages to modify the features you want to install and then click **Next**.

    Note: Ensure that **Portal Server Profile** is selected to create a profile that contains the Portal application server and the product binary files. Clear this option if you need a binary only installation for migration.

    Note: As you select the items, read the **Details** section for information.
14. On Profile configuration details, enter the user ID and password for the configuration wizard administrator. Then, click **Next**.
15. If you selected the **Portal Server Profile** package, click **Enter the Administrator user ID and password for the Portal Server**.
16. Confirm the Summary information and then click **Install**.

What to do next

After a successful installation, the summary displays. Choose the **Portal First Steps** radio button and then click **Finish** to start the servers and begin configuring WebSphere Portal Express with the Configuration Wizard.

Tip: To access First Steps later, you can either select **First Steps** from the **Start** menu or you can run the **./firststeps.sh** task from the *wp_profile_root*/

PortalServer/installer/wp.firststeps directory. Add the LaunchPadLocale *language_code* to the **firststeps** task to change the display to your user locale or to another language.

Attention: When you install WebSphere Application Server and WebSphere Portal Express together, you might find the following message in the SystemErr.log file:
AppServer_root/properties/version/installed.xml (No such file or directory)

This message is part of the installation process and is not repeated after the installation is complete. Therefore, the message can be ignored.

Linux: Installing with the console:

Use the IBM Installation Manager to install IBM WebSphere Portal Express with an existing IBM WebSphere Application Server.

Before you begin

A working installation of IBM WebSphere Application Server is required. Ensure that it is not used by another copy of WebSphere Portal Express. Before you install the WebSphere Portal Express package, install **IBM WebSphere SDK Java Technology Edition** into the same directory where IBM WebSphere Application Server is installed.

About this task

The installation program verifies the operating system and its prerequisites, available disk space, and any required software prerequisites before installation. You cannot install two instances of the server at the same time, even if you are installing to different directories. You must install each server completely before you install the next one.

Procedure

1. Start all servers and applications that require a port number to avoid port conflicts when you install WebSphere Portal Express. If you are installing multiple copies of IBM WebSphere Portal Express on your server, start the existing Configuration Wizard servers.
2. Open a command prompt and change to the InstallationManager_root/eclipse/tools directory.
3. Run the command to start the IBM Installation Manager in console mode: ./imcl -c
4. Complete the following steps to add the repositories:
   a. Enter P to go to the **Preferences** menu.
   b. Enter 1 to go to the **Repositories** menu.
   c. Enter D to add repositories.
   d. Type the path for your WebSphere Portal Express repository file.
   e. Enter A to apply your repositories and return to the **Preferences** menu.
   f. Enter R to return to the **Main** menu.
5. Enter 1 to install the software packages.
6. Select the software packages that you want to install.
   - If you are installing Portal to an existing copy of WebSphere Application Server, then select only **IBM WebSphere Portal Server.** To complete this

installation option, your existing copy of WebSphere Application Server must meet the following requirements:
  - WebSphere Application Server Version 8.5.5.2 or later
  - Java Version 7
  - No existing Portal profile
- If you are not installing Portal to an existing copy of WebSphere Application Server, then select the following packages:
  - **IBM WebSphere Application Server Network Deployment**
  - **IBM WebSphere Portal Server**
  - **IBM WebSphere SDK Java Technology Edition**

    Note: The **IBM WebSphere SDK Java Technology Edition** option is required for a WebSphere Portal Express installation even though it might be marked as "Optional."

Tip: If you have a WebSphere Portal Express Enable license, you must select both the **IBM WebSphere Portal Express** and **IBM WebSphere Portal Enable** packages.

7. Choose one of the following options:
   - Enter `1` to choose version 8.5.0.0 for installation.
   - Enter `2` to show all available versions of the package.
8. Enter `N`.
9. Enter `A` to accept the license agreement.
10. Enter `N`.
11. Choose one of the following options for translation packages:
    - Enter `N` to select the default English package only.
    - Enter the number for the translation package you want to install.
12. On the **Incompatible package group** menu, choose one of the following options:
    - Enter `M` to change the installation directory. Then, enter the new installation directory.
    - Enter `N` to keep the existing directory.
13. Enter the number for the WebSphere Application Server root directory to use as the existing WebSphere Application Server.
14. On the **IBM Installation Manager** > **Install** > **Licenses** > **Location** > **Features** menu, enter `1` to select the **Portal Server Profile** feature. Then, enter `N` to continue.

    Note: Ensure that **Portal Server Profile** is selected to create a profile that contains the Portal application server and the product binary files. Clear this option if you need a binary only installation for migration.
15. Enter the configuration wizard administrator user ID and password.
16. If you selected the **Portal Server Profile** feature, enter the information for the following prompts:
    - **Enter the host name**
    - **Enter the node name**
    - **Enter the cell name**
    - **Enter an administrator user ID for the portal server**
    - **Enter an administrator user password for the portal server**

- **Confirm administrator user password for the portal server**

17. Enter `N`.
18. Review the summary information.
19. Choose one of the following options:
    - Enter `G` to generate a response file.
    - Enter `I` to install WebSphere Portal Express.
20. When the installation is complete, enter `F` to return to the main installation menu.
21. Access the Configuration Wizard. Go to `http://your_server:10200/ibm/wizard`.
22. Log in to the Configuration Wizard with the administrative ID for the configuration wizard profile, `cw_profile`.

    Note: The wizard user interface might not be available in all languages. If the language is not currently supported, you might see the English version. For details on the supported languages for all of the WebSphere Portal Express user interfaces, see "Supported languages" on page 3622.

Linux: Creating a response file:

After IBM Installation Manager is installed, you can use it to record a response file that is based on your environment. Record a response file on the same operating system you plan for the installation. If you have multiple operating systems, you must record a response file for each operating system. Use a response file to automate your installation on multiple servers.

About this task

You can also use the console mode to generate a response file. After you review the summary information, enter `G` to generate a response file instead of `I` to install.

Procedure

1. Start all servers and applications that require a port number to avoid port conflicts when you install WebSphere Portal Express. If you are installing multiple copies of IBM WebSphere Portal Express on your server, start the existing Configuration Wizard servers.
2. Go to the `InstallationManager_root/eclipse` directory.
3. Run the following task to start the recording: `./IBMIM -record pathtoresponsexmlfile -skipInstall tempinstalldirectory`

   `-record`
   This parameter indicates recording the actions and parameters into the response file. The *pathtoresponsexmlfile* text is the name of the response file, for example `Wp8SampleResp.xml`. The *tempinstalldirectory* text is the directory where the response file is recorded.

   `-skipInstall`
   This parameter indicates that no actual installation is completed even though it leads to the final pane.

   tempinstalldirectory
   This value is a directory where the installation saves the history and data when you record the response files.
4. Complete the following steps to add the repositories where the installation media exists:

a. Open the IBM Installation Manager and go to **File** > **Preferences** > **Repositories**.
b. Select **Add Repositories**.
c. Select **Browse** and go to the *Portal-install-eimage*/Setup/repository.config file and then click **OK**.
d. Ensure that all required repositories are checked. Then, click **Test Connections** to ensure that the IBM Installation Manager can successfully access the directory where the service repositories are stored.
e. Select **Apply**.
f. Select **OK**.

5. On the main IBM Installation Manager panel, select **Install**.
6. Optional: If you are installing Portal to an existing copy of WebSphere Application Server, then select only **IBM WebSphere Portal Server** on the Install Packages screen. To complete this installation option, your existing copy of WebSphere Application Server must meet the following requirements:
   - WebSphere Application Server Version 8.5.5.2 or later
   - Java Version 7
   - No existing Portal profile

   Skip the following step, if you choose to install to an existing copy of WebSphere Application Server.
7. On Install Packages: Select packages to install, select the following packages, and then click **Next**:
   - **IBM WebSphere Application Server Network Deployment**
   - **IBM WebSphere Portal Server**
   - **IBM WebSphere SDK Java Technology Edition**

     Note: The **IBM WebSphere SDK Java Technology Edition** option is required for a WebSphere Portal Express installation even though it might be marked as "Optional" on the Select packages to install screen.

   Tip: If you have a WebSphere Portal Express Enable license, you must select both the **IBM WebSphere Portal Express** and **IBM WebSphere Portal Enable** packages.
8. On Install Packages: Select the fixes to install screen, select any required fixes. Then, click **Next**.
9. Accept the license agreement and then click **Next**.
10. Enter the directory where you want to store shared resources and then click **Next**.
11. Complete the following steps on the Install Packages: Installation Directory panel:

    Remember: The installation directory that you specify must NOT contain any files or the following characters: `~ ! @ # $ % ^ & * ( ) + { } | < > ? ` = [ ] ; ' , . "` and spaces.
    a. Select the WebSphere Application Server Package Group Name and then enter the installation directory path.
    b. Select the WebSphere Portal Express Package Group Name and then enter the installation directory path.
    c. Click **Next**.
12. Select the translations to install and then click **Next**.

13. Optional: If you chose to install Portal to an existing copy of WebSphere Application Server, select the available copy of WebSphere Application Server, and click **Next**.

    Note: Skip this step, if you are not installing Portal to an existing copy of WebSphere Application Server.
14. On Install Packages: Select the features to install, expand the WebSphere Application Server and WebSphere Portal Express packages to modify the features you want to install and then click **Next**.

    Note: Ensure that **Portal Server Profile** is selected to create a profile that contains the Portal application server and the product binary files. Clear this option if you need a binary only installation for migration.

    Note: As you select the items, read the **Details** section for information.
15. On Profile configuration details, enter the user ID and password for the configuration wizard administrator. Then, click **Next**.
16. If you selected the **Portal Server Profile** package, click **Enter the Administrator user ID and password for the Portal Server**.
17. Confirm the Summary information and then click **Install**.
18. After the Installation Manager finishes creating the response file, click **Finish** and then close the Installation Manager to complete the response file recording.
19. If you plan to install on a different computer, copy the response file to the response file directory on that computer.

Linux: Installing with the response file:

Use either a sample response file or a response file that you created to install WebSphere Application Server and WebSphere Portal Express. You can create a response file with the graphical or console mode interface.

About this task

Locate the sample response file in the *setup_root*/responsefiles directory. Modify the file with your environment values.

The installation program verifies the operating system and its prerequisites, available disk space, and any required software prerequisites before installation. You cannot install two instances of the server at the same time, even if you are installing to different directories. You must install each server completely before you install the next one.

Procedure

1. If the repository URL requires authentication, use the IBM Installation Manager command-line tool to create a secure **Storage File**.

   The secure **Storage File** stores the credentials that are required for the repositories. The IBM Installation Manager command-line tool **imutilsc** is available from the installation tools directory. The following information is an example of the **imutilsc** key ring file command:

```
./imutilsc saveCredential -url repository_URL -userName credential_userName
 -userPassword password -secureStorageFile storage_file
[ -masterPasswordFile master_password_file ]
[ -preferences com.ibm.cic.common.core.preferences.ssl.nonsecureMode=true|false ]
```

```
[ -proxyHost proxy_host -proxyPort proxy_port
 [ -proxyUsername proxy_username -proxyUserPassword proxyuser_password ]
 [ -useSocks ] ]
[ -verbose ]
```

Tip: If you install on a different computer, copy the secure **Storage File** to that computer.

2. Go to the `InstallationManager_root/eclipse/tools` directory.
3. Run the following task to install WebSphere Portal Express and IBM WebSphere Application Server:

   Tip: Add the **-secureStorageFile** *pathtosecureStorageFile* **-masterPasswordFile** *pathtomasterPasswordFile* parameters to the **imcl** command if you are using a secure **Storage File** to store credentials.
   `./imcl -acceptLicense input` *pathtoresponse.xml* `-log` *dirpath/logfilename*

What to do next

Attention: When you install WebSphere Application Server and WebSphere Portal Express together, you might find the following message in the `SystemErr.log` file:
AppServer_root`/properties/version/installed.xml (No such file or directory)`

This message is part of the installation process and is not repeated after the installation is complete. Therefore, the message can be ignored.

Linux: Converting the evaluation license

If you initially installed IBM WebSphere Portal Express using the evaluation license and then purchased the product, you can convert the installed evaluation license to a production license. You can do this at any time, before or after the evaluation license expires.

Before you begin

Review the evaluation license.

About this task

You do not need to reinstall the program files to convert the evaluation license. However, the license conversion utility must have access to the purchased production software. You cannot use the evaluation software to convert the evaluation license.

Complete the following steps to convert the evaluation license:

Procedure

1. Run the `./stopServer.sh` *WebSphere_Portal* `-username` *admin_userid* `-password` *admin_password* task from the *wp_profile_root*`/bin` directory.
2. If you are using physical media on Linux mount the drive and then insert the CD.
3. Run the `./ConfigEngine.sh install-license -Dlicense=`*media_filepath* task from the *wp_profile_root*`/ConfigEngine` directory. .

   Tip: *media_filepath* is the location of the `PortalExpress` subdirectory on either the WebSphere Portal Express CD or the downloaded installation image, for example `/mnt/cdrom/PortalExpress` if the CD is in `/mnt/cdrom` or `/wpdownload/PortalExpress` for the installation image.

When the task completes, a message confirms that the product is now licensed for use.

4. Run the `./startServer.sh` *`WebSphere_Portal`* task.

Related tasks:

"Linux: Preparing your operating system" on page 182
Prepare the operating system to ensure a successful installation.

"Linux: Preparing the Installation Manager" on page 183
The IBM Installation Manager is used to install installation packages such as IBM WebSphere Portal Express and IBM WebSphere Application Server.

Linux: Next steps

The configuration process has changed. Use the Configuration Wizard to set up your integration with prerequisites, clusters, and more.

Verify that the installation was successful

Unless you selected to install only the binary files, you can log in to the portal and verify that your installation was successful. Access WebSphere Portal Express with the `http://`*`yourserver`*`:`*`yourport`*`/wps/portal` format.

If you are not sure what your port number is, use the **`list-server-ports`** command to determine the port number. Change to the *wp_profile_root*`/ConfigEngine/log`. Run the **`list-server-ports`** task to generate the `WebSphere_Portal_PortMatrix.txt` file. For example: `./ConfigEngine.sh list-server-ports -DWasPassword=`*`password`*.

Change to the `/opt/IBM/WebSphere/AppServer/profiles/cw_profile/ConfigEngine`. Run the **`list-server-ports`** task to generate the `server1_PortMatrix.txt` file. For example: `./ConfigEngine.sh list-server-ports -DWasPassword=`*`password`*.

Modify the `uri.home.substitution` custom property in the Resource Environment Provider

In the **WP ConfigService**, set the **`uri.home.substitution`** custom property to `true` to avoid functional errors that might occur in certain unusual use cases. For more information on **`uri.home.substitution`**, see *Configuration Service* in the related links.

Select a roadmap

If you have not already selected a roadmap to guide you through the installation and deployment process, look at the available roadmaps. The roadmaps are based on typical environments, such as development environment, test environment, and more. They provide a high-level overview of the installation and deployment process. Each roadmap includes a topology diagram, usage recommendations, and instructions. Roadmaps are available for both new deployments and migrations scenarios.

"Roadmaps for installation and deployment" on page 69

"Roadmaps for migration" on page 92

Run the Configuration Wizard to finish the deployment

Then, start the Configuration Wizard and select the option that is defined in your selected roadmap. Use the wizard to run the configuration steps in real time or to generate scripts. The wizard generates instructions and scripts specific to your

environment and the data that you entered. You can save the selection and data that you entered as an XML file. Then, you can load the XML file during a subsequent wizard session to set up a similar configuration on a different server.

If you select to run the configuration in real time, you still have the opportunity to download scripts for selected steps. For example, if only the database administrator can databases, then you can download the database creation script and give it to your database administrator to run.

Default URLs

During the configuration process, you might need to following URLs to access different administration user interfaces.

Use the following default URLs to access IBM WebSphere Portal Express, the WebSphere Integrated Solutions Console, and the Configuration Wizard:

IBM WebSphere Portal Express

```
http://localhost:10039/wps/portal
```

```
https://localhost:10042/wps/portal
```

WebSphere Integrated Solutions Console

```
http://localhost:10038/ibm/console
```

```
https://localhost:10041/ibm/console
```

Configuration Wizard

```
http://localhost:10200/ibm/wizard
```

```
https://localhost:10202/ibm/wizard
```

If you had any processes from other software in the default port range when the installation started, you might have different port numbers than the defaults.

Related concepts:

Chapter 3, “Roadmaps,” on page 69
Review the roadmaps to understand the common deployment patterns that are supported by the configuration wizard.

“Configuration Wizard” on page 232
Use the Configuration Wizard to set up stand-alone servers and new deployments, create clusters, migrate and update to new versions, and add new capabilities to existing deployments.

Related tasks:

“Linux: Preparing your operating system” on page 182
Prepare the operating system to ensure a successful installation.

“Linux: Preparing the Installation Manager” on page 183
The IBM Installation Manager is used to install installation packages such as IBM WebSphere Portal Express and IBM WebSphere Application Server.

“Linux: Running the installation program” on page 184
Use the IBM Installation Manager program to install the software. IBM Installation Manager provides multiple user interfaces. Select the one that you are more comfortable with. You can use the graphical user interface or the console to create a response file for automated installations.

Related reference:

"Configuration Service" on page 297
The portal Configuration Service is responsible for collecting the most essential configuration data of the IBM WebSphere Portal Express engine.

Linux: Upgrading the SDK

Starting with IBM WebSphere Portal Express Combined Cumulative Fix 05, you can change your SDK Java Technology Edition from version 7.0 to version 7.1.

Before you begin

Ensure that IBM WebSphere Application Server and WebSphere Portal Express are installed with SDK version 7.0.

Procedure

1. Start the IBM Installation Manager.
2. Install IBM WebSphere SDK Java Technology Edition version 7.1.
3. Read the following information before you run the **managesdk** task:
   - For stand-alone environments: Stop the profile server (node) before you run the **managesdk** command.
   - For clustered environments:
     – If the profile is a federated node of a deployment manager, ensure that the deployment manager is running before you run the **managesdk** command to update the profile.
     – Stop all the nodes.
     – Ensure that the node agent for each node is started.
     – When you enable the SDK for a node, run the **managesdk** command from the `/bin` directory to which the node belongs. You can also run the command from the `/bin` directory of the profile that contains the node that you want to update.
     – A connection to the deployment manager must exist with a supported connector protocol in the following order of preference:
       - SOAP
       - Inter-Process Communications (IPC)
       - Remote Method Invocation (RMI)

       If the SOAP protocol is enabled, the **managesdk** command uses the SOAP protocol. If the SOAP protocol is not enabled but the IPC protocol is enabled, the command uses the IPC protocol. If the SOAP and IPC protocol are not enabled, then the command uses the RMI protocol.
     – You must provide the administrative user name and password with the **managesdk** command for each profile that contains a federated node or deployment manager node in a cell with security enabled. If you do not specify the **-user** and **-password** parameters, the **managesdk** command might fail or stop processing.
     – When you enable the SDK for a deployment manager, only the deployment manager server is enabled. None of the managed nodes of the deployment manager are enabled to use the specific SDK.
4. Open a command prompt and change to the *AppServer_root*`/bin` directory.
5. Run the following command to enable all existing profiles to use the new SDK version:

   Go to managesdk command for information about the **managesdk** commands.

- Linux: `./managesdk.sh -enableProfileAll -sdkname 1.7.1_64 -enableServers`

6. Run the following commands to make the new SDK version the new default:
   - Linux: `./managesdk.sh -setCommandDefault -sdkname 1.7.1_64`

     `./managesdk.sh -setNewProfileDefault -sdkname 1.7.1_64`
7. Repeat these steps on each node in your environment.

Related tasks:

"Linux: Preparing your operating system" on page 182
Prepare the operating system to ensure a successful installation.

"Linux: Preparing the Installation Manager" on page 183
The IBM Installation Manager is used to install installation packages such as IBM WebSphere Portal Express and IBM WebSphere Application Server.

"Linux: Running the installation program" on page 184
Use the IBM Installation Manager program to install the software. IBM Installation Manager provides multiple user interfaces. Select the one that you are more comfortable with. You can use the graphical user interface or the console to create a response file for automated installations.

"Linux: Converting the evaluation license" on page 192
If you initially installed IBM WebSphere Portal Express using the evaluation license and then purchased the product, you can convert the installed evaluation license to a production license. You can do this at any time, before or after the evaluation license expires.

Windows: Installing WebSphere Portal Express and Web Content Manager

Installing the digital experience software includes preparing your operating system and using IBM Installation Manager to install IBM WebSphere Portal Express and Web Content Manager. Then, use the Configuration Wizard to complete the deployment configuration.

Before you begin

Roadmaps for installation and deployment are included in this product documentation to guide you through the process. Roadmaps provide a high-level overview of the process from beginning to end. Select the configuration that is closest to the configuration that you need.

1. "Windows: Preparing your operating system" on page 197
   Prepare the operating system to ensure a successful installation.
2. "Windows: Preparing the Installation Manager" on page 197
   The IBM Installation Manager is used to install installation packages such as IBM WebSphere Portal Express and IBM WebSphere Application Server.
3. "Windows: Running the installation program" on page 198
   Use the IBM Installation Manager program to install the software. IBM Installation Manager provides multiple user interfaces. Select the one that you are more comfortable with. You can use the graphical user interface or the console to create a response file for automated installations.
4. "Windows: Converting the evaluation license" on page 206
   If you initially installed IBM WebSphere Portal Express using the evaluation license and then purchased the product, you can convert the installed evaluation license to a production license. You can do this at any time, before or after the evaluation license expires.

5. "Windows: Next steps" on page 207
   The configuration process has changed. Use the Configuration Wizard to set up your integration with prerequisites, clusters, and more.
6. "Windows: Upgrading the SDK" on page 209
   Starting with IBM WebSphere Portal Express Combined Cumulative Fix 05, you can change your SDK Java Technology Edition from version 7.0 to version 7.1.

Windows: Preparing your operating system

Prepare the operating system to ensure a successful installation.

Procedure

1. Check that the system logon user ID you plan to use during installation has the following permissions and rights:
   - The user ID must exist before the installation.
   - The user ID must belong to the Windows local Administrators group.
2. Determine whether a user account is a member of the Administrators group:
   a. Click **Start** > **Programs** > **Administrative Tools** > **Computer Management**.
   b. Expand **Local Users and Groups** and select **Groups**.
   c. Open the **Administrators** group to see what members belong to it.
   d. Add the user to the Administrators group if necessary.
3. Consider the following recommendations when you install to avoid excessively long path names:

   Note: If you exceed the 259 maximum character length, you might receive one of the following error messages during configuration or in the IBM Installation Manager log files:
   - `The input line is too long.`
   - `The syntax of the command is incorrect.`
   - `The file name is too long.`

   a. Use a short installation path. For example, use `C:\WebSphere` instead of `C:\Program Files\IBM\WebSphere`
   b. Specify node names; do not use names longer than 5 characters. For example, you might use `node1` instead of `longnodename01`.
   c. Name WAR files with less than 21 characters. If necessary, modify the file name before you install.

Windows: Preparing the Installation Manager

The IBM Installation Manager is used to install installation packages such as IBM WebSphere Portal Express and IBM WebSphere Application Server.

Procedure

If you have an existing Installation Manager, start it and go to **File** > **Preferences**. Then, click **Updates**. Click the **Search for Installation Manager updates** check box. This box enables the Installation Manager to search for updates the next time you run an installation or update.
If you do not have an existing Installation Manager, then complete the following procedure:

1. Start all servers and applications that require a port number to avoid port conflicts when installing WebSphere Portal Express.
2. Type **`ping`** `yourserver.yourcompany.com` on a command line to verify that your fully qualified host name is properly configured.

3. Type **ping** `localhost` on a command line to verify that your network is properly configured.
4. If you are installing on a server with a firewall, antivirus, screen saver, or desktop search engine that is enabled, disable them before you install. If you do not disable them and the installation program detects them, a warning message displays during the installation.
5. Run the following task from the `IIM` directory: `install.bat`
6. Run the following task from the Portal Setup disk if you want to start the launchpad to complete all necessary installation steps: `setup64.exe` Right-click on **setup64.exe**. Then, select **Run as administrator** to start.

   Launchpad tip: On the Launchpad: Install Portal panel, you have the following options:

   Install IBM WebSphere Portal from media
   Choose this option if you are installing directly from the DVD.

   Install IBM WebSphere Portal from network
   Choose this option if you downloaded the electronic images.

   Install IBM Installation Manager only
   Choose this option if you want to install multiple IBM products, or if you want to install WebSphere Portal Express directly from IBM Passport Advantage, and a previous version of Installation Manager is not already installed on your system.

   Restriction: If you have a previously installed 32-bit version of Installation Manager, then you cannot proceed with this option. You must manually install the 64-bit version of Installation Manager that is specific to your operating system using the Setup disc before you proceed with the WebSphere Portal Express installation.

   Tip: Add the `LaunchPadLocale` *language_code* to the **setup** task to change the display to your user locale or to another language.

   Note: If the language is not currently supported for the user interface, you might see the English version. For details on supported languages and the language codes for all of the WebSphere Portal Express user interfaces, see "Supported languages" on page 3622.

   After you complete all necessary steps from the launchpad, verify that your installation was successful.

Windows: Running the installation program

Use the IBM Installation Manager program to install the software. IBM Installation Manager provides multiple user interfaces. Select the one that you are more comfortable with. You can use the graphical user interface or the console to create a response file for automated installations.

"Windows: Installing with the graphical user interface" on page 199
Use the IBM Installation Manager to install IBM WebSphere Portal Express, IBM WebSphere Application Server, and the Apache Derby database.

"Windows: Installing with the console" on page 201
Use the IBM Installation Manager to install IBM WebSphere Portal Express with an existing IBM WebSphere Application Server.

"Windows: Creating a response file" on page 203
After IBM Installation Manager is installed, you can use it to record a response file that is based on your environment. Record a response file on the same

operating system you plan for the installation. If you have multiple operating systems, you must record a response file for each operating system. Use a response file to automate your installation on multiple servers.

"Windows: Installing with the response file" on page 205
Use either a sample response file or a response file that you created to install WebSphere Application Server and WebSphere Portal Express. You can create a response file with the graphical or console mode interface.

Related tasks:

"Windows: Preparing your operating system" on page 197
Prepare the operating system to ensure a successful installation.

Windows: Installing with the graphical user interface:

Use the IBM Installation Manager to install IBM WebSphere Portal Express, IBM WebSphere Application Server, and the Apache Derby database.

About this task

The installation program verifies the operating system and its prerequisites, available disk space, and any required software prerequisites before installation. You cannot install two instances of the server at the same time, even if you are installing to different directories. You must install each server completely before you install the next one.

Procedure

1. Start all servers and applications that require a port number to avoid port conflicts when you install WebSphere Portal Express. If you are installing multiple copies of IBM WebSphere Portal Express on your server, start the existing Configuration Wizard servers.
2. If necessary, start the Installation Manager.
3. After you install or upgrade the Installation Manager, complete the following steps to add the repositories where the installation media exists:
   a. Open the IBM Installation Manager and go to **File** > **Preferences** > **Repositories**.
   b. Select **Add Repositories**.
   c. Select **Browse** and go to the *Portal-install-eimage*/Setup/repository.config file and then click **OK**.
   d. Ensure that all required repositories are checked. Then, click **Test Connections** to ensure that the IBM Installation Manager can successfully access the directory where the service repositories are stored.
   e. Select **Apply**.
   f. Select **OK**.
4. On the main IBM Installation Manager panel, select **Install**.
5. Optional: If you are installing Portal to an existing copy of WebSphere Application Server, then select only **IBM WebSphere Portal Server** on the Install Packages screen. To complete this installation option, your existing copy of WebSphere Application Server must meet the following requirements:
   - WebSphere Application Server Version 8.5.5.2 or later
   - Java Version 7
   - No existing Portal profile

   Skip the following step, if you choose to install to an existing copy of WebSphere Application Server.

6. On Install Packages: Select packages to install, select the following packages, and then click **Next**:
   - **IBM WebSphere Application Server Network Deployment**
   - **IBM WebSphere Portal Server**
   - **IBM WebSphere SDK Java Technology Edition**

     Note: The **IBM WebSphere SDK Java Technology Edition** option is required for a WebSphere Portal Express installation even though it might be marked as "Optional" on the Select packages to install screen.

   Tip: If you have a WebSphere Portal Express Enable license, you must select both the **IBM WebSphere Portal Express** and **IBM WebSphere Portal Enable** packages.
7. On Install Packages: Select the fixes to install screen, select any required fixes. Then, click **Next**.
8. Accept the license agreement and then click **Next**.
9. Enter the directory where you want to store shared resources and then click **Next**.
10. Complete the following steps on the Install Packages: Installation Directory panel:

    Remember: The installation directory that you specify must NOT contain any files or the following characters: `~ ! @ # $ % ^ & * ( ) + { } | < > ? ` = [ ] ; ' , . "` and spaces.
    a. Select the WebSphere Application Server Package Group Name and then enter the installation directory path.
    b. Select the WebSphere Portal Express Package Group Name and then enter the installation directory path.
    c. Click **Next**.
11. Select the translations to install and then click **Next**.
12. Optional: If you chose to install Portal to an existing copy of WebSphere Application Server, select the available copy of WebSphere Application Server, and click **Next**.

    Note: Skip this step, if you are not installing Portal to an existing copy of WebSphere Application Server.
13. On Install Packages: Select the features to install, expand the WebSphere Application Server and WebSphere Portal Express packages to modify the features you want to install and then click **Next**.

    Note: Ensure that **Portal Server Profile** is selected to create a profile that contains the Portal application server and the product binary files. Clear this option if you need a binary only installation for migration.

    Note: As you select the items, read the **Details** section for information.
14. On Profile configuration details, enter the user ID and password for the configuration wizard administrator. Then, click **Next**.
15. If you selected the **Portal Server Profile** package, click **Enter the Administrator user ID and password for the Portal Server**.
16. Confirm the Summary information and then click **Install**.

What to do next

After a successful installation, the summary displays. Choose the **Portal First Steps** radio button and then click **Finish** to start the servers and begin configuring WebSphere Portal Express with the Configuration Wizard.

Tip: To access First Steps later, you can either select **First Steps** from the **Start** menu or you can run the **firststeps.exe** task from the *wp_profile_root*/PortalServer/installer/wp.firststeps directory. Add the LaunchPadLocale *language_code* to the **firststeps** task to change the display to your user locale or to another language.

Attention: When you install WebSphere Application Server and WebSphere Portal Express together, you might find the following message in the SystemErr.log file:
AppServer_root/properties/version/installed.xml (No such file or directory)

This message is part of the installation process and is not repeated after the installation is complete. Therefore, the message can be ignored.

Windows: Installing with the console:

Use the IBM Installation Manager to install IBM WebSphere Portal Express with an existing IBM WebSphere Application Server.

Before you begin

A working installation of IBM WebSphere Application Server is required. Ensure that it is not used by another copy of WebSphere Portal Express. Before you install the WebSphere Portal Express package, install **IBM WebSphere SDK Java Technology Edition** into the same directory where IBM WebSphere Application Server is installed.

About this task

The installation program verifies the operating system and its prerequisites, available disk space, and any required software prerequisites before installation. You cannot install two instances of the server at the same time, even if you are installing to different directories. You must install each server completely before you install the next one.

Procedure

1. Start all servers and applications that require a port number to avoid port conflicts when you install WebSphere Portal Express. If you are installing multiple copies of IBM WebSphere Portal Express on your server, start the existing Configuration Wizard servers.
2. Complete the following steps to run the program as an administrator:
   a. Go to the InstallationManager_root/eclipse/tools directory.
   b. Right-click on the **imcl** command.
   c. Select **Properties**.
   d. Go to the **Compatibility** tab.
   e. Select **Run this program as an administrator**.
   f. Click **OK**.
3. Open a command prompt and change to the InstallationManager_root/eclipse/tools directory.

4. Run the command to start the IBM Installation Manager in console mode: `imcl -c`
5. Complete the following steps to add the repositories:
   a. Enter `P` to go to the **Preferences** menu.
   b. Enter `1` to go to the **Repositories** menu.
   c. Enter `D` to add repositories.
   d. Type the path for your WebSphere Portal Express repository file.
   e. Enter `A` to apply your repositories and return to the **Preferences** menu.
   f. Enter `R` to return to the **Main** menu.
6. Enter `1` to install the software packages.
7. Select the software packages that you want to install.
   - If you are installing Portal to an existing copy of WebSphere Application Server, then select only **IBM WebSphere Portal Server.** To complete this installation option, your existing copy of WebSphere Application Server must meet the following requirements:
     – WebSphere Application Server Version 8.5.5.2 or later
     – Java Version 7
     – No existing Portal profile
   - If you are not installing Portal to an existing copy of WebSphere Application Server, then select the following packages:
     – **IBM WebSphere Application Server Network Deployment**
     – **IBM WebSphere Portal Server**
     – **IBM WebSphere SDK Java Technology Edition**

     Note: The **IBM WebSphere SDK Java Technology Edition** option is required for a WebSphere Portal Express installation even though it might be marked as "Optional."

   Tip: If you have a WebSphere Portal Express Enable license, you must select both the **IBM WebSphere Portal Express** and **IBM WebSphere Portal Enable** packages.
8. Choose one of the following options:
   - Enter `1` to choose version 8.5.0.0 for installation.
   - Enter `2` to show all available versions of the package.
9. Enter `N`.
10. Enter `A` to accept the license agreement.
11. Enter `N`.
12. Choose one of the following options for translation packages:
    - Enter `N` to select the default English package only.
    - Enter the number for the translation package you want to install.
13. On the **Incompatible package group** menu, choose one of the following options:
    - Enter `M` to change the installation directory. Then, enter the new installation directory.
    - Enter `N` to keep the existing directory.
14. Enter the number for the WebSphere Application Server root directory to use as the existing WebSphere Application Server.

15. On the **IBM Installation Manager** > **Install** > **Licenses** > **Location** > **Features** menu, enter `1` to select the **Portal Server Profile** feature. Then, enter `N` to continue.

    Note: Ensure that **Portal Server Profile** is selected to create a profile that contains the Portal application server and the product binary files. Clear this option if you need a binary only installation for migration.
16. Enter the configuration wizard administrator user ID and password.
17. If you selected the **Portal Server Profile** feature, enter the information for the following prompts:
    - **Enter the host name**
    - **Enter the node name**
    - **Enter the cell name**
    - **Enter an administrator user ID for the portal server**
    - **Enter an administrator user password for the portal server**
    - **Confirm administrator user password for the portal server**
18. Enter `N`.
19. Review the summary information.
20. Choose one of the following options:
    - Enter `G` to generate a response file.
    - Enter `I` to install WebSphere Portal Express.
21. When the installation is complete, enter `F` to return to the main installation menu.
22. Access the Configuration Wizard. Go to `http://`*`your_server`*`:`*`10200`*`/ibm/wizard`.
23. Log in to the Configuration Wizard with the administrative ID for the configuration wizard profile, `cw_profile`.

    Note: The wizard user interface might not be available in all languages. If the language is not currently supported, you might see the English version. For details on the supported languages for all of the WebSphere Portal Express user interfaces, see "Supported languages" on page 3622.

Windows: Creating a response file:

After IBM Installation Manager is installed, you can use it to record a response file that is based on your environment. Record a response file on the same operating system you plan for the installation. If you have multiple operating systems, you must record a response file for each operating system. Use a response file to automate your installation on multiple servers.

About this task

You can also use the console mode to generate a response file. After you review the summary information, enter `G` to generate a response file instead of `I` to install.

Procedure

1. Start all servers and applications that require a port number to avoid port conflicts when you install WebSphere Portal Express. If you are installing multiple copies of IBM WebSphere Portal Express on your server, start the existing Configuration Wizard servers.
2. Go to the `InstallationManager_root/eclipse` directory.

3. Run the following task to start the recording: `IBMIM.exe -record` *`pathtoresponsexmlfile`* `-skipInstall` *`tempinstalldirectory`*

   `-record`
   This parameter indicates recording the actions and parameters into the response file. The *pathtoresponsexmlfile* text is the name of the response file, for example `Wp8SampleResp.xml`. The *tempinstalldirectory* text is the directory where the response file is recorded.

   `-skipInstall`
   This parameter indicates that no actual installation is completed even though it leads to the final pane.

   tempinstalldirectory
   This value is a directory where the installation saves the history and data when you record the response files.

4. Complete the following steps to add the repositories where the installation media exists:
   a. Open the IBM Installation Manager and go to **File** > **Preferences** > **Repositories**.
   b. Select **Add Repositories**.
   c. Select **Browse** and go to the *`Portal-install-eimage`*`/Setup/repository.config` file and then click **OK**.
   d. Ensure that all required repositories are checked. Then, click **Test Connections** to ensure that the IBM Installation Manager can successfully access the directory where the service repositories are stored.
   e. Select **Apply**.
   f. Select **OK**.
5. On the main IBM Installation Manager panel, select **Install**.
6. Optional: If you are installing Portal to an existing copy of WebSphere Application Server, then select only **IBM WebSphere Portal Server** on the Install Packages screen. To complete this installation option, your existing copy of WebSphere Application Server must meet the following requirements:
   - WebSphere Application Server Version 8.5.5.2 or later
   - Java Version 7
   - No existing Portal profile

   Skip the following step, if you choose to install to an existing copy of WebSphere Application Server.
7. On Install Packages: Select packages to install, select the following packages, and then click **Next**:
   - **IBM WebSphere Application Server Network Deployment**
   - **IBM WebSphere Portal Server**
   - **IBM WebSphere SDK Java Technology Edition**

     Note: The **IBM WebSphere SDK Java Technology Edition** option is required for a WebSphere Portal Express installation even though it might be marked as "Optional" on the Select packages to install screen.

   Tip: If you have a WebSphere Portal Express Enable license, you must select both the **IBM WebSphere Portal Express** and **IBM WebSphere Portal Enable** packages.
8. On Install Packages: Select the fixes to install screen, select any required fixes. Then, click **Next**.

9. Accept the license agreement and then click **Next**.
10. Enter the directory where you want to store shared resources and then click **Next**.
11. Complete the following steps on the Install Packages: Installation Directory panel:

    Remember: The installation directory that you specify must NOT contain any files or the following characters: `~ ! @ # $ % ^ & * ( ) + { } | < > ? ` = [ ] ; ' , . "` and spaces.

    a. Select the WebSphere Application Server Package Group Name and then enter the installation directory path.
    b. Select the WebSphere Portal Express Package Group Name and then enter the installation directory path.
    c. Click **Next**.
12. Select the translations to install and then click **Next**.
13. Optional: If you chose to install Portal to an existing copy of WebSphere Application Server, select the available copy of WebSphere Application Server, and click **Next**.

    Note: Skip this step, if you are not installing Portal to an existing copy of WebSphere Application Server.
14. On Install Packages: Select the features to install, expand the WebSphere Application Server and WebSphere Portal Express packages to modify the features you want to install and then click **Next**.

    Note: Ensure that **Portal Server Profile** is selected to create a profile that contains the Portal application server and the product binary files. Clear this option if you need a binary only installation for migration.

    Note: As you select the items, read the **Details** section for information.
15. On Profile configuration details, enter the user ID and password for the configuration wizard administrator. Then, click **Next**.
16. If you selected the **Portal Server Profile** package, click **Enter the Administrator user ID and password for the Portal Server**.
17. Confirm the Summary information and then click **Install**.
18. After the Installation Manager finishes creating the response file, click **Finish** and then close the Installation Manager to complete the response file recording.
19. If you plan to install on a different computer, copy the response file to the response file directory on that computer.

Windows: Installing with the response file:

Use either a sample response file or a response file that you created to install WebSphere Application Server and WebSphere Portal Express. You can create a response file with the graphical or console mode interface.

About this task

Locate the sample response file in the *setup_root*`/responsefiles` directory. Modify the file with your environment values.

The installation program verifies the operating system and its prerequisites, available disk space, and any required software prerequisites before installation. You cannot install two instances of the server at the same time, even if you are installing to different directories. You must install each server completely before you install the next one.

Procedure

1. If the repository URL requires authentication, use the IBM Installation Manager command-line tool to create a secure **Storage File**.

   The secure **Storage File** stores the credentials that are required for the repositories. The IBM Installation Manager command-line tool **imutilsc** is available from the installation tools directory. The following information is an example of the **imutilsc** key ring file command:

   ```
   imutilsc saveCredential -url repository_URL -userName credential_userName
    -userPassword password -secureStorageFile storage_file
   [ -masterPasswordFile master_password_file ]
   [ -preferences com.ibm.cic.common.core.preferences.ssl.nonsecureMode=true|false ]
    [ -proxyHost proxy_host -proxyPort proxy_port
     [ -proxyUsername proxy_username -proxyUserPassword proxyuser_password ]
     [ -useSocks ] ]
    [ -verbose ]
   ```

   Tip: If you install on a different computer, copy the secure **Storage File** to that computer.
2. Go to the `InstallationManager_root/eclipse/tools` directory.
3. Run the following task to install WebSphere Portal Express and IBM WebSphere Application Server:

   Tip: Add the **-secureStorageFile** *pathtosecureStorageFile* **-masterPasswordFile** *pathtomasterPasswordFile* parameters to the **imcl** command if you are using a secure **Storage File** to store credentials.
   `imcl -acceptLicense input` *pathtoresponse.xml* `-log` *dirpath/logfilename*

What to do next

Attention: When you install WebSphere Application Server and WebSphere Portal Express together, you might find the following message in the `SystemErr.log` file:
AppServer_root`/properties/version/installed.xml (No such file or directory)`

This message is part of the installation process and is not repeated after the installation is complete. Therefore, the message can be ignored.

Windows: Converting the evaluation license

If you initially installed IBM WebSphere Portal Express using the evaluation license and then purchased the product, you can convert the installed evaluation license to a production license. You can do this at any time, before or after the evaluation license expires.

Before you begin

Review the evaluation license.

About this task

You do not need to reinstall the program files to convert the evaluation license. However, the license conversion utility must have access to the purchased production software. You cannot use the evaluation software to convert the evaluation license.

Complete the following steps to convert the evaluation license:

Procedure

1. Run the `stopServer.bat` *`WebSphere_Portal`* `-username` *`admin_userid`* `-password` *`admin_password`* task from the *wp_profile_root*`\bin` directory.
2. Run the `ConfigEngine.bat install-license -Dlicense=`*`media_filepath`* task from the *wp_profile_root*`\ConfigEngine` directory.

   Tip: *media_filepath* is the location of the `PortalExpress` subdirectory on either the WebSphere Portal Express CD or the downloaded installation image, for example `d:\PortalExpress` if the CD is in drive D or `c:\wpdownload\PortalExpress` for the installation image.

   When the task completes, a message confirms that the product is now licensed for use.
3. Run the `startServer.bat` *`WebSphere_Portal`* task.

Related tasks:

"Windows: Preparing your operating system" on page 197
Prepare the operating system to ensure a successful installation.

"Windows: Preparing the Installation Manager" on page 197
The IBM Installation Manager is used to install installation packages such as IBM WebSphere Portal Express and IBM WebSphere Application Server.

Windows: Next steps

The configuration process has changed. Use the Configuration Wizard to set up your integration with prerequisites, clusters, and more.

Verify that the installation was successful

Unless you selected to install only the binary files, you can log in to the portal and verify that your installation was successful. Access WebSphere Portal Express with the `http://`*`yourserver`*`:`*`yourport`*`/wps/portal` format.

If you are not sure what your port number is, use the **`list-server-ports`** command to determine the port number. Change to the *wp_profile_root*`\ConfigEngine\log\`. Run the **`list-server-ports`** task to generate the `WebSphere_Portal_PortMatrix.txt` file. For example: `ConfigEngine.bat list-server-ports -DWasPassword=`*`password`*.

Change to the `C:\IBM\WebSphere\AppServer\profiles\cw_profile\ConfigEngine`. Run the **`list-server-ports`** task to generate the `server1_PortMatrix.txt` file. For example: `ConfigEngine.bat list-server-ports -DWasPassword=`*`password`*.

Modify the `uri.home.substitution` custom property in the Resource Environment Provider

In the **WP ConfigService**, set the **`uri.home.substitution`** custom property to `true` to avoid functional errors that might occur in certain unusual use cases. For more

information on **uri.home.substitution**, see *Configuration Service* in the related links.

Select a roadmap

If you have not already selected a roadmap to guide you through the installation and deployment process, look at the available roadmaps. The roadmaps are based on typical environments, such as development environment, test environment, and more. They provide a high-level overview of the installation and deployment process. Each roadmap includes a topology diagram, usage recommendations, and instructions. Roadmaps are available for both new deployments and migrations scenarios.

"Roadmaps for installation and deployment" on page 69

"Roadmaps for migration" on page 92

Run the Configuration Wizard to finish the deployment

Then, start the Configuration Wizard and select the option that is defined in your selected roadmap. Use the wizard to run the configuration steps in real time or to generate scripts. The wizard generates instructions and scripts specific to your environment and the data that you entered. You can save the selection and data that you entered as an XML file. Then, you can load the XML file during a subsequent wizard session to set up a similar configuration on a different server.

If you select to run the configuration in real time, you still have the opportunity to download scripts for selected steps. For example, if only the database administrator can databases, then you can download the database creation script and give it to your database administrator to run.

Default URLs

During the configuration process, you might need to following URLs to access different administration user interfaces.

Use the following default URLs to access IBM WebSphere Portal Express, the WebSphere Integrated Solutions Console, and the Configuration Wizard:

IBM WebSphere Portal Express

```
http://localhost:10039/wps/portal
```

```
https://localhost:10042/wps/portal
```

WebSphere Integrated Solutions Console

```
http://localhost:10038/ibm/console
```

```
https://localhost:10041/ibm/console
```

Configuration Wizard

```
http://localhost:10200/ibm/wizard
```

```
https://localhost:10202/ibm/wizard
```

If you had any processes from other software in the default port range when the installation started, you might have different port numbers than the defaults.

Related concepts:

Chapter 3, “Roadmaps,” on page 69
Review the roadmaps to understand the common deployment patterns that are supported by the configuration wizard.

“Configuration Wizard” on page 232
Use the Configuration Wizard to set up stand-alone servers and new deployments, create clusters, migrate and update to new versions, and add new capabilities to existing deployments.

Related tasks:

“Windows: Preparing your operating system” on page 197
Prepare the operating system to ensure a successful installation.

“Windows: Preparing the Installation Manager” on page 197
The IBM Installation Manager is used to install installation packages such as IBM WebSphere Portal Express and IBM WebSphere Application Server.

“Windows: Running the installation program” on page 198
Use the IBM Installation Manager program to install the software. IBM Installation Manager provides multiple user interfaces. Select the one that you are more comfortable with. You can use the graphical user interface or the console to create a response file for automated installations.

Related reference:

“Configuration Service” on page 297
The portal Configuration Service is responsible for collecting the most essential configuration data of the IBM WebSphere Portal Express engine.

Windows: Upgrading the SDK

Starting with IBM WebSphere Portal Express Combined Cumulative Fix 05, you can change your SDK Java Technology Edition from version 7.0 to version 7.1.

Before you begin

Ensure that IBM WebSphere Application Server and WebSphere Portal Express are installed with SDK version 7.0.

Procedure

1. Start the IBM Installation Manager.
2. Install IBM WebSphere SDK Java Technology Edition version 7.1.
3. Read the following information before you run the **managesdk** task:
   - For stand-alone environments: Stop the profile server (node) before you run the **managesdk** command.
   - For clustered environments:
     - If the profile is a federated node of a deployment manager, ensure that the deployment manager is running before you run the **managesdk** command to update the profile.
     - Stop all the nodes.
     - Ensure that the node agent for each node is started.
     - When you enable the SDK for a node, run the **managesdk** command from the `/bin` directory to which the node belongs. You can also run the command from the `/bin` directory of the profile that contains the node that you want to update.
     - A connection to the deployment manager must exist with a supported connector protocol in the following order of preference:
       - SOAP

- Inter-Process Communications (IPC)
- Remote Method Invocation (RMI)

If the SOAP protocol is enabled, the **managesdk** command uses the SOAP protocol. If the SOAP protocol is not enabled but the IPC protocol is enabled, the command uses the IPC protocol. If the SOAP and IPC protocol are not enabled, then the command uses the RMI protocol.

– You must provide the administrative user name and password with the **managesdk** command for each profile that contains a federated node or deployment manager node in a cell with security enabled. If you do not specify the **-user** and **-password** parameters, the **managesdk** command might fail or stop processing.
– When you enable the SDK for a deployment manager, only the deployment manager server is enabled. None of the managed nodes of the deployment manager are enabled to use the specific SDK.

4. Open a command prompt and change to the *AppServer_root*/bin directory.
5. Run the following command to enable all existing profiles to use the new SDK version:

   Go to managesdk command for information about the **managesdk** commands.

   • Windows: `managesdk.bat -enableProfileAll -sdkname 1.7.1_64 -enableServers`
6. Run the following commands to make the new SDK version the new default:

   • Windows: `managesdk.bat -setCommandDefault -sdkname 1.7.1_64`

   `managesdk.bat -setNewProfileDefault -sdkname 1.7.1_64`
7. Repeat these steps on each node in your environment.

Related tasks:

"Windows: Preparing your operating system" on page 197
Prepare the operating system to ensure a successful installation.

"Windows: Preparing the Installation Manager" on page 197
The IBM Installation Manager is used to install installation packages such as IBM WebSphere Portal Express and IBM WebSphere Application Server.

"Windows: Running the installation program" on page 198
Use the IBM Installation Manager program to install the software. IBM Installation Manager provides multiple user interfaces. Select the one that you are more comfortable with. You can use the graphical user interface or the console to create a response file for automated installations.

"Windows: Converting the evaluation license" on page 206
If you initially installed IBM WebSphere Portal Express using the evaluation license and then purchased the product, you can convert the installed evaluation license to a production license. You can do this at any time, before or after the evaluation license expires.

Installing add-ons

You can use the Solution Installer through the Configuration Wizard to install and uninstall add-ons to an IBM WebSphere Portal Express server instance. The Solution Installer uses the Portal Application Archive (PAA) format as the standard format for application distribution. Portal Application Archive (PAA) updates are not supported in the configuration wizard currently. For more information about updating add-ons using a command prompt, see the Managing your existing PAA file section.

About this task

Portal Administrators can use the Solution Installer through the Configuration Wizard to install existing PAA-formatted applications to the WebSphere Portal Express server instance. Portal Solution Developers can use the Solution Installer through the Configuration Wizard to install PAA-formatted applications that they develop for their company.

Complete the following tasks to install, uninstall, and update add-ons.

"Install and uninstall add-ons using the Configuration Wizard" on page 212
You can install add-on functionality to your WebSphere Portal Express with the solution installer through the Configuration Wizard. The add-ons that are accepted by the configuration options are `.paa` files. For more information, see the solution installer documentation.

"Migrating PAA content" on page 213
If you installed a Portal Application Archive (PAA) file on a previous version of IBM WebSphere Portal Express, you can migrate the content to the current version of WebSphere Portal Express.

"Managing your existing Portal Application Archive (PAA) file" on page 214
When you update an existing Portal Application Archive (PAA) file, it is not as simple as overwriting the file with the new content. There might be large differences between the content in both files. Therefore, to manage your PAA file, you can upgrade to the latest version and rollback to the previous version.

"Solution Installer run time configuration" on page 217
The Solution Installer run time configuration properties allows you to control how the Solution Installer installs a Portal Application Archive (PAA) formatting application to IBM WebSphere Portal Express.

Related tasks:

"Planning to install WebSphere Portal Express" on page 101
Before you install IBM WebSphere Portal Express in a production environment, you need to assess your hardware and software needs, possible database configurations, security options, and LDAP server options. Skipping this important step can lead to unexpected results and costly delays.

"Installing and preparing the prerequisite software" on page 148
Before you install the digital experience software, make sure that the prerequisite software is installed and configured. Depending on your environment, you might already have the prerequisites, such as a database server and user registry. Verify that the prerequisite software is the correct version, has the required fix packs applied, and is configured to work with the digital experience software.

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

Related reference:

"Getting the software" on page 135
There are different ways for you to get WebSphere Portal Express and Web Content Manager Version 8.5 software.

Install and uninstall add-ons using the Configuration Wizard

You can install add-on functionality to your WebSphere Portal Express with the solution installer through the Configuration Wizard. The add-ons that are accepted by the configuration options are `.paa` files. For more information, see the solution installer documentation.

Complete the following steps to use the Configuration Wizard to install or uninstall the add-ons:

1. Access the Configuration Wizard. Go to `http://your_server:10200/ibm/wizard`.
2. Log in to the Configuration Wizard with the administrative ID for the configuration wizard profile, `cw_profile`.
3. Click **Add On New Capability**
4. To interact with the solution installer through the configuration wizard, select **Install Add-ons** or **Uninstall Add-ons**.
5. Provide information about your environment.
6. Save your wizard selections to use the same values that you entered later.
7. Choose one of the following options:
   - Click **Download Files** to run the steps remotely.
   - Click **Run All Steps** to run the steps locally.

Typical versus advanced installation scenarios

Two different installation scenarios are supported through the **Add On New Capability** options in the Configuration Wizard. These scenarios are the typical and advanced installation scenarios.

Typical

The typical scenario is for simple `paa` files that often contain only one component, but is not limited to it, and can be installed directly without any additional configuration. The typical scenario installs/registers the `paa` file with the configEngine and then automatically deploys the content to WebSphere Portal Express.

Advanced

The advanced scenario provides you with much greater flexibility in how to install the `paa` file. If you select **Advanced Install** on the Install type screen, when the workflow completes, the `paa` file is installed to the ConfigEngine, but it is not deployed to portal. Instead, a new configuration option specific to that `paa` file is generated. Use the generated instructions to configure the deployment settings. Finally, a **Launch configuration** button displays so that you can continue with the deployment. After you click **Launch configuration**, follow the steps to ensure that the `paa` file is deployed.

You can select the components that you want to install, if there is more than one component from the information that is presented in this additional configuration option. This step is useful where demonstration content is included in the `paa` file. Or, if one of the components is designated as virtual portal content only.

In addition, if there are IBM Web Content Manager libraries that are found in the `paa` file that exist on the server, you can select to replace them from the libraries in the `paa` file. A list of virtual portals that are available at the time of the creation of the configuration option are also presented so that you can choose where the selected content is deployed.

In the advanced scenario, the additional configuration that is called **Deploy-Remove-PAA-*assemblyName*** is also the mechanism in which content is removed from the portal. Depending on what is required, you can select the type of function from the configuration option.

The **Deploy-Remove_PAA-*.*** option is added to the **Add-on new capability** section of the configuration wizard. After the configuration option is added to the configuration wizard, you can add modifications to thepaa file. When you uninstall by using the **Uninstall Add-ons** with the **Advanced** setting the **Deploy-Remove-PAA-*.*** option is removed from the configuration wizard repository. However, the option still displays in the list of active configuration options under the **Add On New Capability section** of the wizard. To remove these items, you need to restart server1 in the `cw_profile`. Restarting refreshes the repository and ensures that the items no longer show up.

Note: If you install and deploy a `paa` file by using the advanced scenario, you must also remove the content that uses the `paa`-specific option before you uninstall. You can remove this content by using the advanced selection during the **Uninstall add-ons** option.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

Migrating PAA content

If you installed a Portal Application Archive (PAA) file on a previous version of IBM WebSphere Portal Express, you can migrate the content to the current version of WebSphere Portal Express.

About this task

Migration is supported from WebSphere Portal Express Version 7.0 to 8.5 and from Version 8.0 to 8.5.

Procedure

1. If you are migrating from Version 7.0 to Version 8.5, copy the PAA content from the directory of the previous release to the directory in the current release.

   Note: If you are migrating from Version 8.0.0.1 to Version 8.5, you can skip this step.

   Tip: Starting with version 8.0, the PAA directory is in the *wp_profile_root*`/paa` directory.

   Remember: Select only the expanded PAA file directories that you want to migrate to the new release.
2. Open a command prompt and change to the following directory:
   - IBM i: *wp_profile_root*`/ConfigEngine`
   - Windows: *wp_profile_root*`\ConfigEngine`
3. Run the following task to migrate the PAA content:

This task deletes the auto-generated code and then re-creates it in the `config/includes` directory for each component. The task then registers the assembly and components with the current version ConfigEngine.

- IBM i: `ConfigEngine.sh migrate-paa -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`*
- Windows: `ConfigEngine.bat migrate-paa -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`*

Managing your existing Portal Application Archive (PAA) file

When you update an existing Portal Application Archive (PAA) file, it is not as simple as overwriting the file with the new content. There might be large differences between the content in both files. Therefore, to manage your PAA file, you can upgrade to the latest version and rollback to the previous version.

About this task

The Solution Installer provides the following tasks to manage your existing PAA files:

- **`install-paa-update`**
- **`remove-paa-update`**

"Updating an existing Portal Application Archive (PAA) file"
You can update an existing Portal Application Archive (PAA) file when a new version is available. The process is not as simple as overwriting the file with the new content. There might be large differences between the content in both versions of the file.

"Uninstalling a Portal Application Archive (PAA) file update" on page 216
If there are issues with the Portal Application Archive (PAA) file update, you can uninstall the update and return to the previous version of the file.

Updating an existing Portal Application Archive (PAA) file

You can update an existing Portal Application Archive (PAA) file when a new version is available. The process is not as simple as overwriting the file with the new content. There might be large differences between the content in both versions of the file.

About this task

The **`install-paa-update`** command creates a backup of your current PAA file. It is stored in the *wp_profile_root*`/paa/backup` directory. The command then completes the following actions:

- Uninstalls the PAA file from IBM WebSphere Portal Express
- Deletes the content from the expanded PAA file
- Installs the new version of the PAA file

After you run the **`install-paa-update`** command, you must deploy the changes to your system.

PAA file developer note: Make all updates in the custom code. No code generation is available for an upgrade through the regular **`install-paa`** process. Read the guide about creating updated PAA files for information.

Cluster note: Complete these steps on the primary node and then on all additional nodes.

Procedure

1. Open a command line.
2. Change to the *wp_profile_root*/ConfigEngine directory.
3. Run the following task to update the PAA file:
   - Linux : `./ConfigEngine.sh install-paa-update -DPAALocation=paaLocation/yourPaa.paa -DappName=assemblyName -DWasPassword=password -DPortalAdminPwd=password`
   - IBM i: `ConfigEngine.sh install-paa-update -DPAALocation=paaLocation/yourPaa.paa -DappName=assemblyName -DWasPassword=password -DPortalAdminPwd=password`
   - Windows: `ConfigEngine.bat install-paa-update -DPAALocation=paaLocation/yourPaa.paa -DappName=assemblyName -DWasPassword=password -DPortalAdminPwd=password`

   Optional parameter: You can add the following parameter to your **install-paa-update** command: **-Dwcmdetect=true**. This parameter controls the behavior of the installed IBM Web Content Manager libraries. If you include this parameter, a properties file is created in the paa/*ComponentName* directory. The name of the properties file matches the PAA file name.
4. After the task successfully completes, verify that the following subdirectory exists:
   - IBM i: *wp_profile_root*/paa/*paa_filename*
   - Windows: *wp_profile_root*\paa*paa_filename*
5. From the PAA file subdirectory, open the components.properties file. Complete the following steps to check for and resolve conflicts with previously installed components:
   a. Search for the component parameters. Parameters that are set to true are not already installed. Parameters that are set to false are already installed.
   b. To update existing components, change their value in the components.properties file to true.
   c. Save your changes to the components.properties file.
6. Optional: Complete the following steps if you included the **-Dwcmdetect=true** parameter:
   a. Open the properties file in the paa/*ComponentName* directory.
   b. Set the value of all the Web Content Manager libraries that you want to update to true.
   c. Set the value of all the Web Content Manager libraries that you want to keep the old version to false.
   d. Save your changes.
7. Optional: Run the following task if you changed values in the components.properties file:

   This task updates the registration of the existing components with the new assembly.
   - IBM i: `ConfigEngine.sh update-paa-components -DappName=assemblyName -DWasPassword=password -DPortalAdminPwd=password`
   - Windows: `ConfigEngine.bat update-paa-components -DappName=assemblyName -DWasPassword=password -DPortalAdminPwd=password`

8. Optional: If you use the Simple and Protected GSS-API Negotiation Mechanism (SPNEGO) for single sign-on, complete the following steps to disable SPNEGO:
   a. Log on to WebSphere Integrated Solutions Console.
   b. Go to **Security** > **Global security** > **Web and SIP security** > **SPNEGO Web authentication**.
   c. Clear the **Enable SPNEGO** check box.
   d. Save your changes.
9. Run the following task to deploy the updated PAA content to WebSphere Portal Express:
   - IBM i: `ConfigEngine.sh deploy-paa -DappName=`*`assemblyName`* `-DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`*
   - Windows: `ConfigEngine.bat deploy-paa -DappName=`*`assemblyName`* `-DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`*

   Clustered environment parameters: If you are deploying to a clustered environment and your PAA file contains XMLAccess script files, add the following two parameters to the **deploy-paa** task:
   - **-DmaxTimeToWait**
   - **-DmaxAppTimeToWait**

   These values define the time that the **wplc-wait-for-sync-to-complete** task waits to synchronize your nodes. The default values are `-DmaxTimeToWait=30` and `-DmaxAppTimeToWait=5`. The values are in minutes. Add these parameters to your **deploy-paa** task with values that meet your requirements.

   Virtual portal parameters: If you are deploying to a virtual portal, you must include the context root and host name parameters for the virtual portal. Add the **-DVirtualPortalHostName** and **-DVirtualPortalContext** parameters to the **deploy-paa** task. Read "Virtual portals" on page 1361 for information.

   Note: By default, only components that are set to `true` in the `components.properties` file are deployed. To deploy all components, add the **-DforceDeploy=true** parameter to the **deploy-paa** task.
10. Optional: Complete the following steps to enable SPNEGO:
    a. Log on to WebSphere Integrated Solutions Console.
    b. Go to **Security** > **Global security** > **Web and SIP security** > **SPNEGO Web authentication**.
    c. Check the **Enable SPNEGO** check box.
    d. Save your changes.

Uninstalling a Portal Application Archive (PAA) file update

If there are issues with the Portal Application Archive (PAA) file update, you can uninstall the update and return to the previous version of the file.

About this task

The **remove-paa-update** command uninstalls the updates to your IBM WebSphere Portal Express server. It then installs and deploys the original PAA file in the *wp_profile_root*`/paa/backup` directory.

PAA file developer note: Custom code is used to remove updates and restore the previous version of the PAA file. The Solution Installer does not generate code for

this task. However, the previously generated code is called from the custom tasks. Previously generated tasks are not run automatically.

Cluster note: Complete these steps on the primary node and then on all additional nodes.

Procedure

1. Optional: If you use the Simple and Protected GSS-API Negotiation Mechanism (SPNEGO) for single sign-on, complete the following steps to disable SPNEGO:
   a. Log on to WebSphere Integrated Solutions Console.
   b. Go to **Security** > **Global security** > **Web and SIP security** > **SPNEGO Web authentication**.
   c. Clear the **Enable SPNEGO** check box.
   d. Save your changes.
2. Open a command prompt.
3. Change to the *wp_profile_root*/ConfigEngine directory.
4. Run the following task to update the PAA file:
   - Linux : `./ConfigEngine.sh remove-paa-update -DappName=`*`assemblyName`* `-DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`*
   - IBM i: `ConfigEngine.sh remove-paa-update -DappName=`*`assemblyName`* `-DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`*
   - Windows: `ConfigEngine.bat remove-paa-update -DappName=`*`assemblyName`* `-DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`*
5. Optional: Complete the following steps to enable SPNEGO:
   a. Log on to WebSphere Integrated Solutions Console.
   b. Go to **Security** > **Global security** > **Web and SIP security** > **SPNEGO Web authentication**.
   c. Check the **Enable SPNEGO** check box.
   d. Save your changes.

Solution Installer run time configuration

The Solution Installer run time configuration properties allows you to control how the Solution Installer installs a Portal Application Archive (PAA) formatting application to IBM WebSphere Portal Express.

The Solution Installer run time configuration properties are located in the `wkplc_comp.properties` file. This file is located in the *wp_profile_root*`\ConfigEngine\properties` directory. Generally the Solution Installer run time configuration properties require no edits; the following exception applies:

To include custom code extension points with suffixes other than **`-applySIFeaturePack`** and **`-removeSIFeaturePack`**. If you want to include additional extension points to **`-applySIFeaturePack`** and **`-removeSIFeaturePack`**, add them to the existing items using a comma separated list.

The properties for adding different extension suffixes are the **`wp.si.configInstallExtensionList`** and **`wp.si.configRemoveExtensionList`**. For example:

- **`wp.si.configInstallExtensionList=`**`-applySIFeaturePack`
- **`wp.si.configRemoveExtensionList=`**`-removeSIFeaturePack`

"Running the Solution Installer without an internet connection"
You can run the Solution Installer from a server that does not have an internet connection.

Running the Solution Installer without an internet connection

You can run the Solution Installer from a server that does not have an internet connection.

Procedure

1. Open the `wkplc_comp.properties` file, which is in the *wp_profile_root*\`ConfigEngine\properties` directory.
2. Locate the **`wp.si.offlineMode`** parameter.
3. Change the value of the **`wp.si.offlineMode`** parameter to `true`.

   Setting this value to `true` means that no attempts are made to retrieve external DTD by the XML parser when reading XML content.
4. Save your changes.

Uninstalling the digital experience software

Uninstalling the digital experience software is a multiple step process. Manual uninstallation instructions are provided for a single-server configuration in case of an error situation.

1. "IBM i: Uninstalling WebSphere Portal Express"
   Use the IBM Installation Manager to uninstall IBM WebSphere Portal Express binary files.
2. "Linux: Uninstalling WebSphere Portal Express" on page 222
   Use the IBM Installation Manager to uninstall IBM WebSphere Portal Express binary files.
3. "Windows: Uninstalling WebSphere Portal Express" on page 226
   Use the IBM Installation Manager to uninstall IBM WebSphere Portal Express binary files.

Related tasks:

"Planning to install WebSphere Portal Express" on page 101
Before you install IBM WebSphere Portal Express in a production environment, you need to assess your hardware and software needs, possible database configurations, security options, and LDAP server options. Skipping this important step can lead to unexpected results and costly delays.

"Installing and preparing the prerequisite software" on page 148
Before you install the digital experience software, make sure that the prerequisite software is installed and configured. Depending on your environment, you might already have the prerequisites, such as a database server and user registry. Verify that the prerequisite software is the correct version, has the required fix packs applied, and is configured to work with the digital experience software.

Related reference:

"Getting the software" on page 135
There are different ways for you to get WebSphere Portal Express and Web Content Manager Version 8.5 software.

IBM i: Uninstalling WebSphere Portal Express

Use the IBM Installation Manager to uninstall IBM WebSphere Portal Express binary files.

"IBM i: Restrictions on moving a node to a stand-alone configuration"
In a working cluster, all nodes share a common database. If you want to remove a node from a cell to use the node in a stand-alone configuration, some restrictions apply.

"IBM i: Preparing to uninstall"
You must prepare your system before you uninstall your IBM WebSphere Portal Express environment. For example, add passwords to the properties files. You must also decide to keep or discard the database information.

"IBM i: Uninstalling WebSphere Portal Express" on page 221
If you have a complete and functional uninstallation program, you can uninstall IBM WebSphere Portal Express only or both WebSphere Portal Express and IBM WebSphere Application Server.

IBM i: Restrictions on moving a node to a stand-alone configuration

In a working cluster, all nodes share a common database. If you want to remove a node from a cell to use the node in a stand-alone configuration, some restrictions apply.

The configuration of all portlets deployed in a cell are stored in a common database. When you remove a node from a cell to use the node in a stand-alone configuration, the portlets that had been available to the node in the cell are no longer available to it as a stand-alone server. Other changes in configuration that were made after the node was federated to the cell, such as enabling LDAP security or applying fix pack maintenance, can prevent the node from operating normally after it is removed from the cell. Starting or modifying the configuration of the stand-alone node before taking steps to back up the database can introduce conflicts between the node and the remaining nodes in the cell.

Before you start or modify the configuration of the stand-alone node, restore the WebSphere Application Server file system and the WebSphere Portal Express databases using backups taken prior to federation. Reconnect to a database that represents the portlet and page configuration of the node before it was added to the cell. Do not reconnect to the default database.

IBM i: Preparing to uninstall

You must prepare your system before you uninstall your IBM WebSphere Portal Express environment. For example, add passwords to the properties files. You must also decide to keep or discard the database information.

About this task

Important cluster note: You must issue the **removeNode** command to unfederate a node before uninstalling because WebSphere Portal Express cannot uninstall a federated node.

Procedure

1. Optional: Make a backup of the WebSphere Portal Express configuration. Use the XML Configuration Interface.

   Important: If you delete the database, the following information is not backed up and is deleted:
   - User attributes that are stored in the database and not in the user registry
   - Credential data that is stored in the default vault implementation

2. Complete the following steps to remove a node from the cell in a clustered environment:

   Note: Removing a WebSphere Portal Express node from the cell does not affect the cluster definition that you originally created for your cluster. The cluster definition remains intact even after you remove all WebSphere Portal Express nodes from the cell. In addition, removing a WebSphere Portal Express node from the cell does not remove the product's enterprise applications from the deployment manager. The enterprise applications remain and continue to be associated with the cluster definition.

   a. Log on to the Deployment Manager WebSphere Integrated Solutions Console.
   b. Go to **Servers** > **WebSphere application server clusters** > ***cluster_name*** > **Cluster members**, where *cluster_name* is the name of your cluster, click the server that you want to stop, and then click **Stop**.
   c. Go to **System Administration** > **Nodes**. Select the node that contains the server that you want to remove from the cell, and then click **Remove Node** to remove the node from the cell.

      Important: Make sure that you choose the **Remove Node** option to remove the node from cell and not the **Delete** option on the **Cluster members** view. The **Delete** option completely deletes the node, which removes the existence of the server from the deployment manager. It does not leave a means for restoring the WebSphere Portal Express node to a stand-alone system. Using the **Delete** option can prevent the WebSphere Portal Express server from working after it is deleted from the cluster. If **Remove Node** does not successfully remove the node, click **Force Delete** to remove the node.
   d. Click **Save** to save the changes to the cell's configuration.
   e. Repeat the previous steps for each node in the cluster and cell that you want to uninstall.
   f. Optional: Complete the following steps if you plan to convert the stand-alone server to a working portal:
      1) Use a text editor to open the `wkplc.properties` file.
      2) Change the value of the **`CellName`** property so that it matches the cell name of the node itself.
         - The cell name for the node reverts to the cell name that was used before you federated the node.
         - The cell name can be identified by the *wp_profile_root*`/config/cells/`*`cell_name`* directory on the node, where *cell_name* indicates the cell to which the node belongs.
      3) Change the value of the **`ServerName`** property to the original WebSphere Portal Express server name.
      4) Ensure that the value of the **`PrimaryNode`** property is set to `true`.
      5) Save your changes.
3. Add passwords to the `wkplc.properties`, `wkplc_dbdomain.properties`, and `wkplc_dbtype.properties` files in the *wp_profile_root*`/ConfigEngine/properties` directory. You can also specify passwords on the command line.
4. Decide whether to keep your database to preserve WebSphere Portal Express information.
   - If you keep the database, no further steps are required.

Note: If you choose to keep the database information, you cannot use it with subsequent installations although you can still access the information through your database software. Also, if you keep the information, you can always delete the WebSphere Portal Express databases and database tables later with the database software.

- Complete the following steps to remove the information from the database:
  a. Stop all the servers. For specific instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.
  b. Open a command prompt and change to the *wp_profile_root*`/ConfigEngine` directory.
  c. Run the `ConfigEngine.sh remove-schema -DWasPassword=`*`password`* `-Drelease.DbPassword=`*`password`* `-Dcustomization.DbPassword=`*`password`* `-Dcommunity.DbPassword=`*`password`* `-Djcr.DbPassword=`*`password`* `-Dfeedback.DbPassword=`*`password`* `-Dlikeminds.DbPassword=`*`password`* task.

  Note: Some tables remain in the IBM Java Content Repository database. Removing the database removes these tables.

IBM i: Uninstalling WebSphere Portal Express

If you have a complete and functional uninstallation program, you can uninstall IBM WebSphere Portal Express only or both WebSphere Portal Express and IBM WebSphere Application Server.

Procedure

1. Stop all the servers. For specific instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.
2. Verify that you are not running any other installation or uninstallation programs.
3. Choose one of the following uninstallation commands to uninstall WebSphere Portal Express and WebSphere Application Server:
   - Graphical user interface: Start the Installation Manager and then select **Uninstall**.
   - Response file: From your installation source directory, run the `imcl input` *`pathtoresponse.xml`* `-log` *`pathtologfiles`* task.

     Important: Do not place the response file in a path that contains a space and do not put a space in the file name.
   - Console mode: Run the `imcl -c` task from the `install_root/Installation Manager/eclipse/tools` directory. Then, type `5`.

     Restriction: You can select only one package group to uninstall at a time. First, uninstall the WebSphere Portal Express package. Then, uninstall the WebSphere Application Server package.

   During the uninstallation, remove all profiles that you do not want to keep for future installations.
4. Remove any remaining WebSphere Portal Express directories from your directory structure.
5. If you uninstall WebSphere Portal Express only, go to the *AppServer_root* directory and remove the following files:
   - `lib/ext/commons-codec-1.3.jar`
   - `lib/ext/commons-httpclient-3.0.1.jar`
   - `lib/ext/openid4java-full-0.9.5.jar`

- lib/ext/wp.auth.base.sua_RedirectServletFilter.jar
- lib/ext/wp.auth.base.sua_loginmodule.jar
- lib/ext/wp.auth.tai.jar
- lib/wp.user.connections.jar
- lib/wp.wire.jar
- plugins/com.ibm.patch.was.plugin.jar
- plugins/com.ibm.wp.was.plugin.jar
- plugins/wp.ext.jar
- properties/jndi.properties

6. Examine all running processes and stop ones that contain the PortalServer_root directory. Restart the server, especially if you intend to reinstall WebSphere Portal Express on the same server.
7. Go to Deleting specific cluster members for information about how to delete a cluster member from the deployment manager.
8. Complete the following steps to delete the WebSphere Portal Express server from the deployment manager:
   a. Log on to the WebSphere Integrated Solutions Console.
   b. Click **Servers** > **Application Servers**.
   c. Select the check box for the WebSphere Portal Express server you want to delete.
   d. Click **Delete**.
9. After uninstalling WebSphere Portal Express and WebSphere Application Server, you can also delete the Installation Manager. Run the ./uninstallc command from the qibm/userdata/InstallationManager/uninstall directory.

Linux: Uninstalling WebSphere Portal Express

Use the IBM Installation Manager to uninstall IBM WebSphere Portal Express binary files.

"Linux: Restrictions on moving a node to a stand-alone configuration"
In a working cluster, all nodes share a common database. If you want to remove a node from a cell to use the node in a stand-alone configuration, some restrictions apply.

"Linux: Preparing to uninstall" on page 223
You must prepare your system before you uninstall your IBM WebSphere Portal Express environment. For example, add passwords to the properties files. You must also decide to keep or discard the database information.

"Linux: Uninstalling WebSphere Portal Express" on page 224
If you have a complete and functional uninstallation program, you can uninstall IBM WebSphere Portal Express only or both WebSphere Portal Express and IBM WebSphere Application Server.

Linux: Restrictions on moving a node to a stand-alone configuration

In a working cluster, all nodes share a common database. If you want to remove a node from a cell to use the node in a stand-alone configuration, some restrictions apply.

The configuration of all portlets deployed in a cell are stored in a common database. When you remove a node from a cell to use the node in a stand-alone configuration, the portlets that had been available to the node in the cell are no longer available to it as a stand-alone server. Other changes in configuration that

were made after the node was federated to the cell, such as enabling LDAP security or applying fix pack maintenance, can prevent the node from operating normally after it is removed from the cell. Starting or modifying the configuration of the stand-alone node before taking steps to back up the database can introduce conflicts between the node and the remaining nodes in the cell.

Before you start or modify the configuration of the stand-alone node, restore the WebSphere Application Server file system and the WebSphere Portal Express databases using backups taken prior to federation. Reconnect to a database that represents the portlet and page configuration of the node before it was added to the cell. Do not reconnect to the default database.

Linux: Preparing to uninstall

You must prepare your system before you uninstall your IBM WebSphere Portal Express environment. For example, add passwords to the properties files. You must also decide to keep or discard the database information.

About this task

Important cluster note: You must issue the **removeNode** command to unfederate a node before uninstalling because WebSphere Portal Express cannot uninstall a federated node.

Procedure

1. Optional: Make a backup of the WebSphere Portal Express configuration. Use the XML Configuration Interface.

   Important: If you delete the database, the following information is not backed up and is deleted:
   - User attributes that are stored in the database and not in the user registry
   - Credential data that is stored in the default vault implementation
2. Complete the following steps to remove a node from the cell in a clustered environment:

   Note: Removing a WebSphere Portal Express node from the cell does not affect the cluster definition that you originally created for your cluster. The cluster definition remains intact even after you remove all WebSphere Portal Express nodes from the cell. In addition, removing a WebSphere Portal Express node from the cell does not remove the product's enterprise applications from the deployment manager. The enterprise applications remain and continue to be associated with the cluster definition.
   a. Log on to the Deployment Manager WebSphere Integrated Solutions Console.
   b. Go to **Servers** > **WebSphere application server clusters** > ***cluster_name*** > **Cluster members**, where *cluster_name* is the name of your cluster, click the server that you want to stop, and then click **Stop**.
   c. Go to **System Administration** > **Nodes**. Select the node that contains the server that you want to remove from the cell, and then click **Remove Node** to remove the node from the cell.

      Important: Make sure that you choose the **Remove Node** option to remove the node from cell and not the **Delete** option on the **Cluster members** view. The **Delete** option completely deletes the node, which removes the existence of the server from the deployment manager. It does not leave a means for

restoring the WebSphere Portal Express node to a stand-alone system. Using the **Delete** option can prevent the WebSphere Portal Express server from working after it is deleted from the cluster. If **Remove Node** does not successfully remove the node, click **Force Delete** to remove the node.

d. Click **Save** to save the changes to the cell's configuration.

e. Repeat the previous steps for each node in the cluster and cell that you want to uninstall.

f. Optional: Complete the following steps if you plan to convert the stand-alone server to a working portal:

1) Use a text editor to open the `wkplc.properties` file.

2) Change the value of the **`CellName`** property so that it matches the cell name of the node itself.

- The cell name for the node reverts to the cell name that was used before you federated the node.
- The cell name can be identified by the *wp_profile_root*`/config/cells/`*`cell_name`* directory on the node, where *cell_name* indicates the cell to which the node belongs.

3) Change the value of the **`ServerName`** property to the original WebSphere Portal Express server name.

4) Ensure that the value of the **`PrimaryNode`** property is set to `true`.

5) Save your changes.

3. Add passwords to the `wkplc.properties`, `wkplc_dbdomain.properties`, and `wkplc_dbtype.properties` files in the *wp_profile_root*`/ConfigEngine/properties` directory. You can also specify passwords on the command line.

4. Decide whether to keep your database to preserve WebSphere Portal Express information.

- If you keep the database, no further steps are required.

  Note: If you choose to keep the database information, you cannot use it with subsequent installations although you can still access the information through your database software. Also, if you keep the information, you can always delete the WebSphere Portal Express databases and database tables later with the database software.

- Complete the following steps to remove the information from the database:

  a. Stop all the servers. For specific instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.

  b. Open a command prompt and change to the *wp_profile_root*`/ConfigEngine` directory.

  c. Run the `./ConfigEngine.sh remove-schema -DWasPassword=`*`password`* `-Drelease.DbPassword=`*`password`* `-Dcustomization.DbPassword=`*`password`* `-Dcommunity.DbPassword=`*`password`* `-Djcr.DbPassword=`*`password`* `-Dfeedback.DbPassword=`*`password`* `-Dlikeminds.DbPassword=`*`password`* task.

  Note: Some tables remain in the IBM Java Content Repository database. Removing the database removes these tables.

Linux: Uninstalling WebSphere Portal Express

If you have a complete and functional uninstallation program, you can uninstall IBM WebSphere Portal Express only or both WebSphere Portal Express and IBM WebSphere Application Server.

Procedure

1. If you uninstall as a non-root user, verify that all product directories and files have the correct permissions. If not, set their permissions to the non-root user.
2. Stop all the servers. For specific instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.
3. Verify that you are not running any other installation or uninstallation programs.
4. Choose one of the following uninstallation commands to uninstall WebSphere Portal Express and WebSphere Application Server:
   - Graphical user interface: Start the Installation Manager and then select **Uninstall**.
   - Response file: From your installation source directory, run the `./imcl input` *`pathtoresponse.xml`* `-log` *`pathtologfiles`* task.

     Important: Do not place the response file in a path that contains a space and do not put a space in the file name.
   - Console mode: Run the `./imcl -c` task from the `install_root/Installation Manager/eclipse/tools` directory. Then, type `5`.

     Restriction: You can select only one package group to uninstall at a time. First, uninstall the WebSphere Portal Express package. Then, uninstall the WebSphere Application Server package.

   During the uninstallation, remove all profiles that you do not want to keep for future installations.
5. Remove any remaining WebSphere Portal Express directories from your directory structure.
6. If you uninstall WebSphere Portal Express only, go to the *AppServer_root* directory and remove the following files:
   - `lib/ext/commons-codec-1.3.jar`
   - `lib/ext/commons-httpclient-3.0.1.jar`
   - `lib/ext/openid4java-full-0.9.5.jar`
   - `lib/ext/wp.auth.base.sua_RedirectServletFilter.jar`
   - `lib/ext/wp.auth.base.sua_loginmodule.jar`
   - `lib/ext/wp.auth.tai.jar`
   - `lib/wp.user.connections.jar`
   - `lib/wp.wire.jar`
   - `plugins/com.ibm.patch.was.plugin.jar`
   - `plugins/com.ibm.wp.was.plugin.jar`
   - `plugins/wp.ext.jar`
   - `properties/jndi.properties`
7. Examine all running processes and stop ones that contain the `PortalServer_root` directory. Restart the server, especially if you intend to reinstall WebSphere Portal Express on the same server.
8. Go to Deleting specific cluster members for information about how to delete a cluster member from the deployment manager.
9. Complete the following steps to delete the WebSphere Portal Express server from the deployment manager:
   a. Log on to the WebSphere Integrated Solutions Console.
   b. Click **Servers** > **Application Servers**.

c. Select the check box for the WebSphere Portal Express server you want to delete.
d. Click **Delete**.

10. After uninstalling WebSphere Portal Express and WebSphere Application Server, you can also delete the Installation Manager. Run the `./uninstall` command from the `ibm/InstallationManager/uninstall` directory.

Windows: Uninstalling WebSphere Portal Express

Use the IBM Installation Manager to uninstall IBM WebSphere Portal Express binary files.

"Windows: Restrictions on moving a node to a stand-alone configuration"
In a working cluster, all nodes share a common database. If you want to remove a node from a cell to use the node in a stand-alone configuration, some restrictions apply.

"Windows: Preparing to uninstall"
You must prepare your system before you uninstall your IBM WebSphere Portal Express environment. For example, add passwords to the properties files. You must also decide to keep or discard the database information.

"Windows: Uninstalling WebSphere Portal Express" on page 228
If you have a complete and functional uninstallation program, you can uninstall IBM WebSphere Portal Express only or both WebSphere Portal Express and IBM WebSphere Application Server.

Windows: Restrictions on moving a node to a stand-alone configuration

In a working cluster, all nodes share a common database. If you want to remove a node from a cell to use the node in a stand-alone configuration, some restrictions apply.

The configuration of all portlets deployed in a cell are stored in a common database. When you remove a node from a cell to use the node in a stand-alone configuration, the portlets that had been available to the node in the cell are no longer available to it as a stand-alone server. Other changes in configuration that were made after the node was federated to the cell, such as enabling LDAP security or applying fix pack maintenance, can prevent the node from operating normally after it is removed from the cell. Starting or modifying the configuration of the stand-alone node before taking steps to back up the database can introduce conflicts between the node and the remaining nodes in the cell.

Before you start or modify the configuration of the stand-alone node, restore the WebSphere Application Server file system and the WebSphere Portal Express databases using backups taken prior to federation. Reconnect to a database that represents the portlet and page configuration of the node before it was added to the cell. Do not reconnect to the default database.

Windows: Preparing to uninstall

You must prepare your system before you uninstall your IBM WebSphere Portal Express environment. For example, add passwords to the properties files. You must also decide to keep or discard the database information.

About this task

Important cluster note: You must issue the **removeNode** command to unfederate a node before uninstalling because WebSphere Portal Express cannot uninstall a federated node.

Procedure

1. Optional: Make a backup of the WebSphere Portal Express configuration. Use the XML Configuration Interface.

   Important: If you delete the database, the following information is not backed up and is deleted:
   - User attributes that are stored in the database and not in the user registry
   - Credential data that is stored in the default vault implementation
2. Complete the following steps to remove a node from the cell in a clustered environment:

   Note: Removing a WebSphere Portal Express node from the cell does not affect the cluster definition that you originally created for your cluster. The cluster definition remains intact even after you remove all WebSphere Portal Express nodes from the cell. In addition, removing a WebSphere Portal Express node from the cell does not remove the product's enterprise applications from the deployment manager. The enterprise applications remain and continue to be associated with the cluster definition.
   a. Log on to the Deployment Manager WebSphere Integrated Solutions Console.
   b. Go to **Servers** > **WebSphere application server clusters** > ***cluster_name*** > **Cluster members**, where *cluster_name* is the name of your cluster, click the server that you want to stop, and then click **Stop**.
   c. Go to **System Administration** > **Nodes**. Select the node that contains the server that you want to remove from the cell, and then click **Remove Node** to remove the node from the cell.

      Important: Make sure that you choose the **Remove Node** option to remove the node from cell and not the **Delete** option on the **Cluster members** view. The **Delete** option completely deletes the node, which removes the existence of the server from the deployment manager. It does not leave a means for restoring the WebSphere Portal Express node to a stand-alone system. Using the **Delete** option can prevent the WebSphere Portal Express server from working after it is deleted from the cluster. If **Remove Node** does not successfully remove the node, click **Force Delete** to remove the node.
   d. Click **Save** to save the changes to the cell's configuration.
   e. Repeat the previous steps for each node in the cluster and cell that you want to uninstall.
   f. Optional: Complete the following steps if you plan to convert the stand-alone server to a working portal:
      1) Use a text editor to open the `wkplc.properties` file.
      2) Change the value of the **CellName** property so that it matches the cell name of the node itself.
         - The cell name for the node reverts to the cell name that was used before you federated the node.

- The cell name can be identified by the *wp_profile_root*/config/cells/*cell_name* directory on the node, where *cell_name* indicates the cell to which the node belongs.

3) Change the value of the **ServerName** property to the original WebSphere Portal Express server name.
4) Ensure that the value of the **PrimaryNode** property is set to true.
5) Save your changes.

3. Add passwords to the wkplc.properties, wkplc_dbdomain.properties, and wkplc_dbtype.properties files in the *wp_profile_root*/ConfigEngine/properties directory. You can also specify passwords on the command line.
4. Decide whether to keep your database to preserve WebSphere Portal Express information.
   - If you keep the database, no further steps are required.

     Note: If you choose to keep the database information, you cannot use it with subsequent installations although you can still access the information through your database software. Also, if you keep the information, you can always delete the WebSphere Portal Express databases and database tables later with the database software.
   - Complete the following steps to remove the information from the database:
     a. Stop all the servers. For specific instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.
     b. Open a command prompt and change to the *wp_profile_root*/ConfigEngine directory.
     c. Run the ConfigEngine.bat remove-schema -DWasPassword=*password* -Drelease.DbPassword=*password* -Dcustomization.DbPassword=*password* -Dcommunity.DbPassword=*password* -Djcr.DbPassword=*password* -Dfeedback.DbPassword=*password* -Dlikeminds.DbPassword=*password* task.

     Note: Some tables remain in the IBM Java Content Repository database. Removing the database removes these tables.

Windows: Uninstalling WebSphere Portal Express

If you have a complete and functional uninstallation program, you can uninstall IBM WebSphere Portal Express only or both WebSphere Portal Express and IBM WebSphere Application Server.

Procedure

1. Stop all the servers. For specific instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.
2. Verify that you are not running any other installation or uninstallation programs.
3. Choose one of the following uninstallation commands to uninstall WebSphere Portal Express and WebSphere Application Server:
   - Graphical user interface: Start the Installation Manager and then select **Uninstall**.
   - Response file: From your installation source directory, run the imcl input *pathtoresponse.xml* -log *pathtologfiles* task.

     Important: Do not place the response file in a path that contains a space and do not put a space in the file name.

- Console mode: Run the `imcl -c` task from the `install_root/Installation Manager/eclipse/tools` directory. Then, type `5`.

  Restriction: You can select only one package group to uninstall at a time. First, uninstall the WebSphere Portal Express package. Then, uninstall the WebSphere Application Server package.

During the uninstallation, remove all profiles that you do not want to keep for future installations.

4. Remove any remaining WebSphere Portal Express directories from your directory structure.
5. If you uninstall WebSphere Portal Express only, go to the *AppServer_root* directory and remove the following files:
   - `lib/ext/commons-codec-1.3.jar`
   - `lib/ext/commons-httpclient-3.0.1.jar`
   - `lib/ext/openid4java-full-0.9.5.jar`
   - `lib/ext/wp.auth.base.sua_RedirectServletFilter.jar`
   - `lib/ext/wp.auth.base.sua_loginmodule.jar`
   - `lib/ext/wp.auth.tai.jar`
   - `lib/wp.user.connections.jar`
   - `lib/wp.wire.jar`
   - `plugins/com.ibm.patch.was.plugin.jar`
   - `plugins/com.ibm.wp.was.plugin.jar`
   - `plugins/wp.ext.jar`
   - `properties/jndi.properties`
6. Examine all running processes and stop ones that contain the `PortalServer_root` directory. Restart the server, especially if you intend to reinstall WebSphere Portal Express on the same server.
7. Go to Deleting specific cluster members for information about how to delete a cluster member from the deployment manager.
8. Complete the following steps to delete the WebSphere Portal Express server from the deployment manager:
   a. Log on to the WebSphere Integrated Solutions Console.
   b. Click **Servers** > **Application Servers**.
   c. Select the check box for the WebSphere Portal Express server you want to delete.
   d. Click **Delete**.
9. After uninstalling WebSphere Portal Express and WebSphere Application Server, you can also delete the Installation Manager. Run the `uninstall.bat` command from the `ibm\InstallationManager\uninstall` directory.

Chapter 5. Configuring

Run the following tasks after you install and deploy IBM WebSphere Portal Express. They address tasks that are typically run one time and have a global effect. Some configuration changes are made more frequently or do not have a global effect. These tasks are addressed in the Administering section.

About this task

Review the Performance Tuning Guide to tune your stand-alone or clustered environment. Even if you have a clustered environment, review the Base Portal Tuning scenarios and the Tuning a cluster environment chapter. In addition, the tuning guide provides information about caches for WebSphere Portal Express. Until the latest tuning guide is available, refer to the previous tuning guide.

"Configuration Wizard" on page 232
Use the Configuration Wizard to set up stand-alone servers and new deployments, create clusters, migrate and update to new versions, and add new capabilities to existing deployments.

"Setting up a stand-alone server" on page 239
Use the Configuration Wizard to set up a stand-alone server. A stand-alone server is useful for different environments, such as a content authoring environment, test environment, and more.

"Tune your environment" on page 240
Tuning the servers is important to the performance of your environment. By default, IBM WebSphere Portal Express is not tuned for a production environment. Therefore, to ensure optimal performance review and complete the steps in the IBM WebSphere Portal Tuning Guide. If a tuning guide is not available for the current release, use the tuning guide from the previous product version.

"Web servers" on page 242
IBM WebSphere Portal Express uses the internal HTTP transport within IBM WebSphere Application Server to handle requests. However, because WebSphere Application Server also supports the use of an external web server, you can access WebSphere Portal Express from your web server. You can use a local web server on the same server as WebSphere Portal Express or you can use a remote web server on a different server. A remote web server is typical for a production environment or other high-traffic configuration and is also typically placed in DMZ outside a firewall to protect portal ports.

"WebSphere Portal" on page 247
First, finish your IBM WebSphere Portal Express deployment with the Configuration Wizard. Then, you can configure your environment further. For example, you can tune your servers to improve performance. You can change the default context root.

"Web Content Manager" on page 392
Set up a content server by installing IBM Web Content Manager in various deployments to provide robust and flexible environments for web content development and delivery. After you install the content server, more configuration steps must be completed according to the role that the server plays in your web content environment.

"Syndication" on page 448
Use syndication to replicate web content library data from one server to

another server. Syndication is based on a syndicator and subscriber relationship. The syndicator has the current data. The subscriber received the current data from the syndicator.

"Database Management Systems" on page 466
Configure the connection between IBM WebSphere Portal Express and your database management system. The **Database Transfer** configuration option in the Configuration Wizard assigns users and permissions, creates databases, obtains support for database collation, and transfers your database.

"User registry" on page 562
User information is stored in your user registry. You can enable LDAP referrals, configure IBM WebSphere Portal Express to use dynamic groups, update your user registry, or delete your user registry configurations.

"Search" on page 608
You use Portal Search to search for text that is displayed in websites that are created by IBM Web Content Manager.

Chapter 6, "Document Conversion Services," on page 727
Document Conversion Services are used when you work with the Common Mail Portlet, IBM Web Content Manager authoring and previewing, and search.

Chapter 7, "IBM Connections," on page 747
IBM Connections portlets give the IBM WebSphere Portal Express users access to more collaboration and social networking features such as activities, blogs, and bookmarks.

Configuration Wizard

Use the Configuration Wizard to set up stand-alone servers and new deployments, create clusters, migrate and update to new versions, and add new capabilities to existing deployments.

In the Configuration Wizard, you answer questions about the environment that you are configuring. Based on your answers, the wizard prompts you for custom values that are needed to configure your environment. Finally, the wizard generates custom steps and scripts to set up your environment.

Configuration options in the Configuration Wizard

Using the following configuration options, you can configure the portal server that you are connected to in real time. You can also download scripts and instructions to run on other servers, create reusable scripts for common configuration patterns, and create scripts to compare to scripts created in previous releases.

Database Transfer
Select this option to transfer data from Apache Derby to any of the database types that are supported by WebSphere Portal Express.

This option is found in **Set Up a Stand-alone Server** and in **Set Up a Cluster**.

Documentation resource: Database transfer

Documentation resource: "Troubleshooting: Database Transfer" on page 3561

Enable Federated Security
Add an LDAP user registry to the default federated repository to store user account information for authorization.

This option is found in **Set Up a Stand-alone Server** and in **Set Up a Cluster**.

Documentation resource: “Enable federated security” on page 563

Documentation resource: “Troubleshooting: Enable federated security option” on page 3582

Create a Deployment Manager

Create a deployment manager profile that is augmented with WebSphere Portal Express resources.

This option is found in **Set Up a Cluster**.

Documentation resource: Create a deployment manager

Documentation resource: “Troubleshooting: Create a deployment manager” on page 3587

Create a Cluster

Use the Configuration Wizard to create the primary node in your cluster.

This option is found in **Set Up a Cluster**.

Documentation resource: Create a cluster

Documentation resource: “Troubleshooting: Create a cluster option” on page 3589

Create an Additional Cluster Node

Use the Configuration Wizard to add nodes to a cluster.

This option is found in **Set Up a Cluster**.

Documentation resource: Create an additional cluster node

Documentation resource: “Troubleshooting: Create an additional cluster node” on page 3591

Install and Uninstall Add-ons

You can install add-on functionality to your WebSphere Portal Express with the solution installer through the Configuration Wizard.

This option is found in **Add On New Capability**.

Documentation resource: “Install and uninstall add-ons using the Configuration Wizard” on page 212

Migrate a Stand-alone Server

Use the Configuration Wizard to migrate a stand-alone server environment.

This option is found in **Migrate to a New Version**.

Documentation resource: “Migrate a stand-alone server” on page 842

Documentation resource: “Troubleshooting: Migrate a stand-alone server” on page 3599

Migrate a Cluster Step 1: Migrate the Deployment Manager Profile

Use the Configuration Wizard to migrate the deployment manager profile for a cluster environment. These steps must be completed before you start the migration of any nodes.

This option is found in **Migrate to a New Version**.

Documentation resource: “Cluster: Migrate the deployment manager profile” on page 847

Documentation resource: “Troubleshooting: Migrate the deployment manager profile for a cluster environment” on page 3605

Migrate a Cluster Step 2: Migrate Node Profiles

Use the Configuration Wizard to upgrade the node profiles for a cluster environment. These steps must be completed on all portal nodes in the cell before you begin the next cluster migration step.

This option is found in **Migrate to a New Version**.

Documentation resource: "Cluster: Migrate node profiles" on page 850

Documentation resource: "Troubleshooting: Migrate node profiles for a cluster environment" on page 3608

Migrate a Cluster Step 3: Upgrade Node Profiles
Use the Configuration Wizard to upgrade the nodes profiles for a cluster environment. Start these steps only after you migrate node profiles on all portal nodes in the cell.

This option is found in **Migrate to a New Version**.

Documentation resource: "Cluster: Upgrade node profiles" on page 853

Documentation resource: "Troubleshooting: Upgrade node profiles for a cluster environment" on page 3612

Recycle a Managed IBM WebSphere Portal Express Cell
Select this option to recycle the deployment manager and node agents. This configuration option runs the **action-cluster-recycle-dmgr** task.

This option is found in **More Options**.

Remove the IBM WebSphere Portal ExpressProfile
Use the Configuration Wizard to remove a portal profile.

This option is found in **More Options**.

Documentation resource: "Remove a WebSphere Portal profile" on page 384

Documentation resource: "Troubleshooting: Remove a WebSphere Portal profile" on page 3598

Configuration Wizard and clusters

In clustered environments, you can use the configuration wizard from the primary node. You can use the scripts from the primary node on the other nodes. You do not need to deploy the configuration wizard to the deployment manager.

Configuration Wizard profile

The wizard has a unique profile, `cw_profile`, and administrator credentials.

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Creating scripts and instructions" on page 235
The Configuration Wizard creates multiple files to help you complete your configuration objectives in a three-step process (**Answer Questions**, **Customize Values**, and **Configure**). When you select an option to customize for a specific configuration scenario, you provide information about your environment. Instructions, scripts, and helper files with updated property values are created to guide you in completing your configuration goal. You can run your configuration in real time or you can save your settings to use on another server.

"Configuration wizard runtime properties" on page 237
The Configuration wizard reads a properties file at startup. You can change these values and restart the wizard to control runtime behaviors.

"Running the configuration wizard silently" on page 237
You can run the configuration wizard silently if you have multiple deployments or if you want to develop a common deployment template.

"Installing or uninstalling the configuration wizard" on page 238
Use these instructions for the configuration wizard when installing or uninstalling the configuration wizard to another server with WebSphere Application Server installed.

Related concepts:

Chapter 3, "Roadmaps," on page 69
Review the roadmaps to understand the common deployment patterns that are supported by the configuration wizard.

Accessing the Configuration Wizard

The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

About this task

Click < and > to move back or forward in the Configuration Wizard.

Procedure

1. To get the latest updates for the wizard, apply the most recent combined cumulative fix. For more information about applying the latest fix pack, see WebSphere Portal and Web Content Manager V8.5.0.0 combined cumulative fix overview.

   Note: Skip this step, if you have the most recent fix pack applied.
2. Optional: Restart server1 to ensure that the Configuration Wizard uses the updated environment variables. Go to *AppServer_home*/profiles/cw_profile/bin and stop the server:
   - Linux : ./stopServer.sh server1 -username *username* -password *password*
   - IBM i: stopServer cw_profile -username *username* -password *password*
   - Windows: stopServer.bat server1 -username *username* -password *password*

   Then, start the server:
   - Linux : ./startServer.sh server1
   - IBM i: startServer cw_profile
   - Windows: startServer.bat server1
3. Access the Configuration Wizard. Go to http://*your_server*:*10200*/ibm/wizard.
4. Log in to the Configuration Wizard with the administrative ID for the configuration wizard profile, cw_profile.

   Note: The wizard user interface might not be available in all languages. If the language is not currently supported, you might see the English version. For details on the supported languages for all of the WebSphere Portal Express user interfaces, see "Supported languages" on page 3622.

Creating scripts and instructions

The Configuration Wizard creates multiple files to help you complete your configuration objectives in a three-step process (**Answer Questions**, **Customize**

Values, and **Configure**). When you select an option to customize for a specific configuration scenario, you provide information about your environment. Instructions, scripts, and helper files with updated property values are created to guide you in completing your configuration goal. You can run your configuration in real time or you can save your settings to use on another server.

Procedure

1. From the Configuration Wizard, click an option to customize your configuration goal.

Answer Questions

2. If you saved values from a previous session, click **Upload Saved Selections** to locate the `.xml` file with your values. Then, click **Load**.
3. Answer questions about your environment, configuration goals, and preferences. The Configuration Wizard uses these values to filter steps when it creates your scripts. For example, you can select the targeted operating system where portal is installed to customize the instructions and scripts that are created for your environment.
4. Click the forward arrow to proceed to **Customize Values**.

Customize Values

5. Review parameter and property values. Use the default values or enter new values. Default values are provided for some parameters. Sample values are also provided to guide you through the configuration process. To see more parameters that are not part of the basic configuration path, click **Advanced**.
6. Click the forward arrow to proceed to **Configure**.

Configure

You can start your configuration on the same system that you are using to access the configuration wizard or save your settings to use on another server.

7. Click **Download Wizard Selections** to save your configuration settings to configure similar environments. By saving your settings, you can reuse parameter values in the future. When you use this option, an XML file is created for you to download with the information that you provided.
8. Review parameter and property values. Some of the properties and parameters that display might not be relevant to your task.
9. Click **Download Configuration Scripts** to create a compressed file that contains scripts, instructions, and other files. This compressed file is downloaded to the location specified in your browser download settings. The configuration wizard creates the following files:

   Instruction file (html) named after your task
   The instruction file provides you with tailored steps for your configuration and your target operating system. Based on the information that you provided when customizing your task, an instruction file is created. This file guides you when you run scripts in combination with other manual configuration steps for your specific configuration goal.

   Scripts
   Depending on the conditions of your environment, the script files use a `.sh` or `.bat` file extension or are simple text files. Use the scripts, along with other manual configuration steps described by your instruction file, to run your configuration. For example, your script file might provide you with a script to run a ConfigEngine task rather than you running the task.

Updated properties
Configuration helper files with updated property values entered for your configuration are created when customizing your task. You can use these helper files to save time.

`.wfi` file
File used for troubleshooting your task.

10. Click **Start Configuration** to run your configuration.

Related concepts:

Chapter 3, "Roadmaps," on page 69
Review the roadmaps to understand the common deployment patterns that are supported by the configuration wizard.

Configuration wizard runtime properties

The Configuration wizard reads a properties file at startup. You can change these values and restart the wizard to control runtime behaviors.

The properties file is in the *wp_profile_root*`/WebSphere/AppServer/systemApps/isclite.ear/wizard.war/display.properties` directory. Any property change requires a restart of the wizard before they take effect.

The following actions are supported:

Run the wizard in non-execution mode
It is possible to run the wizard in a non-execution mode. The instance is created and the wizard proceeds through the steps. However, no commands are run. This method is useful with the **`printDebug`** parameter to debug issues. It is also useful to create the custom data that is used to run the wizard silently. Set the value to `false` to enable this mode.

Enable tracing
To enable tracing, add the **`printDebug`** parameter to the `display.properties` file. Set the value to `true`.

Running the configuration wizard silently

You can run the configuration wizard silently if you have multiple deployments or if you want to develop a common deployment template.

Procedure

1. Choose one of the following options to prepare to run the configuration wizard silently:
   - Generate the customization data file in the configuration wizard without running the tasks.
   - Modify the sample customization data file that matches your deployment. This file is in the *PortalServer_root*`/installer/samples/configwizard` directory.

     Important: The sample scenarios do not cover all deployment options. Therefore, if the samples do not match your deployment scenario, generate the customization data file.
2. Open a command prompt.
3. Change to the *wp_profile_root*`/ConfigEngine` directory.
4. Create a silent installation from a set of customization data:

Note: All parameters are necessary. Run this task once per deployment cycle.

- Linux : `./ConfigEngine.sh customize-workflow -DwfId=`*`workflow-identifier`* `-DwfData=`*`customization-data-file`* `-DwfOutput=`*`output-directory`* `-DWasPassword=`*`password`*
- IBM i: `ConfigEngine.sh customize-workflow -DwfId=`*`workflow-identifier`* `-DwfData=`*`customization-data-file`* `-DwfOutput=`*`output-directory`* `-DWasPassword=`*`password`*
- Windows: `ConfigEngine.bat customize-workflow -DwfId=`*`workflow-identifier`* `-DwfData=`*`customization-data-file`* `-DwfOutput=`*`output-directory`* `-DWasPassword=`*`password`*

Where *`workflow-identifier`* is the name in the repository.

Where *`customization-data-file`* is the name of the customization data file.

Where *`output-directory`* is the directory where resulting files are written. This directory includes the following content:

- The *`workflow-identifier`*`.wfi` file that represents the combination of silent installation tasks and the custom data. Subsequent tasks use this file.
- The *`workflow-identifier`* directory where scripts and related files are created.

5. Start the silent installation:
   - Linux : `./ConfigEngine.sh execute-workflow -DwfInstance=`*`workflow-instance`* `-DWasPassword=`*`password`*
   - IBM i: `ConfigEngine.sh execute-workflow -DwfInstance=`*`workflow-instance`* `-DWasPassword=`*`password`*
   - Windows: `ConfigEngine.bat execute-workflow -DwfInstance=`*`workflow-instance`* `-DWasPassword=`*`password`*

   This task runs until:
   - A step in the silent installation fails. You must correct the error and then run the **`resume-workflow`** task.
   - A step is reached that requires manual action. You must finish the requested action and then run the **`resume-workflow`** task.
6. To resume the silent installation, run the **`resume-workflow`** task:
   - Linux : `./ConfigEngine.sh resume-workflow -DWasPassword=`*`password`*
   - IBM i: `ConfigEngine.sh resume-workflow -DWasPassword=`*`password`*
   - Windows: `ConfigEngine.bat resume-workflow -DWasPassword=`*`password`*

   This task works in the following way:
   - If you resume the silent installation after an error, the task runs the failed step.
   - If you resume after you complete a manual step, the task runs the next step in the sequence.

Installing or uninstalling the configuration wizard

The configuration wizard is installed by default when you install IBM WebSphere Portal Express. As an alternative to using the default installation, you can install the configuration wizard on another server that already has IBM WebSphere Application Server installed.

Procedure

1. Change to the *AppServer_root*`/ConfigEngine` directory.
2. To install the configuration wizard, run the following command:

- Linux : **./ConfigEngine.sh deploy-wizard-war-standalone**
- IBM i: **ConfigEngine.sh deploy-wizard-war-standalone**
- Windows: **ConfigEngine.bat deploy-wizard-war-standalone**

3. To uninstall the configuration wizard, run the following command:
   - Linux : **./ConfigEngine.sh remove-wizard-war-standalone**
   - IBM i: **ConfigEngine.sh remove-wizard-war-standalone**
   - Windows: **ConfigEngine.bat remove-wizard-war-standalone**

Setting up a stand-alone server

Use the Configuration Wizard to set up a stand-alone server. A stand-alone server is useful for different environments, such as a content authoring environment, test environment, and more.

Before you begin

Roadmaps provide an overview of the steps that are required for common environment configurations. Select the roadmap that is most like the environment that you want to set up.

About this task

Use your selected roadmap and the Configuration Wizard to complete a new environment set up.

Procedure

1. Select the roadmap that is most like the configuration that you need to configure.
2. Access the Configuration Wizard. Go to `http://`*`your_server`*`:`*`10200`*`/ibm/wizard`.
3. Log in to the Configuration Wizard with the administrative ID for the configuration wizard profile, `cw_profile`.

   Note: The wizard user interface might not be available in all languages. If the language is not currently supported, you might see the English version. For details on the supported languages for all of the WebSphere Portal Express user interfaces, see “Supported languages” on page 3622.
4. Click **Set Up a Stand-alone Server**.
5. Complete each sub step in the order that is shown in the wizard. Use the Configuration Wizard in conjunction with your selected Roadmap.

Related concepts:

“Roadmaps for stand-alone servers” on page 69
A stand-alone server topology is useful for many environments. While the underlying topology is similar for many environments, the configuration steps to achieve a desired environment varies.

Database transfer
Your portal is installed with an Apache Derby database. The database that is immediately available for use is good for demonstration and portlet and theme development environments. Otherwise, you must configure portal to use a production-level database. Use the Configuration Wizard to transfer the data and configure a new database server.

"Enable federated security" on page 563
You can use the Configuration Wizard to configure WebSphere Portal to use a federated LDAP for security. Use the following information to get familiar with the information you must provide in the wizard and the configuration procedure that it generates.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Tune your environment

Tuning the servers is important to the performance of your environment. By default, IBM WebSphere Portal Express is not tuned for a production environment. Therefore, to ensure optimal performance review and complete the steps in the IBM WebSphere Portal Tuning Guide. If a tuning guide is not available for the current release, use the tuning guide from the previous product version.

About this task

Tuning your environment involves tuning and configuring the various systems and components. The tuning guide provides general concepts and detailed configuration instructions. Instructions are included for the following areas:

- Configuring the application server and the resources that are defined for that application server
- Determining the cloning strategy for expanding or extending the environment
- Tuning the database and database server
- Tuning the directory server and its database
- Tuning the web server
- Tuning the operating system and network
- Tuning the WebSphere Portal Express services

Procedure

1. Use the performance tuning tool to tune the portal server.
2. Read the tuning guide for assistance with tuning other servers in your environment.

   "Portal server performance tuning tool"
   Run the performance tuning tool on a new deployment to tune the servers that are based on performance recommendations.

Portal server performance tuning tool

Run the performance tuning tool on a new deployment to tune the servers that are based on performance recommendations.

Note: The performance tuning tool does not tune the database, the LDAP user registry, the web servers, or the operating system. In a clustered environment, it tunes only the cluster members. For advanced tuning, refer to the performance tuning guides.

Properties files

If necessary, modify the following properties files before you run the **tune-initial-portal-performance** task:

Remember: Review the performance tuning guide for information first. Before you modify the properties files, make a local copy of the *PortalServer_root*\installer\wp.config\config\TuningTask directory. Modify the files in the local copy. Then, add the -DTuningPropertiesDirectory=*local_dir_path* parameter to the **tune-initial-portal-performance** task.

- *PortalServer_root*\installer\wp.config\config\TuningTask\tuning.properties
- *PortalServer_root*\installer\wp.config\config\TuningTask\portal\CacheManagerService.properties
- *PortalServer_root*\installer\wp.config\config\TuningTask\portal\CommonComponentConfigService.properties
- *PortalServer_root*\installer\wp.config\config\TuningTask\portal\ConfigService.properties
- *PortalServer_root*\installer\wp.config\config\TuningTask\portal\CPConfigurationService.properties
- *PortalServer_root*\installer\wp.config\config\TuningTask\portal\NavigatorService.properties
- *PortalServer_root*\installer\wp.config\config\TuningTask\portal\RegistryService.properties
- *PortalServer_root*\installer\wp.config\config\TuningTask\portal.8\VirtualPortalConfigService.properties
- *PortalServer_root*\installer\wp.config\config\TuningTask\wcm\WCMConfigService.properties
- *PortalServer_root*\installer\wp.config\config\TuningTask\authoring\AccessControlDataManagementService.properties

Note: Starting with CF02, you can add the **-DAuthoringServer** parameter to the **tune-initial-portal-performance** task. Set this parameter to either true or false. Setting this parameter to true turns off the Web Content Manager advanced cache feature and any tuning features that are specific to a Subscriber server.

Stand-alone and clustered environments

Use the following syntax to run the **tune-initial-portal-performance** task on a stand-alone or clustered environment:

Note: In a clustered environment, run this task on each clustered node.

- Linux :./ConfigEngine.sh tune-initial-portal-performance -DWasPassword=*password* -DPortalAdminPwd=*password* -DTuningPropertiesDirectory=*local_dir_path*
- IBM i: ConfigEngine.sh tune-initial-portal-performance -DWasPassword=*password* -DPortalAdminPwd=*password* -DTuningPropertiesDirectory=*local_dir_path*

- Windows: `ConfigEngine.bat tune-initial-portal-performance -DWasPassword=password -DPortalAdminPwd=password -DTuningPropertiesDirectory=local_dir_path`

Vertical cluster member servers

Use the following syntax to run the **tune-initial-portal-performance** task on vertical cluster members:

Note: Run this task on each vertical cluster member on each node in the cluster.

- Linux :`./ConfigEngine.sh tune-initial-portal-performance -DServerName=vertical_cluster_servername -DWasPassword=password -DPortalAdminPwd=password -DTuningPropertiesDirectory=local_dir_path`
- IBM i: `ConfigEngine.sh tune-initial-portal-performance -DServerName=vertical_cluster_servername -DWasPassword=password -DPortalAdminPwd=password -DTuningPropertiesDirectory=local_dir_path`
- Windows: `ConfigEngine.bat tune-initial-portal-performance -DServerName=vertical_cluster_servername -DWasPassword=password -DPortalAdminPwd=password -DTuningPropertiesDirectory=local_dir_path`

Web servers

IBM WebSphere Portal Express uses the internal HTTP transport within IBM WebSphere Application Server to handle requests. However, because WebSphere Application Server also supports the use of an external web server, you can access WebSphere Portal Express from your web server. You can use a local web server on the same server as WebSphere Portal Express or you can use a remote web server on a different server. A remote web server is typical for a production environment or other high-traffic configuration and is also typically placed in DMZ outside a firewall to protect portal ports.

Configure your remote web server. Then, configure your web server to work with features such as IBM Web Content Manager.

"Configuring a remote web server" on page 243
To enable communication between the web server and WebSphere Application Server, a web server plug-in is required. The web server plug-in determines whether a request is handled by the web server or by the application server. The plug-in can be installed into a web server that is located either on the same server as WebSphere Application Server or on a separate server. The web server plug-in uses an XML configuration file (`plugin-cfg.xml`) that contains settings that describe how to handle and pass on requests to the WebSphere Application Server made accessible through the plug-in.

"Accessing WebSphere Portal Express through another HTTP port " on page 246
By default WebSphere Portal Express is configured to be accessed through the internal HTTP port in WebSphere Application Server. For example, `http://hostname.example.com:10039/wps/portal`, where *hostname.example.com* is the fully qualified host name of the server where Portal is running and *10039* is the default transport port that is created by WebSphere Application Server. The port number might be different for your environment. The default host name and port that is used by WebSphere Portal Express are specified by the **WpsHostName** and **WpsHostPort** properties in the `wkplc.properties` file.

Configuring a remote web server

To enable communication between the web server and WebSphere Application Server, a web server plug-in is required. The web server plug-in determines whether a request is handled by the web server or by the application server. The plug-in can be installed into a web server that is located either on the same server as WebSphere Application Server or on a separate server. The web server plug-in uses an XML configuration file (`plugin-cfg.xml`) that contains settings that describe how to handle and pass on requests to the WebSphere Application Server made accessible through the plug-in.

About this task

In the WebSphere Integrated Solutions Console, the web server is represented as a specific server type, and you can view or modify all of the configuration properties that are used in the `plugin-cfg.xml` file for the web server plug-in from the WebSphere Integrated Solutions Console.

Note: For some portal functions to work, you must make sure that the web server write and delete operations are allowed. These operations enable the HTTP operations POST, PUT, and DELETE. For example, these operations are required for the toolbar.

Choose the type of web server to configure:

"Configuring your Apache web server"
Configure the communication between IBM WebSphere Portal Express and your Apache web server.

"Configuring your Domino web server" on page 244
Configure the communication between IBM WebSphere Portal Express and your IBM Domino web server.

"Configuring your Oracle iPlanet web server" on page 245
Configure the communication between IBM WebSphere Portal Express and your Oracle iPlanet web server.

"Configuring your OnDemand Router" on page 245
Configure the communication between IBM WebSphere Portal Express and your OnDemand Router (ODR).

Configuring your Apache web server

Configure the communication between IBM WebSphere Portal Express and your Apache web server.

Procedure

1. If you are using IBM HTTP Server or Apache Server, edit the `httpd.conf` file on the web server. Set the **`AllowEncodedSlashes`** directive to `On`. Add the directive to the root level as a global directive.

HTTP server type	Documentation link
Read the appropriate HTTP Server documentation	IBM HTTP Server
Read the appropriate Apache Server documentation	**`AllowEncodedSlashes`** directives

2. Stop the web server.

3. Install and configure the web server plug-in on the system where the web server is located. Use the plug-ins installation wizard that is provided with WebSphere Application Server. Refer to the following topic for information:
   - Linux Windows: Selecting a Web server topology diagram and road map
   - IBM i: Selecting a Web server topology diagram and road map

   Important: Depending on how you use the web server, you must adjust the **ServerIOTimeout** parameter. It defines how long the plug-in must wait for a response from the application. The minimum value is `60` but you must increase this value if you are retrieving data from a database. To update this value, locate and open your `plugin-cfg.xml` file and set **ServerIOTimeout** to an appropriate value. For information, read Common questions about the Web server plug-in.
4. Web 2.0 REST features in portal might require an enabled PUT and DELETE method. If your web server has these methods disabled, complete one of the following options:
   - Enable HTTP tunneling to simulate PUT and DELETE requests, which means that POST requests are used instead. See the "Switch for tunneling of HTTP methods" link for information.
   - Follow the instructions for your web server to enable PUT and DELETE requests.
5. Start the web server.
6. Optional: If you want to use the short version of vanity URLs, add a rewrite rule to your web server. For more information, read *Providing short vanity URLs*.

Configuring your Domino web server

Configure the communication between IBM WebSphere Portal Express and your IBM Domino web server.

Procedure

1. Edit the `NOTES.INI` file on the web server. Set the **HTTPEnableConnectorHeaders** and **HTTPAllowDecodedUrlPercent** parameters to 1. Also, if you are using WebDAV, enable it in the Domino web server administrative console.
2. Stop the web server.
3. Install and configure the web server plug-in on the system where the web server is located. Use the plug-ins installation wizard that is provided with WebSphere Application Server. Refer to the following topic for information:
   - Linux Windows: Selecting a Web server topology diagram and road map
   - IBM i: Selecting a Web server topology diagram and road map

   Important: Depending on how you use the web server, you must adjust the **ServerIOTimeout** parameter. It defines how long the plug-in must wait for a response from the application. The minimum value is `60` but you must increase this value if you are retrieving data from a database. To update this value, locate and open your `plugin-cfg.xml` file and set **ServerIOTimeout** to an appropriate value. For information, read Common questions about the Web server plug-in.
4. Web 2.0 REST features in portal might require an enabled PUT and DELETE method. If your web server has these methods disabled, complete one of the following options:
   - Enable HTTP tunneling to simulate PUT and DELETE requests, which means that POST requests are used instead. See the "Switch for tunneling of HTTP methods" link for information.

- Follow the instructions for your web server to enable PUT and DELETE requests.

5. Start the web server.
6. Optional: If you want to use the short version of vanity URLs, add a rewrite rule to your web server. For more information, read *Providing short vanity URLs*.

Configuring your Oracle iPlanet web server

Configure the communication between IBM WebSphere Portal Express and your Oracle iPlanet web server.

Procedure

1. Install and configure the web server plug-in on the system where the web server is located. Use the plug-ins installation wizard that is provided with WebSphere Application Server. Refer to the following topic for information:
   - Linux Windows: Selecting a Web server topology diagram and road map
   - IBM i: Selecting a Web server topology diagram and road map

   Important: Depending on how you use the web server, you must adjust the **ServerIOTimeout** parameter. It defines how long the plug-in must wait for a response from the application. The minimum value is `60` but you must increase this value if you are retrieving data from a database. To update this value, locate and open your `plugin-cfg.xml` file and set **ServerIOTimeout** to an appropriate value. For information, read Common questions about the Web server plug-in.
2. If you are using an Oracle iPlanet web server, some portlets require that you disable the **unix-uri-clean** or **nt-uri-clean** directives. Edit the `obj.conf` file to enable or disable these directives. Refer to the Oracle iPlanet web server documentation to determine the appropriate setting for your environment.

   Note: If you are using Oracle iPlanet web server Version 7, you must disable **uri-clean**.
3. Web 2.0 REST features in portal might require an enabled PUT and DELETE method. If your web server has these methods disabled, complete one of the following options:
   - Enable HTTP tunneling to simulate PUT and DELETE requests, which means that POST requests are used instead. See the "Switch for tunneling of HTTP methods" link for information.
   - Follow the instructions for your web server to enable PUT and DELETE requests.
4. Start the web server.
5. Optional: If you want to use the short version of vanity URLs, add a rewrite rule to your web server. For more information, read *Providing short vanity URLs*.

Configuring your OnDemand Router

Configure the communication between IBM WebSphere Portal Express and your OnDemand Router (ODR).

Procedure

1. Log in to the WebSphere Integrated Solutions Console.
2. Go to **Servers** > **Server Types** > **Web server**.
3. Select your ODR.

4. Expand **On Demand Router Properties** in the **On Demand Router Settings** section.
5. Click **Custom Properties** in the **Additional Properties** section.
6. Add a new custom property with the following information:

   Name
   `cache.query.string`

   Value
   `true`
7. Save your changes.
8. Restart your ODR server.

Accessing WebSphere Portal Express through another HTTP port

By default WebSphere Portal Express is configured to be accessed through the internal HTTP port in WebSphere Application Server. For example, `http://`*`hostname.example.com`*`:`*`10039`*`/wps/portal`, where *hostname.example.com* is the fully qualified host name of the server where Portal is running and *10039* is the default transport port that is created by WebSphere Application Server. The port number might be different for your environment. The default host name and port that is used by WebSphere Portal Express are specified by the **WpsHostName** and **WpsHostPort** properties in the `wkplc.properties` file.

About this task

After you configure WebSphere Portal Express to use an external web server, you access the portal with the web server host name and port (for example, *80*). For stand-alone servers or vertical cluster members, you are unable to access the portal with the WebSphere Portal Express host name and port (for example, *10039*). You can access it if there is a corresponding virtual host definition for port *10039* in the WebSphere Application Server configuration.

Many of the WebSphere Portal Express configuration tasks rely on the **WpsHostName** and **WpsHostPort** properties from the `wkplc.properties` file. You must ensure that WebSphere Portal Express can be accessed with the host name and port that is specified by these property values. Choose one of the following methods:

- Modify the **WpsHostName** and **WpsHostPort** property values to specify the web server host name and port.
- Add the appropriate virtual host definition.

If you want to access WebSphere Portal Express with a host name and port different from your web server, add the required virtual host definition with the WebSphere Integrated Solutions Console. In a clustered environment, use the deployment manager WebSphere Integrated Solutions Console to complete these steps.

Procedure

1. Log on to the WebSphere Integrated Solutions Console.
2. Go to **Environment** > **Virtual Hosts**.
3. Select the **default_host** entry or the entry for the virtual host that is being used to access the WebSphere Portal Express application.
4. Select **Host Aliases**, and verify whether there is a host name and port entry corresponding to the values used to access WebSphere Portal Express. If the

entry does not exist, select **New**, and enter the information for the host name and port you want to use. The following information is the host alias examples:

- IBM i Linux Windows: `*:10039`

5. Save your changes.
6. Regenerate the web server plug-in.
7. If you are using a remote web server, copy the updated `plugin-cfg.xml` file to the web server in the web server home directory.
8. If you are running a system under stress and are expecting requests to take longer than the **ServerIOTimeout** default value, increase this value to avoid sending requests twice.
9. Recycle your web server, and your portal.
10. In a clustered environment, resynchronize the nodes and restart the cluster.

WebSphere Portal

First, finish your IBM WebSphere Portal Express deployment with the Configuration Wizard. Then, you can configure your environment further. For example, you can tune your servers to improve performance. You can change the default context root.

"Configuring portal behavior" on page 248
Configure various options related to your portal.

"Changing ports" on page 361
You can change the IBM WebSphere Portal Express ports values after installation if there are port conflicts with other cells on the system.

"Completing the portal URI change started during installation" on page 362
If you changed the context root on the **Configuration for IBM WebSphere Portal: Profile configuration details: Advanced** pane during installation, there are more steps to take to complete the change.

"Changing the portal URI after an installation" on page 368
You can change the default portal Uniform Resource Identifier (URI) any time after you install IBM WebSphere Portal Express. Some applications have a fixed context root that cannot be changed.

"Configuring managed pages" on page 375
When you create a new installation of IBM WebSphere Portal Express 8.5, managed pages are enabled by default. However, you can also manually disable and enable the feature as needed.

"Create a WebSphere Portal profile" on page 382
During the installation process, the IBM Installation Manager creates the WebSphere Portal Express profile. If you are on WebSphere Portal Express Version 8.5 without a combined cumulative fix applied, then you can use this option in the Configuration Wizard to create an additional profile.

"Remove a WebSphere Portal profile" on page 384
Use the Configuration Wizard to remove a portal profile.

"Managing your WebSphere Portal Express environment" on page 386
After you install IBM WebSphere Portal Express, you can use the IBM Installation Manager function to manage your environment. The Installation Manager function consists of updating and modifying the environment. You can also uninstall or roll back the modifications you made to your environment.

"Configuring the IBM License Metric Tool" on page 389
IBM License Metric Tool monitors license compliance. It recognizes and monitors what product offerings and their versions, releases, and fix packs are

installed and used on the system. It measures the processor value units (PVU) available to and used by these assets. The tool ensures compliance with IBM subcapacity licensing requirements and to demonstrate good IT governance. Information about installed software is collected from monitored computers by an agent that can be deployed on a range of operating systems. It is stored on a central server in a DB2 database and can be accessed through pre-configured reports that are available from a web user interface.

"Upgrading your existing product offering" on page 391
After your initial WebSphere Portal Express installation, you can purchase a license for an upgraded product offering.

Configuring portal behavior

Configure various options related to your portal.

"Setting the language of the portal" on page 249
Specify the default language in which the portal appears with the Global Settings portlet.

"Renaming the HTTP session cookie" on page 250
The Java Servlet API up to Version 2.5 states that the session identification cookie must be named `JSESSIONID` . WebSphere Application Server Version 8 supports the Java Servlet API 3.0 that offers applications the option to rename the JSESSIONID cookie name. Therefore WebSphere Portal Express Version 8.5 also supports this option.

"Customizing the home page login URL with the theme " on page 251
In the theme the login link points to a protected URL to the home page of the default portal installation. If you remove this page, or if you want your users to be directed to a different page after login, modify the theme by the following procedure.

"Using portal light mode" on page 251
Portal now provides a portal light mode which can improve portal startup time and reduce memory consumption in production environments.

"Creating or editing a custom unique name" on page 254
You can create a new custom unique name for a portal resource or update an existing custom unique name.

"Setting the portal entry page" on page 254
Use the **Global Settings** portlet to specify the page that a user sees when the user logs in to IBM WebSphere Portal Express.

"Configuring your time settings" on page 255
Some applications, like Calendar, are capable of adapting to the time zone settings of a user. By default, this functionality is disabled. If you want to enable your system to adjust to your local time settings, you must add a mapping to the **ibm-timeZone** attribute. See Adapting the attribute configuration in the Installing section in the Configuring portal to use a user registry topic.

"Setting the search engine that opens when users select Find" on page 255
Specify the search engine that is used when users click **Find** with the Global Settings portlet.

"Configuring how to handle portlets that a user is not authorized to view" on page 256
Specify how you want the portal to display portlets that a user is not authorized to view with the Global Settings portlet.

"Configuring user session persistence" on page 256
With the persistent session state feature, portal users can resume and continue a previously interrupted working session at the same state where they ended the session. When the user logs out or the session times out, the portal stores the

current navigational state into the database. As a portal administrator, you can give users the option to resume the navigational state of their last session when they log in again. When the user chooses to resume the last session, the navigational state that is stored previously is restored, and the user can continue working where the user stopped before.

"Configuring dynamic fragment cache" on page 260
Dynamic fragment cache (also known as servlet caching) is a component of WebSphere Application Server that provides content caching. You configure and enable dynamic fragment cache by using the WebSphere Integrated Solutions Console.

"Configuring portlet filtering" on page 260
A portlet filter enables the administrator of a portal to intercept and modify the output of a portlet before it is aggregated to the entire portal page. This way you can support different languages and markups other than those for which the portlet was originally designed. You can use portlet filters also for adding additional information to the portlet output, for example, a copyright statement, deleting unimportant or restricted content, and for parsing destructive JavaScript.

"Configuring authentication filters" on page 265
The portal authentication filters are a set of plug-in points. You can use them to intercept or extend the portal login, logout, session timeout, and request processing by custom code, for example to redirect users to a specific URL.

"Caching" on page 267
Caching affects the performance of your IBM WebSphere Portal Express environment. Learn about some simple ways to improve the caching performance. After you have reviewed this content, you should also review the WebSphere Portal Express and Web Content Manager Performance Tuning Guide which provides more information about caches for both WebSphere Portal Express and Web Content Manager.

"URL mapping" on page 277
URL mappings were deprecated starting with WebSphere Portal Express Version 8.5. Instead, you can now use friendly URLs or Vanity URLs as an alternative to URL mapping.

"HTTP proxy configuration" on page 278
Some portlets use IBM WebSphere Portal Express resources to support HTTP proxy. Loading and caching remote URLs (such as RSS streams or HTML files) is done in the portal by the URL Manager service. If you specify an HTTP proxy in the configuration of the service, all remote requests are loaded using this HTTP proxy. This feature enables servers behind a firewall with no direct access to the Internet to load external data, such as news or stock information.

"Delayed cleanup of deleted portal pages" on page 278
Get an overview of the cleanup service for portal pages and their dependent resources.

"Deleting orphaned data" on page 280
You use the SLCheckerTool to delete orphaned data in the database.

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Setting the language of the portal

Specify the default language in which the portal appears with the Global Settings portlet.

About this task

Note: The Global Settings portlet does not work in portal cluster configurations. For portal clusters set the portal default language in the portal Localizer service by using the WebSphere Integrated Solutions Console. For details about how to set the language see *Setting service configuration properties*.

Procedure

1. Click the **Administration menu** icon. Then, click **Portal Settings** > **Global Settings**.
2. Select the language for the portal from the **Default portal language** list.
3. Click **Save**.

What to do next

For a list of the languages that are supported by the portal, see *Language support*.

Related concepts:

"Language support" on page 1419
To reach as many users as possible, WebSphere Portal Express supports different languages for different locations. For instance, a large, international corporation might address users in different countries or regions through multilingual Web sites. In this context the portal can concurrently serve portal views to large numbers of users, each in the user's preferred language.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Renaming the HTTP session cookie

The Java Servlet API up to Version 2.5 states that the session identification cookie must be named `JSESSIONID` . WebSphere Application Server Version 8 supports the Java Servlet API 3.0 that offers applications the option to rename the JSESSIONID cookie name. Therefore WebSphere Portal Express Version 8.5 also supports this option.

About this task

A common use case for changing the JSESSIONID cookie name results from cookie name clashes due to HTTP proxy server usage. You can avoid such conflicts by either of the following methods:

Enable HTTP session ID reuse
: This prevents cookie name clashes by reusing the session ID values across different servers. To protect WebSphere Portal Express user sessions, you need to enable security integration in conjunction with session ID reuse. For details see the topic about Session management custom properties under the section about Http Session ID Reuse and the topics about Session security support and Session management settings in the appropriate WebSphere Application Server information center for your portal environment.

Procedure

1. Rename the HTTP session cookie on WebSphere Application Server. To do this, proceed as follows:
   a. Open the WebSphere Integrated Solutions Console.
   b. Select **Servers** > **Application Servers** > **Server_Name** > **Web Container Settings** > **Session management** > **Enable Cookies**.
   c. Change the value for session cookie name as required.
   d. Click **OK**.
   e. Save your changes.
   f. Restart WebSphere Application Server
   g. Regenerate the plug-in configuration file.
   h. If you are running a remote system, copy the plug-in configuration file to the remote server.
2. To synchronize your portal with the changes you made in the previous step, add the required properties to the Resource Environment Providers for your portal:
   a. Open the WebSphere Integrated Solutions Console.
   b. Add the following properties to the Resource Environment Providers:
      - In the **WP ConfigService**, add the following property:
        `cookie.sessionid.name=`*`cookiename`*
      - In the **WP PortletServiceRegistryService**, add the following property:
        `com.ibm.wps.pb.service.PropertyBrokerServiceImpl.sessionid.cookie.names=`*`cookiena`*

      For both properties, replace the variable *`cookiename`* by the new name of the JSESSIONID cookie.

Customizing the home page login URL with the theme

In the theme the login link points to a protected URL to the home page of the default portal installation. If you remove this page, or if you want your users to be directed to a different page after login, modify the theme by the following procedure.

Procedure

1. Open the file *PortalServer_root*`/theme/wp.theme.themes/default80/installedApps/DefaultTheme80.ear/DefaultTheme80.war/themes/html/dynamicSpots/commonActions.jsp` .
2. Locate the line: `<portal-navigation:urlGeneration allowRelativeURL="true" keepNavigationalState="false" contentNode="wps.content.root" home="protected" >`
3. Change `wps.content.root` to the unique name of the page to which you want your users to be directed after logging in.

Using portal light mode

Portal now provides a portal light mode which can improve portal startup time and reduce memory consumption in production environments.

About this task

When you enable portal light mode, specific portlet applications are not started at portal start up. Instead they are started later by the first standard HTTP request that occurs and renders a portal page with the portlet application on the server. This occurs, for example, when a user accesses the portlet application.

The default list of these applications whose initialization is deferred until first use (sometimes called "lazy applications"), contains administrative and sample portlet applications.

To benefit from a higher performance improvement, you can adapt the default list of these applications to your needs.

"Configuring portal light mode"
To benefit from a higher performance improvement, you can adapt the default list of lazy applications to your needs.

"Enabling and disabling portal light mode" on page 253
When you enable portal light mode, a portlet application is not started by a user request, but by the first standard HTTP request that occurs and renders a portal page that contains the portlet application on the server. Direct access to the portlet, for example an Ajax request, does not start the portlet.

Configuring portal light mode:

To benefit from a higher performance improvement, you can adapt the default list of lazy applications to your needs.

Procedure

1. Determine which of the applications that are deployed in your portal you want to configure as lazy applications. To do this procedure, perform the following steps:
   a. Log on to the WebSphere Integrated Solutions Console.
   b. Select **Applications** > **Application Types** > **WebSphere Enterprise Applications**.
   c. Write down the names of the applications that you want to add to the list of lazy applications. Add only applications to the list that are not used by your usual scenarios and that are not required for portal start.
2. Stop the portal server.
3. Make sure that portal light mode is disabled. To disable portal light mode, change to the directory *wp_profile_root*/ConfigEngine and run the configuration task

   ```
   ConfigEngine.sh|bat disable-portal-light-startup-performance -DWasPassword=password
   ```
4. Change to the directory *wp_profile_root*/PortalServer/config/StartupPerformance.
5. Modify the file wp.base_TargetMapInclList.properties. This file contains the list of applications that are not loaded when the portal server is started.
   - To add an application to this list, type the required application names from the **WebSphere Enterprise Applications** list.
   - To remove an application, comment out the appropriate application name or delete it from the list.

   Note: Make sure that your list of lazy applications does not contain any applications that are either required for portal startup or frequently used. Do not add portlet applications to the list that hold portal services or a plug-in for an Eclipse extension point. You can use the white list in the wp.base_TargetMapExclList.properties file as a reference. Never disable any of the applications that are listed in that properties file. For more information, see *Configuring developer mode on Windows*.
6. Save your changes.

7. Enable portal light mode. To do this step, change to the directory *wp_profile_root*/ConfigEngine and run the configuration task
   ConfigEngine.sh|bat **en**able-portal-light-startup-performance -DWasPassword=*password*
8. Restart the portal server.
9. After this adaptation, verify that your scenarios are still running.

What to do next

Limitations:

1. If you have portal light mode that is enabled and you stop an application manually, for example by using the WebSphere Integrated Solutions Console or the wsadmin command-line interface, and a user then accesses that application, that applications are restarted.
2. If you have portal light mode enabled and you use the activation task ConfigEngine.sh|bat activate-portlets to activate all portlets, all portlets are indeed started, even if they are set for lazy load.

Enabling and disabling portal light mode:

When you enable portal light mode, a portlet application is not started by a user request, but by the first standard HTTP request that occurs and renders a portal page that contains the portlet application on the server. Direct access to the portlet, for example an Ajax request, does not start the portlet.

About this task

Follow the appropriate procedures to enable or disable portal light mode.

Procedure

- Enable: To enable portal light mode, change to the directory *wp_profile_root*/ConfigEngine and run the configuration task
  - Linux : ./ConfigEngine.sh enable-portal-light-startup-performance -DWasPassword=*password*
  - Windows: ConfigEngine.bat enable-portal-light-startup-performance -DWasPassword=*password*
  - IBM i: ConfigEngine.sh enable-portal-light-startup-performance -DWasPassword=*password*
- Disable: To disable portal light mode, change to the directory *wp_profile_root*/ConfigEngine and run the configuration task
  - Linux : ./ConfigEngine.sh disable-portal-light-startup-performance -DWasPassword=*password*
  - Windows: ConfigEngine.bat disable-portal-light-startup-performance -DWasPassword=*password*
  - IBM i: ConfigEngine.sh disable-portal-light-startup-performance -DWasPassword=*password*

What to do next

Limitations: When portal light mode is enabled, the following limitations apply:

1. If you manually stop applications, for example by using the WebSphere Integrated Solutions Console or the wsadmin user ID, and they are then accessed by users, the applications are automatically started again.

2. When you use the activation task **activate-portlets** to activate all portlets, it starts all portlets, including the portlets that are set for lazy load.

Creating or editing a custom unique name

You can create a new custom unique name for a portal resource or update an existing custom unique name.

Before you begin

Proceed by the following steps:

About this task

Procedure

1. Select a resource type. Manage Custom Unique Names initially displays a list of resource types. Select a resource type by clicking it, for example, **Portlets**. Manage Custom Unique Names then lists all portal resources of the selected type, for example, all portlets of the portal.
2. Select a resource. From the list of resources, click the **Edit** icon for the portal resource for which you want to assign or change the custom name. Manage Custom Unique Names displays the **Unique name:** entry field for specifying the unique name, together with the identifying information for the selected resource. For example, this can be the portlet title and the Unique Identifier used internally by the portal.

   Note: To select a portlet instance as the resource, click **Pages**, navigate to the page containing the portlet instance, and click **Edit** by the portlet name.
3. Type the unique name that you want to assign to the selected portal resource or update the name as required.

   Notes:

   a. This name must be unique within the portal.
   b. A unique name must not exceed 255 characters in length.
   c. A unique name must not start with an IBM internal prefix, such as `ibm` or `com.ibm` . Otherwise clashes with internal unique names might occur, for example with pages provided with the default portal installation.

   If you want to remove the unique name from the resource, delete the name from the field.
4. Click **OK** to save your updates, or click **Cancel** if you do not want to save the updates. Manage Custom Unique Names returns to the previous panel. If you clicked **OK**, it displays the new name with the resource in the table.
5. Return to the resource type list by clicking **Select type**.

Setting the portal entry page

Use the **Global Settings** portlet to specify the page that a user sees when the user logs in to IBM WebSphere Portal Express.

About this task

Note: The Global Settings portlet does not work in portal cluster configurations.

Procedure

1. Click the **Administration menu** icon. Then, click **Portal Settings** > **Global Settings**.
2. Make a selection from the field **When a user logs in, the first page displayed**:

   Will always be the user's default page (portal session will not resume)
   Choose this option if you want users to always return to the default page after login.

   Will be the page the user most recently visited (portal session will resume)
   Choose this option if you want users to return to the page from their last visit. This option is helpful when users lose their portal session in the middle of a task and need to login to return.

   Will be dependent upon the choice the user makes at login
   Choose this option to let users determine the initial view by their choice after login.
3. Click **Save**.

Results

If you have a portal cluster configuration, use the WebSphere Integrated Solutions Console to set the properties `persistent.session.level` and `persistent.session.option` in the portal configuration service. For more information, refer to the topics about *Configuring user session persistence*, *Portal configuration services, and Setting service configuration properties*.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

"Configuring user session persistence" on page 256
With the persistent session state feature, portal users can resume and continue a previously interrupted working session at the same state where they ended the session. When the user logs out or the session times out, the portal stores the current navigational state into the database. As a portal administrator, you can give users the option to resume the navigational state of their last session when they log in again. When the user chooses to resume the last session, the navigational state that is stored previously is restored, and the user can continue working where the user stopped before.

Configuring your time settings

Some applications, like Calendar, are capable of adapting to the time zone settings of a user. By default, this functionality is disabled. If you want to enable your system to adjust to your local time settings, you must add a mapping to the **`ibm-timeZone`** attribute. See Adapting the attribute configuration in the Installing section in the Configuring portal to use a user registry topic.

About this task

Setting the search engine that opens when users select Find

Specify the search engine that is used when users click **Find** with the Global Settings portlet.

About this task

The Global Settings portlet does not work in portal cluster configurations.

Procedure

1. Click the **Administration menu** icon. Then, click **Portal Settings** > **Global Settings**.
2. In the field **Use this URL for "Find:"**, enter the URL for the search engine that you want to open when a user selects **Find**.
3. Restart the portal.

   The portal now displays a **Find:** button in the theme action bar of user pages.

   Note: If you use any themes with WebSphere Portal Express, you must modify these themes to make the Find function available. To complete this step, include the `<portal:find/>` tag in your theme.
4. Click **Save**.

Configuring how to handle portlets that a user is not authorized to view

Specify how you want the portal to display portlets that a user is not authorized to view with the Global Settings portlet.

About this task

Note: The Global Settings portlet does not work in portal cluster configurations.

Procedure

1. Click the **Administration menu** icon. Then, click **Portal Settings** > **Global Settings**.
2. Make a selection from the field **If a user is not authorized to view a portlet**:

   Portlet is not displayed
   Choose this option if you want nothing to display.

   Portlet is not displayed, replaced by an informative message
   Choose this option if you want a message to be displayed instead of the portlet.
3. Click **Save**.

Results

If you have a portal cluster configuration, use the WebSphere Integrated Solutions Console to set the property `portlets.unauthorized.visible` in the portal Configuration Service.

Configuring user session persistence

With the persistent session state feature, portal users can resume and continue a previously interrupted working session at the same state where they ended the session. When the user logs out or the session times out, the portal stores the current navigational state into the database. As a portal administrator, you can give users the option to resume the navigational state of their last session when they log in again. When the user chooses to resume the last session, the navigational state that is stored previously is restored, and the user can continue working where the user stopped before.

About this task

Example: A user logs in and maximizes some portlets. Afterwards the user logs out. When the user logs in again, all of the previously maximized portlets are still maximized.

"Session settings stored by the portal"
After a user logs out or the session times out, the portal stores the complete navigational state into the database.

"User option during login"
Depending on the configuration defined by the administrator, the user can choose whether to resume the last session or not.

"How administrators define persistent session options" on page 258
As an administrator, you can configure the persistent session behavior

Session settings stored by the portal:

After a user logs out or the session times out, the portal stores the complete navigational state into the database.

The settings include the following navigational state information:

Note: Authenticated and remembered users must have cookies enabled on their browser. Users can access portal sites without cookies enabled if they are anonymous users. If you turn on session tracking for anonymous users, then anonymous users also require cookies.

- Portlet states:
  - Normal
  - Minimized
  - Maximized
- Portlet modes:
  - `config`
  - `edit_defaults`
  - `edit`
  - `view`
  - `help`
- Page selection:
  - The last page that was active before the user logged out.

Note: Resuming the session state is only possible if either the portal theme login link or WebSphere Application Server TAI-based authentication is used to log in to the portal. Logging in or addressing a resource with a URL overwrites the complete session state that is to be resumed and addresses the named resource with the default states and modes for its portlets. For example, this occurs when users use `../wps/myportal/` or `../wps/myportal/`*`name`*`/` for a URL mapping or friendly name.

User option during login:

Depending on the configuration defined by the administrator, the user can choose whether to resume the last session or not.

If the portal administrator has enabled the resume option for users, the login page displays a check box **Resume last session**. If the user selects this check box, the previous session is resumed. Otherwise the previous session is not resumed and the user starts as if logging in for the first time. The options that administrators can set are described in the following sections.

How administrators define persistent session options:

As an administrator, you can configure the persistent session behavior

About this task

You do this by setting the following properties:

persistent.session.option
: This property determines whether the login portlet displays a check box that enables the user to decide whether to resume the session or not. For details refer to the topic about *Giving users the resume option*.

persistent.session.level
: This property determines which navigational state information should be restored when resuming the session. The administrator can choose from three predefined levels. For details refer to the topic about *Setting the session resume level for users*.

timeout.resume.session = (false)
: This property determines whether resuming the session after a session timeout requires user authentication. The default value is `false`. If this property is set to `false` and the user tries to continue working after a session timeout, the portal shows an error message stating that the session has timed out and the user has to log in again. If you set this property to `true`, the portal ignores the session timeout and does not show the error message. The user can resume the previous session without authentication and continue to work. In both cases the previous session is resumed according to the setting of the `persisted.session.level` property described previously.

You set these properties in the portal Configuration Service as described in the topic about *Setting service configuration properties*. The following sections describe the persistent session properties in more detail.

"Giving users the resume option"
As a portal administrator, you can define if users can resume their last session during login.

"Setting the session resume level for users" on page 259
The session resume level specifies which navigational state information is resumed (if any) when the respective user logs in again. As a portal administrator you can configure the session resume level.

Giving users the resume option:

As a portal administrator, you can define if users can resume their last session during login.

About this task

To set the resume option, you set the property `persistent.session.option` in the Portal Configuration Service, as described in the topic about *Setting service configuration properties*. You can configure the property to one of two settings: `0` or `1`.

The two persistent session option values have the following effects:

`persistent.session.option = 0`
This value means that the user does not have the choice to resume the last session or not.

`persistent.session.option = 1`
This value means that at login the user is presented with the option to resume the session in the navigational state of the last session.

The default setting is `0`, that is, users cannot resume their last session.

Notes:

- The defined session preservation settings as described in the topic about *Setting the session resume level for users* are in effect, independent of whether the administrator gave the users the resume option.
- If you give users the resume option, set the session resume level to `1` or higher. Otherwise, the `persistent.session.option` property setting has no effect.

Setting the session resume level for users:

The session resume level specifies which navigational state information is resumed (if any) when the respective user logs in again. As a portal administrator you can configure the session resume level.

About this task

You configure the session resume level by setting the property `persistent.session.level` in the Portal Configuration Service as described in the topic about *Setting service configuration properties*. You can configure the property to one of four predefined values: `0`, `1`, `2`, and `3`.

Note: If you want the user to benefit from the setting, give users the resume option as described in the topic about *Giving users the resume option*. However, the defined setting is in effect, independent of whether you give users the resume option or not.
The four persistent session level values have the following effects:

`persistent.session.level = 0`
This setting means that no persistent session state at all applies to the user session. During logout or session timeout no navigational state information is stored into the database. After a login no navigational state is restored. This value is the default setting.

`persistent.session.level = 1`
The portlet states and the portlet modes are stored in the database and are restored to the user session when the respective user logs in again. For example, all maximized portlets are still maximized. However, no information is stored about the last active page or its render parameters. With this setting, the user starts with the default page after a login.

persistent.session.level = 2
This setting is the maximum level of persistent session state. Using this level, the complete navigational state information is stored. This information includes page selection information and portlet-specific navigational state, such as portlet states, portlet modes, and render parameters. In contrast to persistent session level 1, the session now starts with the last page that was active before the user logged out.

persistent.session.level = 3
If you choose the setting 3, users will stay on the login page after they log in, rather than being redirected to another page. If you set this parameter to 3, this setting does not affect implicit logins, such as single sign-on with LTPA token or through an external security manager.

The default setting is 0, that is, no persistent session state is stored or restored.

The following table gives an overview of the settings and their effect on the user session when the user logs back in to the portal:

Table 28. Persistent session state settings and their effect on resuming user sessions

	Portal navigational state	Session persistence level 0	Session persistence level 1	Session persistence level 2	Session persistence level 3
Portlet states	Normal, minimized, maximized	Not restored	Restored	Restored	Not restored
Portlet modes	configure, edit_defaults, edit, view, help	Not restored	Restored	Restored	Not restored
Render parameters	Not applicable	Not restored	Not restored	Restored	Not restored
Pages	The last active page before the user logged out	Not restored	Not restored	Restored	Not restored

Configuring dynamic fragment cache
Dynamic fragment cache (also known as servlet caching) is a component of WebSphere Application Server that provides content caching. You configure and enable dynamic fragment cache by using the WebSphere Integrated Solutions Console.

About this task

For information about how to enable and configure dynamic fragment cache, go to the WebSphere Application Server topics in the following list. However, complete only the steps in the *Procedure*. Do not proceed with the **What to do next** instructions. Do not configure cacheable objects with the cachespec.xml file.

- Linux : Configuring portlet fragment caching
- IBM i: Configuring portlet fragment caching
- Windows: Configuring portlet fragment caching
- IBM WebSphere Developer Domain - Portal Zone: WebSphere Portal Zone

Configuring portlet filtering
A portlet filter enables the administrator of a portal to intercept and modify the output of a portlet before it is aggregated to the entire portal page. This way you can support different languages and markups other than those for which the portlet was originally designed. You can use portlet filters also for adding

additional information to the portlet output, for example, a copyright statement, deleting unimportant or restricted content, and for parsing destructive JavaScript.

About this task

To use portlet filters, you need to perform all of the following procedures:

- Enable portlet filtering for the portal
- Register the portlet filters, that is define and activate them in a properties file
- Assign the filters to the portlet.

Note: Using this portlet filter only applies to the IBM portlet API. For portlets written against the JSR 286 specification, portlet filtering is already defined within the JSR 286 standard and is configured differently.

"Enabling portlet filtering"
You enable the usage of portlet filters by setting the `legacy.portlet.enable.filtering` property in the Portlet Container Service.

"Registering portlet filters"
Before you can use a portlet filter and assign it to a portlet, you must register it in the `PortletFilterService`.

"Assigning filters to a portlet" on page 262
After you have registered a portlet filtered, you can assign it to a portlet. You can assign multiple portlet filters to a portlet.

"Portlet filter life cycle" on page 263
For performance reasons, portlet filters have a limited life cycle.

"Supported filter targets" on page 263
Calls to the portlet that do not have a request attached are not available to the portlet filter.

"Programming tips: wrapper objects" on page 264
To enable the easy usage of portlet filters, WebSphere Portal Express provides a predefined set of wrapper objects. You can use these wrapper objects to modify the standard behavior of the wrapped components.

"Request flow of portlet filters" on page 264
On server startup, all portlet filters that are registered in the `PortletFilterService` are initialized and are made available for filter registration. After that, the portlet filters go through a sequence of processing steps.

Enabling portlet filtering:

You enable the usage of portlet filters by setting the `legacy.portlet.enable.filtering` property in the Portlet Container Service.

About this task

To enable portlet filtering for the portal, set this property value to `true`:

```
legacy.portlet.enable.filtering = true
```

The default value is `true`.

Registering portlet filters:

Before you can use a portlet filter and assign it to a portlet, you must register it in the `PortletFilterService`.

About this task

The following example code snippet shows the declaration of a filter:

```
1: # Example:
2: filter.SampleFilter.class = com.example.SampleFilter
3: filter.SampleFilter.configValue = some configuration value
4: filter.SampleFilter.transcodeMarkup.1 = html->wml
5: # methods to be filtered
6: filter.SampleFilter.method.1 = service
7: filter.SampleFilter.method.2 = doTitle
```

In this example, the portlet filter `SampleFilter` is defined as follows:

- Line 2 defines the filter name to be `SampleFilter`. This name is later used in the portlet settings to attach the filter to the portlet. Line 2 also defines the class that implements the filter interface.
- Line 3 is a filter specific setting that is given to the filter during initialization. The filter name is removed from the parameter name for later use. For example, the parameter defined in line 4 is later seen by the filter with the name `configValue` and the value `some configuration value`. Several settings of this type can exist.
- Line 4 declares this filter as a transcoding filter that can transcode from HTML to a WML markup. Several settings of this type can exist. If this filter is attached to a portlet that can render HTML, you can now also place this portlet on pages for WML devices. The filter is invoked to transcode the html output to wml as soon as a WML device connects to the portal.
- Line 6 specifies that the filter is called when the `service` method of the portlet is called.
- Line 7 specifies that the filter is called when the `doTitle` method of the portlet is called.

The information in lines 6 and 7 is used to call the filter only for the specified methods to improve the performance. The possible methods for the filter are the following: `login`, `beginPage`, `service`, `endPage`, `doTitle`, `ActionEvent`, `MessageEvent`, and `WindowEvent`. If you want the filter to be called for every method, you do not need to define any method in the properties file.

Assigning filters to a portlet:

After you have registered a portlet filtered, you can assign it to a portlet. You can assign multiple portlet filters to a portlet.

About this task

To assign multiple portlet filters to a portlet, you define the portlet setting `FilterChain` and specify a list of filters separated by commas or semicolons as the value.

To add this setting to the relevant portlet during run time, use the Modify Parameters option in the Manage Portlets portlet.

The request will pass to the filters in the list in sequential order and then proceed to the portlet. The portlet response will then traverse back through all filters in reverse order. Refer to the following example:

```
FilterChain = MyTranscodingFilter, MyAdStripper
```

In this filter chain example, the filters perform the following actions:

- `MyTranscodingFilter` can transcode HTML to WML.
- `MyAdStripper` removes the advertisement from the output of the portlet.

The request processes the following steps:

1. `MyTranscodingFilter` is called first. This filter checks the requested markup and detects that WML is requested. Since the portlet supports only HTML, the client in the request is wrapped by this filter with a client-wrapper, which mimes an HTML client for the portlet. In addition, the response must be wrapped with a wrapper that intercepts the output of the following portlet or filters, in order to work on their output.
2. `MyAdStripper` does not have to modify the request because it only works on the output, but the response must be wrapped to store the output of the portlet.
3. The portlet generates its HTML output.
4. `MyAdStripper` on the output of the portlet and will remove the advertisement.
5. `MyTranscodingFilter` then works on the output of the previous filter and transcodes this output to WML.

You cannot use multiple filters that can transcode from one markup to another in a single filter chain.

You can also declare the filter chain or parts of the filter chain on a global level for all portlets written against the IBM API. To achieve this, you can edit or add a property `FilterChain` containing a comma-separated list of filters in the `PortletFilterService.properties` file, just like in the example given previously (`FilterChain = MyTranscodingFilter, MyAdStripper`). This global filter chain is merged with the one defined on a per-portlet basis, while the global ones are applied before the local ones.

Portlet filter life cycle:

For performance reasons, portlet filters have a limited life cycle.

All filters have the following stages:

Init The filter will be initialized. To get information about the specific filter settings and portal information, the `FilterConfig` is given to the `init` method of the filter. After the call of the init method, the filter will switch to the use mode.

Use Filter requests are handled by the `doFilter` method.

Destroy
The filter is taken out of service and the `doFilter` method will not be called again. This way, system resources can be given back to the portal.

Supported filter targets:

Calls to the portlet that do not have a request attached are not available to the portlet filter.

The following calls to the portlet are available to the filter:

- SERVICE
- DOTITLE
- ACTIONEVENT

- MESSAGEEVENT
- WINDOWEVENT
- LOGIN
- BEGINPAGE
- ENDPAGE

The following calls to the portlet are **not** available to the filter:

- Logout
- AttributeAdded
- AttributeReplaced
- AttributeRemoved
- Init
- InitConcrete
- Destroy
- DestroyConcrete

Programming tips: wrapper objects:

To enable the easy usage of portlet filters, WebSphere Portal Express provides a predefined set of wrapper objects. You can use these wrapper objects to modify the standard behavior of the wrapped components.

Wrappers are defined for the following objects:

Table 29. The wrappers defined for three objects

Object	**Predefined wrapper**
PortletRequest	PortletRequestWrapper
PortletResponse	PortletResponseWrapper
Client	ClientWrapper

A transcoding filter for markups, for example, from HTML to WML needs to use all of these wrappers to emulate the HTML environment that an HTML portlet assumes.

Request flow of portlet filters:

On server startup, all portlet filters that are registered in the `PortletFilterService` are initialized and are made available for filter registration. After that, the portlet filters go through a sequence of processing steps.

1. When a portlet is dispatched, the portlet container gets the required filters from the filter registration and calls these filters successively in the specified order.
2. Before the portlet is rendered, the portlet request and portlet response is forwarded through this chain of portlet filters.
3. Afterwards, the portlet container passes the request and response back through the chain of filters in the reverse order.

If a filter is to manipulate the output of a portlet, it must exchange the actual writer in the portlet response with one that stores data for later changes. This can be implemented by using the wrapper classes delivered with the portal. After the portlet is called and the response is returned, the filter can then manipulate the output of the portlet and write the resulting output to the original writer.

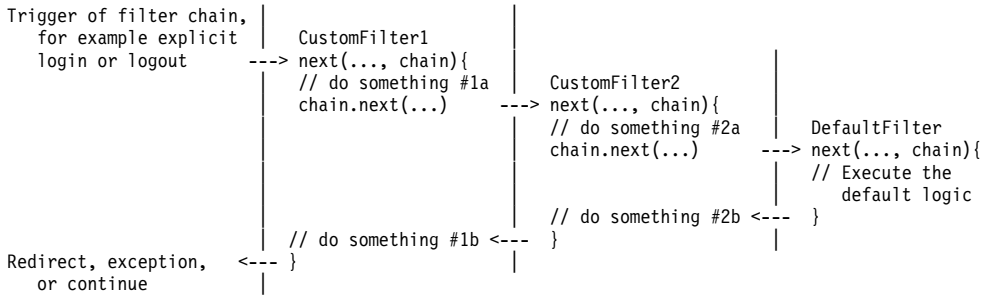
Configuring authentication filters

The portal authentication filters are a set of plug-in points. You can use them to intercept or extend the portal login, logout, session timeout, and request processing by custom code, for example to redirect users to a specific URL.

- "The authentication filter chain concept"
- "Available authentication filter chains "
- "Configuring the filter chains" on page 266
- "Example of a custom authentication filter" on page 266

The authentication filter chain concept

The authentication filters in the portal use the same pattern as defined by the J2EE servlet filter facility. For more information see the web article about The Essentials of Filters. The following example of a code flow shows how this pattern is applied to the authentication filters described in this section.



A default filter performs the default logic for a particular use case, for example, login. You can chain a set of custom filters to be executed before that default filter. When the filter chain is invoked, it calls the first element in the chain (in the example `CustomFilter1`) and passes a chain object as an argument to the call. The filter implementation can then perform some operations before calling the appropriate method on the chain object to trigger the next element in the chain (`CustomFilter2`). This filter again can implement some individual logic that is executed before calling the next element. The last element of the chain is the predefined `DefaultFilter` that makes sure that the default logic for the respective use case is executed.

After a filter has been executed or if an exception is thrown, each filter returns to the one that has called it, so it is possible to implement a customized exception handling or perform additional operations after having called the successor. This way you can chain a custom set of filters. Each custom filter can perform operations before and after the following element(s) in the chain. You can specify the order and the fully qualified class names of the custom filters by portal configuration properties. For details see the topic about the portal WP Authentication Service. The portal provides only the DefaultFilter implementations and enforces that they are always the last element in the chains; if there are no custom login filters defined, the default filters are the only element.

Available authentication filter chains

The filter chain concept described in the previous section is applied to six types of events that concern the flows of Portal login, logout, and session handling. This provides a flexible approach to plug custom logic to each of those flows. In particular, there are filter chains for the following events:

- Explicit login: This is a login by user name and password as represented by the interface `com.ibm.portal.auth.ExplicitLoginFilter`. For example, this can be a login by using the login portlet or the login URL.
- Implicit login: For example, this can be when a user is already authenticated by WAS, but not yet to Portal. This is represented by the interface `com.ibm.portal.auth.ImplicitLoginFilter`.
- Explicit logout: This means that the user triggers a logout action directly, for example by clicking the **Logout** button in the user interface, `interface com.ibm.portal.auth.ExplicitLogoutFilter`.
- Implicit logout: For example, this can be after a session timeout, or if an authenticated user accesses a public page, or if the user navigates to a virtual portal without being member of the associated user realm. This is represented by the interface `com.ibm.portal.auth.ImplicitLogoutFilter`.
- Session Timeout: This is called immediately after an idle timeout of the user session occurred. This is represented by the interface `com.ibm.portal.auth.SessionTimeoutFilter`.
- Session Validation: This is called for every request before actions are triggered and the page is rendered. This is represented by the interface `com.ibm.portal.auth.SessionValidationFilter`.

Besides the session timeout filter, each of the previous filters has access to the HTTP request and response objects. A special context object can be used to share information between filters and set redirects that are executed after the filter chain has been processed. For more detailed information about each of the filter and the filter chain interfaces see the documentation for both WebSphere Portal Express and the API JavaDoc. For a filter chain example see the topic with the Example of a custom authentication filter.

Configuring the filter chains

You can specify the order of filters for each filter chain by setting the following properties in the portal WP Authentication Service:

```
login.explicit.filterchain    = colon or semicolon-separated list of fully qualified class names
login.implicit.filterchain    = colon or semicolon-separated list of fully qualified class names
logout.explicit.filterchain   = colon or semicolon-separated list of fully qualified class names
logout.implicit.filterchain   = colon or semicolon-separated list of fully qualified class names
sessiontimeout.filterchain    = colon or semicolon-separated list of fully qualified class names
sessionvalidation.filterchain = colon or semicolon-separated list of fully qualified class names
```

Note: Use the properties to specify only the custom filter elements, as the default filter implementation is added implicitly by the Portal infrastructure. Thus, by default no value is set for the properties.
In addition, you can set properties in the portal WP Authentication Service according to the following pattern:

`filterchain.properties.`*`fully qualified class name of the filter implementation.property name`*

This makes the value of this property available in the filter configuration object of the specified class by using the key `property name` .

For details about setting portal configuration properties see the topic about Setting service configuration properties.

Example of a custom authentication filter

The following gives an example of a custom filter plugged into the filter chain for the explicit Portal login. The custom filter holds properties that define particular

redirect URLs for particular user IDs and triggers the corresponding redirect if one of those users logged in successfully. To implement such an example, proceed by the following steps:

1. Implement the `com.ibm.portal.auth.ExplicitLoginFilter` interface and make your class available to the portal class path by adding the JAR file to the extended classpath directory of the WebSphere Portal Express application: *PortalServer_root*`/shared/app` . For an example for how to implement the methods of the interface refer to the following code sample:

```
package com.ibm.portal.example;

public class UserRedirectLoginFilter implements ExplicitLoginFilter {

    // hash map to store the mappings from user id to redirect URL
       private java.util.Map userToRedirectURLs = new java.util.HashMap();

public void init(SecurityFilterConfig filterConfig)
       throws SecurityFilterInitException {
    // iterate the list of init parameters and store
       the mappings of user to redirect urls for
       (java.util.Iterator it = filterConfig.getInitParameterNames(); it.hasNext(); ) {
           String currentParameter = (String)it.next();
           userToRedirectURLs.put(currentParameter,
           filterConfig.getInitParameter(currentParameter));
        }
    }

    public void login(HttpServletRequest req, HttpServletResponse resp,
                String userID, char[] password,
                FilterChainContext portalLoginContext, Subject subject,
                String realm, ExplicitLoginFilterChain chain)
                throws LoginException, WSSecurityException,
                PasswordInvalidException, UserIDInvalidException,
                AuthenticationFailedException, AuthenticationException,
                SystemLoginException, com.ibm.portal.auth.exceptions.LoginException {
        // call the next element in the filter chain to trigger the default login
           chain.login(req, resp, userID, password, portalLoginContext, subject, realm);

        // if no exception occured, the login was successful
           if (userToRedirectURLs.containsKey(userID)) {
        // set the redirect url for the user if we have an entry
           portalLoginContext.setRedirectURL((String)userToRedirectURLs.get(userID));
        }
    }

    public void destroy() {
           // nothing to do here
    }
 }
```

2. Specify the class name of the custom filter in the WP Authentication Service properties:

   ```
   login.explicit.filterchain=com.ibm.portal.example.UserRedirectLoginFilter
   ```

3. To define the redirect URLs for individual user IDs, specify your custom set of properties for this class accordingly. Example:

   ```
   filterchain.properties.com.ibm.portal.example.UserRedirectLoginFilter.alice=/wps/myportal/pageA
   filterchain.properties.com.ibm.portal.example.UserRedirectLoginFilter.bob=/wps/myportal/pageB
   ```

4. Restart the portal.

The new filter for the explicit login is now available. Users defined in the properties will be redirected to the specified URL after logging in through the login portlet or login URL.

Caching

Caching affects the performance of your IBM WebSphere Portal Express environment. Learn about some simple ways to improve the caching performance. After you have reviewed this content, you should also review the WebSphere Portal Express and Web Content Manager Performance Tuning Guide which provides more information about caches for both WebSphere Portal Express and Web Content Manager.

The WebSphere Portal and Web Content Manager Performance Tuning Guide provides information about caches for WebSphere Portal Express and Web Content Manager. Choose the latest tuning guide for information:

Essential tuning and performance resources

Use the following files to create an effective cache for your environment:

"Caching pages shared by multiple users"
If you use a proxy server such as WebSphere Edge Server and your system has content that can be shared among multiple users, you can improve performance by caching this shared content.

"**cookie.ignore.regex** parameter" on page 269
Use the **cookie.ignore.regex** parameter to configure which cookies to ignore from the header field. Ignoring these cookies excludes them from the digest computation.

"Cache scope" on page 270
The cache scope determines where the content is cached.

"Expiry time" on page 270
The expiry time determines how long the page is stored in a cache.

"Cache scope and expiry time settings" on page 271
There are resources that contribute to the overall remote cache information on a page.

"Default cache scope and expiry time settings" on page 274
You can set the default cache scope and expiry time settings for WebSphere Portal Express in the portal WP Navigator Service.

"Factors affecting cache scope and expiry time" on page 276
Multiple factors can affect the cache scope and expiry time for a page.

"Cache limitations" on page 276
When tuning your environment to improve performance, review the limitations to ensure success.

"Security Issues" on page 277
Storing authenticated pages in a shared cache introduces security holes. If a malicious user discovered the URL for an authenticated page, that user could read pages containing private information.

"Troubleshooting the cache" on page 277
In general, you should monitor the cache hit rate on the proxy server and adjust the cache size appropriately. If the hit rate is not what you expected, increase the cache size.

Caching pages shared by multiple users:

If you use a proxy server such as WebSphere Edge Server and your system has content that can be shared among multiple users, you can improve performance by caching this shared content.

WebSphere Portal Express allows you to configure the cache scope and the cache expiry time of the specified content. The cache scope and cache expiry time are configured by page, portlet, and theme. WebSphere Portal Express combines this information to produce a final cache scope and expiry time for each page it serves. You can configure these cache settings in one of the following two ways:

The XML configuration interface
See the XML configuration interface for information.

Portlets

The Manage Pages, Manage Portlets, and Properties portlets allow you to configure cache settings. For detailed information, refer to the portlet helps.

Note: When caching JSR portlets, the cache scope is only for proxy server caching policies and requires the use of an edge server cache. Local display caching policies are not affected by this setting. The cache expiration setting is used for both local and remote caching policies.

cookie.ignore.regex parameter:

Use the **cookie.ignore.regex** parameter to configure which cookies to ignore from the header field. Ignoring these cookies excludes them from the digest computation.

About this task

Complete the following steps to add the **cookie.ignore.regex** parameter to the portal resource environment provider:

Procedure

1. If necessary, start the WebSphere_Portal server.
2. Log in to the WebSphere Integrated Solutions Console.
3. Go to **Resources** > **Resource Environment** > **Resource Environment Providers**.
4. Select **WP ConfigService**.
5. Under Additional Properties, select **Custom Properties**.
6. Click **New**.
7. Specify the name of the required property and set the value of the property to the appropriate value as needed.

   To specify cookies that are NOT included in the digest computation, specify `cookie.ignore.regex = digest\.ignore.*|Item1|Item2|Item3|Item4,` where Item1, Item2, Item3, Item4 are the items that you want to exclude from getting cached. The default value is:

   `digest\.ignore.*|${com.ibm.websphere.security.customLTPACookieName}|${com.ibm.websphere.security.customSSOCookieName}|${cookie.session`

   The variables are defined such that the variables resolve to the values listed. For example,

   `digest\.ignore.*|LTPAToken|LTPAToken2|JSESSIONID|WASReqURL` where

   `${com.ibm.websphere.security.customLTPACookieName}= LTPAToken`
   `${com.ibm.websphere.security.customSSOCookieName}= LTPAToken2`
   `${cookie.sessionid.name}=JSESSIONID`

   Important: Any cookie that is set or modified by any component causes the digests in the URL to change, directly affecting the cache of those resources. If a particular cookie is required for some custom code or feature to work but it is not designed to invalidate the cache, that cookie name must be included in the **cookie.ignore.regex** list or at least matched successfully by the regular expression in that property. This process ensures that changes to the cookie value do not have any adverse impact on performance by prematurely invalidating cache entries.
8. Click **Apply** and save your changes.

9. Log out of the WebSphere Integrated Solutions Console.
10. Restart the WebSphere_Portal server.

Related tasks:

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

Cache scope:

The cache scope determines where the content is cached.

There are two types of caching:

Shared cache across users
This type of caching provides the biggest performance improvement. A proxy server caches content and serves requests for the content. This caching eases the load on the server because requests that are served by the proxy do not reach WebSphere Portal Express. If most requests are for customized content, this type of caching provides no performance improvement. The following default values exist for portlet definitions and themes if nothing is provided:

Note: This type of caching should be used only for pages that contain public content that is not personalized.

- Remote cache scope is `non-shared`
- Remote cache expiry is `0 seconds`

Non-shared cache for a single user (Web browser cache)
This type of caching provides a much smaller performance improvement. The cache is typically located in each user's Web browser. This type of caching can be used for all content, including content that is personalized. If the computer is shared among multiple users, a user may see personalized content from other users if served from the browser cache. To prevent this from happening, do not enable private caching, even for personalize content.

Expiry time:

The expiry time determines how long the page is stored in a cache.

WebSphere Portal Express allows three options for specifying expiry time:

Cache always expires
The content is never cached in either a shared or a private cache; set the remote cache expiry to **0**.

Cache never expires
The content can be stored indefinitely in either a shared or a private cache; set the remote cache expiry to **-1**.

Cache expires after this many seconds
The content is stored for the number of seconds specified in either a shared or a private cache; set the remote cache expiry to a positive integer up to `2^31 -1`.

Cache scope and expiry time settings:

There are resources that contribute to the overall remote cache information on a page.

The following information lists the resources that contribute to the overall remote cache information on a **Page**:

remote cache scope

Key com.ibm.portal.remote-cache-scope

Possible values
SHARED, NON_SHARED

Set with XML Access
Yes

Set with user interface
Yes

remote cache expiry

Key com.ibm.portal.remote-cache-expiry

Possible values
Time in seconds, given as an integer between -1 and the value ((2 to the power of 31)-1)

Note: Use the value -1 if you never want the cache to expire.

Set with XML Access
Yes

Set with user interface
Yes

Ignore Access Control in Caches

Key com.ibm.portal.IgnoreAccessControlInCaches

Possible values
True false

Set with XML Access
Yes

Set with user interface
Yes

Example for XML access:

```
<parameter update="set" name="com.ibm.portal.remote-cache-scope" type="string">SHARED</parameter>
<parameter update="set" name="com.ibm.portal.remote-cache-expiry" type="string">3000</parameter>
<parameter update="set" name="com.ibm.portal.IgnoreAccessControlInCaches" type="string">true</parameter>
```

The following information lists the resources that contribute to the overall remote cache information for **Themes**:

remote cache scope

Key com.ibm.portal.remote-cache-scope

Possible values
SHARED NON_SHARED

Set with XML Access
Yes

Set with user interface
No

remote cache expiry

Key com.ibm.portal.remote-cache-expiry

Possible values
Time in seconds, given as an integer between -1 and the value ((2 to the power of 31)-1)

Note: Use the value -1 if you never want the cache to expire.

Set with XML Access
Yes

Set with user interface
No

Example for XML access:

```
<parameter update="set" name="com.ibm.portal.remote-cache-scope" type="string">SHARED</parameter>
<parameter update="set" name="com.ibm.portal.remote-cache-expiry" type="string">3000</parameter>
```

The following information lists the resources that contribute to the overall remote cache information on a **Portlet Definition**:

remote cache scope

Key remote-cache-scope

Possible values
SHARED NON_SHARED

Set with XML Access
Yes

Set with user interface
No

expiration cache

Key EXPIRATION_CACHE

Possible values
Time in seconds, given as an integer between -1 and the value ((2 to the power of 31)-1)

Note: Use the value -1 if you never want the cache to expire.

Set with XML Access
Yes

Set with user interface
Yes

remote cache dynamic

Key remote-cache-dynamic

Possible values
True falso

Set with XML Access
Yes

Set with user interface
No

Note: The Standard Portlet API specification defines the meaning of the `EXPIRATION_CACHE` value. WebSphere Portal Express uses this value to determine the lifetime of the portlet's output in a remote cache, just like the remote cache expiry for themes. The remote cache dynamic setting is an optimization to notify the container whether a portlet window can publish remote cache information at render time. The deployment descriptor specification shows how to deal with these settings.

The following information lists the resources that contribute to the overall remote cache information on a **Portlet Window**:

remote cache scope

Key remote-cache-scope

Possible values
SHARED NON_SHARED

Set with XML Access
No, only published during render time

Set with user interface
No

expiration cache

Key EXPIRATION_CACHE

Possible values
Time in seconds, given as an integer between -1 and the value ((2 to the power of 31)-1).

Note: Use the value -1 if you never want the cache to expire.

Set with XML Access
No, only published during render time.

Set with user interface
No

Note: The portlet definition describes the portlet at a deployment time level given in the portlet deployment descriptor. Thus, attributes specified in the deployment descriptor are valid on all occurrences on all pages of this portlet. The portlet window describes the runtime entity for a portlet. While in the rendering phase of a portlet, the portlet can publish values or attributes via an API. Thus, attributes specified while rendering the portlet is portlet instance specific.
Example code snippet for publishing the information at render time:

```
String paramExpiry = "3000";
String paramScope = "SHARED";
renderResponse.setProperty( "portlet.remote-cache-scope", paramScope );
renderResponse.setProperty( RenderResponse.EXPIRATION_CACHE, paramExpiry );
```

The following information lists the resources that contribute to the overall remote cache information on **Portlet Wide Settings**:

remote cache expiration

Key remote.cache.expiration

Possible values
no, property in `WP NavigatorService`

Set with XML Access
No

Set with user interface

vary

Key vary

Possible values
List of HTTP header fields that can be put into the vary response header

Set with XML Access
No, property in `WP NavigatorService`

Set with user interface
No

Default cache scope and expiry time settings:

You can set the default cache scope and expiry time settings for WebSphere Portal Express in the portal WP Navigator Service.

The following list shows the properties of the portal WP Navigator Service:

`public.session`
Use this property to specify whether an anonymous user always has a public session. This may be useful when a portlet requires a session for anonymous users. the default value is `false` . To enable public sessions for pages that anonymous users can view without logging in, set this property to `true`.

The setting of `public.session` influences the remote cache scope for public pages. If `public.session` is set to `true`, then the cache scope is set to non-shared (private). If `public.session` is set to `false`, then the cache scope is set to shared (public).

Note: Setting `public.session` to `true` might reduce performance.

`public.expires`
Use this property to specify the cache expiration time (in seconds) for caches outside of WebSphere Portal Express and for unauthenticated pages only. These caches must adhere to the HTTP 1.1 specification (RFC 2616). The `public.expires` key specifies the time after which HTTP caches should drop the response. You can further restrict this time by the `remote.cache.expiration` key.

This value is used as a maximum value for the cache expiration time and as a global default value for unauthenticated pages. If you also set the property `remote.cache.expiration` to a value greater than or equal to zero (`0`), the smaller one of the two values is used.

WebSphere Portal Express calculates and aggregates the remote cache information, that is the scope and expiration time, by a number of properties contributed by themes, pages, and portlets besides the properties described here. Therefore WebSphere Portal Express can do any of the following internally while processing a request:

- Reduce the cache lifetime
- Reduce the cache scope, for example, from public (shared) to private (non-shared)
- Switch off the overall cachability of pages.

Therefore this value might not be static for all responses resulting from requests to unauthenticated pages.

The response of WebSphere Portal Express sets the following header fields:

- The `Expires` header with the expiration time added to the system date and time.
- The `Cache-Control : max-age =` header with the expiration time as its parameter.

The default value specified for this property is `60` seconds. If no value is specified, WebSphere Portal Express defaults the value to `60` seconds.

`remote.cache.expiration`
This property specifies the maximum cache lifetime of a page, both public and private, in seconds. Use this property to specify a global value for the expiration of pages in remote caches. Setting this value to zero (`0`) switches caching off in remote caches. If the legacy setting is not available, this property is used for authenticated and unauthenticated pages. If the legacy setting is available, then the smaller of the two values is used for unauthenticated pages only. In this case the `remote.cache.expiration` property is used for authenticated pages in general. If theme, composition, and portlets contribute remote cache information, then the global settings also contribute to the information. In this case the lowest of the values of all contributors is used, including the global settings.

The default value for this property is `60` seconds. If no value is specified, WebSphere Portal Express defaults the value to zero (`0` seconds).

`remoteCacheInfo.response.header.vary`
This property specifies the HTTP headers that force a proxy to cache different variants of the same URL. Use this property to specify a comma separated list of HTTP header fields to which WebSphere Portal Express should refer in its vary field of the generated HTTP response. This is required to ensure that proxy caches can invalidate entries in their cache if the specified header fields do not match from request to request. The default for this property is `User-Agent` .

`public.cache-control`
This property specifies the HTTP headers that force a proxy to cache different variants of the same URL. Use this property to specify a comma separated list of HTTP header fields to which WebSphere Portal Express should refer in its vary field of the generated HTTP response. This is required to ensure that proxy caches can invalidate entries in their cache if the specified header fields do not match from request to request. The default for this property is `no-cache` .

`private.cache-control`
This property specifies the value that is set for the `cache-control` HTTP header field when the portal generates a response in request for private pages. This header field controls the behavior of all caching mechanisms along the request-response chain. The default for this property is `no-cache` .

Factors affecting cache scope and expiry time:

Multiple factors can affect the cache scope and expiry time for a page.

The remote-cache-scope and remote-cache-expiration of a rendered page view is calculated as the minimum of the following factors:

- Cache scope and expiry time specified for the page
- Cache scope and expiry time of the portlets on the page

  Note: If any of the portlets on a page can only be cached in a private cache, then the entire page can only be cached in a private cache. A page cannot be stored in a shared cache unless all portlets on the page can also be stored in a shared cache. Make sure cache settings for portlets and pages are consistent.
- Cache scope and expiry time of the theme
- Global values as defined in the Navigator Service in the WebSphere Integrated Solutions Console.

Cache limitations:

When tuning your environment to improve performance, review the limitations to ensure success.

- For best results, use WebSphere Application Server Edge Components version 6.0.0.1 (or later), version 5.1.0.7 (or later), version 5.0.2.30 (or later) for the proxy server. Previous versions do not serve multiple markup types from the same cache. If there are multiple requests for the same page, but with different markup, the cache is not used. These versions of WebSphere Application Server Edge Components correct this problem.
- If your WebSphere Portal Express serves only one markup, make sure the vary is set appropriately. If your WebSphere Portal Express serves multiple markups, set the vary appropriately and use a larger cache size. Use `remoteCacheInfo.response.header.vary = space separated list of other http header fields` to appropriately set the vary.

  Note: Enter any HTTP header field names; you must use the HTTP 1.1 specification. The two most common HTTP headers to specify here are `vary = accept-language user-agent`.

  Note: The Vary value indicates the set of request-header fields that force a proxy to cache different variants of the same URL.
- If your WebSphere Portal Express serves only one language, make sure the vary is set appropriately. If your WebSphere Portal Express serves multiple languages, set the vary appropriately and use a larger cache size. Use `remoteCacheInfo.response.header.vary = accept-language` to appropriately set the vary.
- Only JSR portlets can override the cache lifetime setting at run time.

Note: With the previous settings, it is possible to generate an HTTP response header like `Cache-Control: max-age=-1`, which indicates an unlimited cache expiration when a page is rendered. This is beyond the HTTP 1.1 specification but if a proxy cache does not support an unlimited cache expiration, WebSphere Portal Express supports it. If the cache infrastructure does not properly work with this response header, set the `remote.cache.expiration` value in the **`WP NavigatorService`** to a large value. To set an unlimited cache expiration is not possible if the cache infrastructure does not support it.

Security Issues:

Storing authenticated pages in a shared cache introduces security holes. If a malicious user discovered the URL for an authenticated page, that user could read pages containing private information.

By default, WebSphere Portal Express does not permit shared caching for authenticated pages. You can use the Properties portlet or the XML configuration interface to override these default settings using the **com.ibm.portal.IgnoreAccessControlInCaches** parameter, but in most cases this is not recommended.

Note: In some rare circumstances, it might be useful to store authenticated pages in a shared cache. For example, if all authenticated users receive identical content, then storing authenticated pages in a shared cache might be acceptable.

Troubleshooting the cache:

In general, you should monitor the cache hit rate on the proxy server and adjust the cache size appropriately. If the hit rate is not what you expected, increase the cache size.

You can also check the following item:

Ensure that the cache settings for all portlets on the page are consistent with the cache settings for the page. For example, if one portlet on a page is set to only be cached in a private browser cache, then the entire page can only be cached in a private browser cache, and performance is not optimal.

URL mapping

URL mappings were deprecated starting with WebSphere Portal Express Version 8.5. Instead, you can now use friendly URLs or Vanity URLs as an alternative to URL mapping.

About this task

Depending on your requirements, you can use vanity URLs, or friendly URLs:

- If you want to have a short URL as an entry point to a specific portal page or content item, use a vanity URL.
- If you want to have a friendly URL that your site visitors see when the portal shows the page, use a friendly name.
- If you want to be able to publish the page through the Web Content Manager workflow, use a vanity URL. For example, this URL can be useful for a marketing campaign.
- If you want to address a specific portal page through URL generation tags or APIs, use unique name IDs. For more information, see *URL generation in WebSphere Portal*.

You can create both vanity URLs and friendly URLs for the same portal page.

Related concepts:

"Vanity URLs" on page 2094
You can associate vanity URLs with portal pages and labels. Vanity URLs are short URLs that people can easily remember. They are shorter than full WebSphere Portal Express URLs. They are sometimes also called marketing URLs. You can publish vanity URLs for marketing campaigns through different channels, such as email or print. This way, you can use vanity URLs to direct customers to a specific portal page or content item. Interested site visitors who want to view your campaign can then remember or copy the short vanity URL and type it into the browser address field.

Related tasks:

"Using friendly URLs" on page 1280
You can associate friendly URLs with portal pages and labels. You and your users can use these friendly URLs to access specific portal pages or labels by using a human readable path, which is easy to remember.

HTTP proxy configuration

Some portlets use IBM WebSphere Portal Express resources to support HTTP proxy. Loading and caching remote URLs (such as RSS streams or HTML files) is done in the portal by the URL Manager service. If you specify an HTTP proxy in the configuration of the service, all remote requests are loaded using this HTTP proxy. This feature enables servers behind a firewall with no direct access to the Internet to load external data, such as news or stock information.

To configure WebSphere Portal Express for HTTP proxy, you use the `WP PortletServiceRegistryService` property in the portal WP Content Access Service in the WebSphere Integrated Solutions Console. For more information, see the topic about Content Access Service and go to the section about Proxy protocol and port settings.

Delayed cleanup of deleted portal pages

Get an overview of the cleanup service for portal pages and their dependent resources.

Portal resources, such as pages, components or portlet instances are kept persistent in the portal database. When an administrator deletes a page, all its derived pages and dependent resources and content are deleted with it. The actual deletion can take considerable time, depending on the size of the portal and the number of resources affected by the deletion. Therefore, if the deletion takes place immediately after the user completes the deletion task, this might impact portal performance for users. On the other hand, if the deletion is delayed and scheduled for an off peak time, it will not affect portal response time and thereby user experience.

The delayed deletion of pages is performed by a cleanup service.

"Configuring immediate or delayed deletion of portal pages" on page 279
You can configure the deletion cleanup to happen either immediately when you delete the page or later.

"Configuring your own delayed deletion schedule by using the XML configuration interface" on page 279
You can use the IBM WebSphere Portal Express XML configuration interface to configure the delayed deletion schedule according to your requirements. You can define a daily, weekly, or monthly schedule. You can also use the XML configuration interface to run individual cleanup tasks at arbitrary intervals.

Configuring immediate or delayed deletion of portal pages:

You can configure the deletion cleanup to happen either immediately when you delete the page or later.

About this task

- **Immediate deletion**. This means that the page and all resources that depend on it are deleted immediately after the user completes the action for the deletion.
- **Delayed deletion**. This means that the page is marked for deletion, but the page and all dependent resources are actually deleted later.

  Note: Once the page has been marked for deletion, users cannot view or otherwise access the page any longer.

You can change between the immediate and delayed deletion of portal pages by configuring the property value `scheduler.cleanup.enabled` in the portal Data Store Service in the WebSphere Integrated Solutions Console. Set the following Data Store Service configuration parameters as required:

scheduler.cleanup.enabled = (true)

Determines whether deletion of portal pages is performed later by the scheduled cleanup service, or immediately after the user completes the deletion task. This affects the deletion of portal pages and all their dependent resources, such as components and portlet instances.

true This setting enables delayed deletion of portal pages by the scheduled cleanup service. Pages and dependent resources are deleted by the scheduled cleanup service.

false This setting disables deletion of portal pages by the scheduled cleanup service. Deletion of the pages and their dependent resources is triggered immediately when the administrative user runs the cleanup task.

Set this property to `true` if you want the deletion of pages to be delayed and performed by the scheduled cleanup service. This property defaults to true, if the portal installation is based on a version of WebSphere Application Server that includes the Scheduler service.

By its default schedule configuration, the cleanup service runs weekly, on Saturdays at 8 pm.

Configuring your own delayed deletion schedule by using the XML configuration interface:

You can use the IBM WebSphere Portal Express XML configuration interface to configure the delayed deletion schedule according to your requirements. You can define a daily, weekly, or monthly schedule. You can also use the XML configuration interface to run individual cleanup tasks at arbitrary intervals.

Procedure

1. Edit the following sample XML configuration file:
   - IBM i: *PortalServer_root*`/PortalServer/doc/xml-samples/Task.xml`
   - Linux: *PortalServer_root*`/PortalServer/doc/xml-samples/Task.xml`
   - Windows: *PortalServer_root*`\PortalServer\doc\xml-samples\Task.xml`
2. Uncomment and edit the entry that corresponds to the scheduled time that you want to set.

Note: By default, the `Task.xml` file is set to run the scheduler immediately one time.

3. Save your changes.
4. Import the modified `Task.xml` file by using the XML configuration interface (XMLAccess).

What to do next

Notes:

1. For your customized schedule to be observed by the portal, you must enable the **`scheduler.cleanup.enabled`** property by setting it to `true` in the WP Data Store Service. You do so in the resource environment providers in the WebSphere Integrated Solutions Console. For more information, read *Configuring immediate or delayed deletion of portal pages*. For more information about portal configuration properties and how to set them, read *Setting service configuration properties* and *Portal service configuration*.
2. If you delete a page with an object ID and then re-create it with XML configuration interface, you might receive an error message. The message indicates that the operation was canceled because it would cause a duplicate key value.
3. When you run the cleanup task, the XML configuration interface schedules only the task to be run in WebSphere Application Server and returns. It does not mean that WebSphere Application Server runs the task immediately. To determine when a task started and ended, check the `SystemOut.log` log file for the messages EJPDE0005I and EJPDE0006I. These messages confirm that the cleanup task successfully completed. After you confirm, you can run the XML script for re-creating a page with the same object ID as it had before the deletion.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Related reference:

"Portal service configuration" on page 289
IBM WebSphere Portal Express comprises a framework of configuration services to accommodate the different scenarios that portals of today need to address. You can configure some of these services.

Deleting orphaned data

You use the SLCheckerTool to delete orphaned data in the database.

Before you begin

When portal resources are deleted, dependent resources that are stored in a different database domain are not deleted at the same time. These remaining resources are still available for backup scenarios or other production lines that might share the database domain. For example, when an administrator deletes pages, the user customization to the deleted pages are not deleted. The SLCheckerTool enables the detection of orphaned data that is not needed any more by any of the production lines that share the domain. You can use the SLCheckerTool to prepare an XML script for later deletion of the orphaned data by the XML configuration interface.

The SLCheckerTool is included in the file wp.db.slchecker.jar in the following directory:

- Linux : *wp_profile_root*/PortalServer/bin/slcheckertool.sh
- IBM i: *wp_profile_root*/PortalServer/bin/slcheckertool.sh
- Windows: *wp_profile_root*\PortalServer\bin\slcheckertool.bat

You start the SLCheckerTool by using the following shell scripts:

- Linux : *wp_profile_root*/PortalServer/bin/slcheckertool.sh
- IBM i: *wp_profile_root*/PortalServer/bin/slcheckertool.sh
- Windows: *wp_profile_root*\PortalServer\bin\slcheckertool.bat

To prepare and complete deleting orphaned data, you also complete some steps by using the XML configuration interface. For details about the XML configuration interface and how to use it refer to the appropriate topics. A sample XML script for exporting orphaned data that are listed enables the preparation of the orphaned data for work with the SLCheckerTool.

About this task

Notes:

1. Before you delete the orphaned data by using the SLCheckerTool, secure your databases by making a backup.
2. To delete all orphaned data, you must include every production line that shares the database domain. Otherwise, resources that are still valid in a skipped production line might be unintentionally removed.

To delete the orphaned data, proceed by the following steps. For each step, use the target file from the previous step as the source file.

Procedure

1. Run the cleanup service to delete resources that are marked for delayed deletion. To run the cleanup service, use the XML configuration interface and the XML sample file Task.xml as the input.
2. Create a full export that includes orphaned data of each production line that shares the particular domain. To complete this step, use the XML configuration interface and the XML sample file ExportIncludingOrphanedData.xml.
3. Use the SLCheckerTool to create an XML script file to delete the orphaned data. Complete this step with the substeps given later. Observe the following rules:
   - For each step, start the SLCheckerTool with the appropriate parameters as described under that step.
   - Replace all file name variables (given in italics) by the complete directory path location and file name.
   - For each step, use the target file or files from the previous step as the source file or files.

   Start the SLCheckerTool by using the following shell scripts:
   - Linux : *wp_profile_root*/PortalServer/bin/slcheckertool.sh
   - IBM i: *wp_profile_root*/PortalServer/bin/slcheckertool.sh
   - Windows: *wp_profile_root*\PortalServer\bin\slcheckertool.bat

a. Find candidates for orphaned data in each production line. Repeat this step for each production line, but specify a different output file for each iteration. Otherwise, the results are overwritten. Use the following parameters:

   --find-candidates -s *xml_source_file* **-d** *cand_target_file* **-domain** *domain_identifier*

   Use these parameters to find the candidates for orphaned data within one production line. Replace the variables as follows:

   xml_source_file

   Specify a full XML export file that you created in step 2, the step for creating the XML export including orphaned data.

   cand_target_file

   Specify the target file. The orphaned data candidates are saved to that file.

   domain_identifier

   Specify the database domain in which you want candidates to be searched. Valid values are `comm` for the community database domain, `cust` for the customization database domain or `all` for both database domains.

b. Identify the orphaned data. Complete this step once, but with all the candidate files generated by previous step at the same time. This step determines the intersection of all result files, that is, the data that are orphaned in all production lines. Use the following parameters:

   --identify-orphans -s cand_source_files_and_directories -dorph_target_file

   Use these parameters to identify the orphaned data by matching the information from all the candidate files. Replace the variables as follows:

   cand_source_files_and_directories

   Specify all files that were generated as `cand_target_files` by the substep for finding the candidates. If you specify one or more directories with the files, make sure that these directories contain only candidate files and no other files.

   orph_target_file

   Specify the target file. The identified orphaned data is saved to that file.

c. Generate an XML script file. You can later use that script for deleting the orphaned data. Use the following parameters:

   --delete-orphans -s *orph_source_ file* **-d** *xml_target_file*

   Use these parameters to generate an XML script file for deleting the orphaned data. Replace the variables as follows:

   orph_source_ file

   Specify the file that was generated as the `orph_target_file` in the substep for identifying the orphaned data.

   xml_target_file

   Specify the target file. This file will be the XML script file that contains the information about the orphaned data. You can later use this file to delete the orphaned data.

4. Optional: You can check whether all production lines were considered during the creation of the XML script file. To check, review the comment in the file header. The header contains information about all full exports that were used.
5. Delete the orphaned data by starting the XML configuration interface with the XML script that you obtained as the `xml_target_file` with the orphaned data in the substep for generating the XML script. You must start the XML configuration interface with the XML script only on one production line that shares the database domain. This step deletes the orphaned data for all production lines that share that database domain.

Results

You removed all orphaned data from your portal database.

Related tasks:

"Preparing the deletion of orphaned resources" on page 1081
You can use the XML configuration interface to delete orphaned data from your WebSphere Portal Express.

Setting service configuration properties

IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

About this task

The configuration for each service is stored in and accessible through the WebSphere Integrated Solutions Console. Each service is registered as a separate resource environment provider with custom properties that represent the service configuration. Initially in a default installation, only the most common properties are shown as custom properties. You might need to add more properties with key and value as required. Or you might want to remove properties that can be used with their default values or are no longer required.

Procedure

To set configuration service properties, proceed as follows:

1. Select the appropriate WebSphere Integrated Solutions Console, depending on your environment:
   - If your portal runs stand-alone, use the local console.
   - If your portal is installed in a cluster, use the console of the deployment manager.
2. Start the WebSphere Integrated Solutions Console by entering the following string in the **URL location** field of a web browser:

   `http://example.com:admin_port/ibm/console`

   where *example*`.com` is the name of your server and *admin_port* is the port that is assigned to the WebSphere Integrated Solutions Console.
3. Go to **Resources** > **Resources Environment** > **Resource Environment Providers**.
4. In the Resource Environment Providers page, make the appropriate selection. Select the appropriate node or cluster from the scopes list, or clear the **Show Scope** check box and select one of the following options, depending on your portal environment:

- If your portal is running as a single server, select **Browse Nodes** and select the node.
- If your portal is installed in a cluster, select **Browse Clusters** and select the portal cluster.

5. Select the service in which you want to change a property.

   Note: In the list, the service names are preceded by a product prefix and a blank space. For example, the WebSphere Portal Express configuration service is identified as `WP ConfigService`. The Web Content Manager configuration service is identified as `WCM WCMConfigService`.
6. Click **Custom Properties**.
7. Do one of the following tasks as needed:
   - To set a property, select that property and change its value.
   - If the property that you want to set does not exist yet, create it new. When you create a new property, use `java.lang.String` as its type and do not mark the property as **required**. Otherwise, you might not be able to delete it later.
   - Select one or more properties for removal.
8. When you are done, click **Save** at the start of the page under **Message(s)**.
9. Click **Save** again when prompted to confirm your changes.
10. If you have a cluster configuration, replicate your changes to the cluster.
11. Restart the server for the changes to become effective.

By alternative, you can also set the properties in the properties files and then enable them by using a configuration task.

Notes:

- This option for setting service configuration properties is not available for all properties.
- Changes to properties configuration files do not affect runtime properties until you run the configuration tasks that are described in the following procedure.

12. Locate the properties file for the appropriate Resource Environment Provider. The properties files are in the directory *wp_profile_root*`/PortalServer/config`. If there is no default properties file for a Resource Environment Provider, for example, WP_DynamicContentSpotMappings, create a new properties file as follows:
    a. Create a properties file by using a text editor.
    b. Give the file the name of the Resource Environment Provider without the `WP` prefix. Save the file in the directory *wp_profile_root*`/PortalServer/config`. Example:

       wp_profile_root`/PortalServer/config/DynamicContentSpotMappings.properties`
13. Edit the properties file and modify or add configuration properties as needed. Example:

    ```
    newDynamicContent=res:/CustomThemeContext/themes/html/MyTheme/dynamicContent.jsp
    ```
14. Save the updated properties file.
15. Run the appropriate configuration task to update the configuration properties: For changes to WebSphere Portal Express properties files to take effect, run the following task from the *wp_profile_root*`/ConfigEngine` directory.
    - Windows: `ConfigEngine.bat update-properties -DPortalAdminPwd=`*password* `-DWasUserid=`*username* `-DWasPassword=`*password*

- Linux : `./ConfigEngine.sh update-properties -DPortalAdminPwd=password -DWasUserid=username -DWasPassword=password`
- IBM i: `ConfigEngine.sh update-properties -DPortalAdminPwd=password -DWasUserid=username -DWasPassword=password`

For changes to Web Content Manager properties files, run the following task from the *wp_profile_root*/ConfigEngine directory.

- Windows: `ConfigEngine.bat update-wcm-service-properties -DPortalAdminPwd=password -DWasUserid=username -DWasPassword=password`
- Linux : `./ConfigEngine.sh update-wcm-service-properties -DPortalAdminPwd=password -DWasUserid=username -DWasPassword=password`
- IBM i: `ConfigEngine.sh update-wcm-service-properties -DPortalAdminPwd=password -DWasUserid=username -DWasPassword=password`

Results

Your service configuration properties updates are now in effect.

"Overview of configuration services" on page 287
Get an overview of the WebSphere Portal Express configuration services available for the portal.

"Portal service configuration" on page 289
IBM WebSphere Portal Express comprises a framework of configuration services to accommodate the different scenarios that portals of today need to address. You can configure some of these services.

"Web Content Manager service configuration" on page 353
Configuration services for IBM Web Content Manager contain settings for the general operation of the web content system, including settings for messaging, pre-rendering, and searching.

Related concepts:

"Pre-render methods" on page 2090
Pre-rendering can be configured to run automatically, or you can manually pre-render a website by using a URL.

"Syndication properties" on page 454
You can tailor the syndication behavior of your web content environment by changing configuration settings such as the syndication interval and automatic syndication.

"Web content authoring options" on page 396
You can tailor the authoring behavior of your web content environment by changing configuration settings such as workflow, profiling, and version control.

"Data cache configuration" on page 425
Data caching is used to cache data that is retrieved by the IBM Web Content Manager application from external sources that use connect tags or by requests that are made through URLs.

http://www-10.lotus.com/ldd/portalwiki.nsf/dx/Web_content_cache_configuration_wcm8

"How to configure authoring tools components" on page 1895
Authoring tools components that are rendered in a web content viewer allow you to create, read, edit, delete, approve, or reject content items directly in the web content viewer, instead of requiring you to browse to the IBM Web Content Manager authoring portlet to run the same action. The web content viewer either opens a window from the current page or redirects the user to another portal page that contains the authoring portlet.

"How to configure authoring portlet search" on page 402
You can change the configuration of the authoring portlet to change how search works.

"How to access the pre-rendered site" on page 2091
Pre-rendered sites are accessed either through IBM Web Content Manager, or through a web server.

"Pre-rendering options" on page 425
You can enable pre-rendering so that content can be viewed either through a IBM Web Content Manager application or as a stand-alone site that is accessed through a web server.

"Creating category selection trees" on page 1823
You use category selection trees to allow users to personalize menus.

"Text, rich text and HTML elements" on page 1812
You use the short text, text, rich text, and HTML elements to store blocks of text, but each has slightly different properties.

"Syndication troubleshooting" on page 459
If you encounter issues when syndicating, there are some common methods available to troubleshoot these issues.

"User roles and access" on page 1560
Different users will have different access to items and functions in your system depending on the role they are assigned. Roles can be assigned at the library level, and also assigned on individual items.

 Using the web content member fixer task

"Cache tuning for federated documents" on page 410
The federated documents feature uses the document list cache, the document data cache, and the feed type cache to manage information about the list of documents, the document data, and the types of feeds a server provides.

Related tasks:

"Enabling connect tags" on page 435
Enable connect tags to reference web content components and apply customized caching to the components.

"Enabling email" on page 435
To use the email workflow action, you must configure Web Content Manager to use your SMTP server.

Using a page navigation element
A page navigation element provides navigation controls that are used to browse through a set of results that are generated by menus, navigators, personalization, and search elements.

"Web Content Manager" on page 392
Set up a content server by installing IBM Web Content Manager in various deployments to provide robust and flexible environments for web content development and delivery. After you install the content server, more configuration steps must be completed according to the role that the server plays in your web content environment.

"Configuring remote server access for links" on page 404
Before you can add links to files and documents that are stored in remote content management systems into web content elements, you must configure your server with information about the remote system and the settings that are used to handle communication with the system.

"Configuring access to remote systems for federated documents" on page 406
To retrieve metadata information for documents on remote content management systems, configure the federated documents feature with information about the remote system and the settings that are used to handle communication with the system.

"Filtering the content model" on page 2869
By applying filters to the content model, you can exclude parts of the page hierarchy from the content model. Filtering is performed based on request data and metadata assigned to the pages.

Overview of configuration services:

Get an overview of the WebSphere Portal Express configuration services available for the portal.

The following table lists the configuration services:

1. The first column gives the names of the services and provides links to the respective topics for those services that you can configure in the WebSphere Integrated Solutions Console.
2. The second column gives the name of each service by which you can access the service in the WebSphere Integrated Solutions Console.
3. The third column lists the related properties files.

Note: The configuration for each service is stored in the WebSphere Integrated Solutions Console. You can change the configuration only there.

Table 30. WebSphere Portal Express configuration services

WebSphere Portal Express service	Service name in the WebSphere Integrated Solutions Console	WebSphere Portal Express properties file
"Administrator Unique Names Mapping Service" on page 291	WP AdminUniqueNamesMappingService	AdminUniqueNamesMappingService.properties
"Cache Manager Service" on page 292	WP CacheManagerService	CacheManagerService.properties
"Common Component Configuration Service" on page 295	WP CommonComponentConfigService	CommonComponentConfigService.properties
"Configuration Service" on page 297	WP ConfigService	ConfigService.properties
"CP Configuration Service for tagging and rating" on page 312	WP CPConfigurationService	CPConfigurationService.properties
"Content Access Service" on page 318	WP PortletServiceRegistryService See "Content Access Service" on page 318.	PortletServiceRegistryService.properties See "Content Access Service" on page 318.
"Data Store Service" on page 320	WP DataStoreService	DataStoreService.properties
"Deployment Service" on page 322	WP DeploymentService	DeploymentService.properties
"HTTP Client Service" on page 325	WP HTTPClientService	HTTPClientService.properties
"Live Object Service" on page 326	WP LiveObjectService	LiveObjectService.properties
"Loader Service" on page 327	WP LoaderService	LoaderService.properties
"Localizer Service" on page 328	WP LocalizerService	LocalizerService.properties
"Model WebDAV Service" on page 328	WP ModelWebDAVService	ModelWebDAVService.properties
"Navigator Service" on page 328	WP NavigatorService	NavigatorService.properties
"Portlet Container Service" on page 346.	WP PortletContainerService	PortletContainerService.properties
"Project Identification Service" on page 346	WP ProjectIdentificationService	ProjectIdentificationService.properties

Table 30. WebSphere Portal Express configuration services (continued)

WebSphere Portal Express service	Service name in the WebSphere Integrated Solutions Console	WebSphere Portal Express properties file
"Registry Service" on page 346	WP RegistryService	RegistryService.properties
"State Manager Service" on page 347 See also "URL normalization for search of portal pages by external search engines" on page 350	WP StateManagerService	StateManagerService.properties
"Virtual Portal Configuration Service" on page 352	WP VirtualPortalConfigService	VirtualPortalConfigService.properties

Table 31. WebSphere Portal Express security services

WebSphere Portal Express security service	Service name in the WebSphere Integrated Solutions Console	WebSphere Portal Express properties file
"Authentication Service" on page 331	WP AuthenticationService	AuthenticationService.properties
"Credential Vault Service" on page 332	WP VaultService	VaultService.properties
"Portal Access Control Services" on page 333		
"Access Control Data Management Service" on page 334	WP AccessControlDataManagementService	AccessControlDataManagementService.properties
"External Access Control Service" on page 336	WP ExternalAccessControlService	ExternalAccessControlService.properties
"Auditing Service" on page 340	WP AuditService	AuditService.properties
Access Control Service	WP AccessControlService	AccessControlService.properties
Access Control WarmUp Service	WP AccessControlWarmUpService	AccessControlWarmUpService.properties
PAC Group Management Service	WP PACGroupManagementService	PACGroupManagementService.properties
"Puma Store and Validation Services" on page 343		
"Puma Store Service" on page 343	WP PumaStoreService	PumaStoreService.properties
"Puma Validation Service" on page 344	WP ValidationService	ValidationService.properties

Table 32. Other WebSphere Portal Express configuration services

WebSphere Portal Express service	Service name in the WebSphere Integrated Solutions Console	WebSphere Portal Express properties file
Credential Type Registry Service	WP CredentialTypeRegistryService	CredentialTypeRegistryService.properties
Dynamic UI Manager Factory Service	WP DynamicUIManagerFactoryService	DynamicUIManagerFactoryService.properties
Identification	WP Identification	Identification.properties
Plugin Manager Service	WP PluginManagerService	PluginManagerService.properties
Portal Filter Service	WP PortalFilterService	PortalFilterService.properties
PortletFilterService	WP PortletFilterService	PortletFilterService.properties
Site Analyzer Log Service	WP SiteAnalyzerLogService	SiteAnalyzerLogService.properties
Virtual Portal Identification Service	WP VirtualPortalIdentificationService	VirtualPortalIdentificationService.properties
WSRP Web Service Security	WP WSRPWebServiceSecurity	WSRPWebServiceSecurity.properties
Web Content Service	WP WebContentService	WebContentService.properties
Work Manager Service	WP WorkManagerService	WorkManagerService.properties

Table 33. Web Content Manager services

"Web Content Manager service configuration" on page 353	Service name in the WebSphere Integrated Solutions Console	Properties file
"Web Content Manager configuration service" on page 353	WCM_WCMConfigService	WCMConfigService.properties
"Web Content Manager messaging service" on page 357	WCM_MessagingService	MessagingService.properties
"Web Content Manager pre-rendering service" on page 357	WCM_PrerenderService	PrerenderService.properties

Table 33. Web Content Manager services (continued)

"Web Content Manager service configuration" on page 353	Service name in the WebSphere Integrated Solutions Console	Properties file
"Web Content Manager search service" on page 359	WCM_SearchService	SearchService.properties

There is a number of other Web Content Manager configuration services. The Web Content Manager properties files are located in the *wp_profile_root*/PortalServer/wcm/shared/app/config/wcmservices directory.

Portal service configuration:

IBM WebSphere Portal Express comprises a framework of configuration services to accommodate the different scenarios that portals of today need to address. You can configure some of these services.

The configuration for each service is stored in and accessible for configuration through the WebSphere Integrated Solutions Console. In the WebSphere Integrated Solutions Console, the portal configuration services are spelled as one word, for some services abbreviated, and preceded by the letters WP. Example: In the WebSphere Integrated Solutions Console, the portal Configuration Service is listed as **WP ConfigService**. For more information about how to set properties see the topic about Setting service configuration properties.

Notes:

- The following topics describe the services that can be of interest to the portal administrator. Services that are not described in the following are purely for portal internal usage. Do not modify them in any way.
- The following topics describe the portal services and their configuration properties. In these lists, the values given in parentheses are the default values. Properties given with a value of <none> have no default values.
- You configure the portal configuration services in the WebSphere Integrated Solutions Console. You cannot set the service configuration properties by simply changing the property value in the properties file and restarting the portal.
- For details about how to export a configuration from an existing portal and import it to another portal, refer to the documentation about the portal XML configuration interface.

  "Administrator Unique Names Mapping Service" on page 291
  Administration portlets and themes create URL links to other administration portlets and pages. If these links were hardcoded, they would no longer be usable if you changed the unique names of these pages. Therefore a service for obtaining those unique names is provided in the portal Administrator Unique Names Mapping Service. This service contains properties with key-value pairs that map internal keys to the actual unique names that are assigned to the referenced pages.

  "Cache Manager Service" on page 292
  The portal Cache Manager Service is responsible for managing the different caches used in WebSphere Portal Express.

  "Common Component Configuration Service" on page 295
  You can use the Common Component Configuration service to configure the behavior of the common components framework, the enabler widget container, and the client-side APIs.

  "Configuration Service" on page 297
  The portal Configuration Service is responsible for collecting the most essential configuration data of the IBM WebSphere Portal Express engine.

"Portlet Container Service" on page 346
The portal Portlet Container service provides properties for portlet filtering.

"Project Identification Service" on page 346
The Project Identification Service provides access to the identifier for a currently selected project in IBM Web Content Manager. Projects enable you to make changes to a set of items and publish those changes at the same time.

"Registry Service" on page 346
The portal Registry Service loads and caches a small number of objects that are regularly accessed in the engine. This improves performance. However the trade off is that the cached objects can be stale compared to their database counterparts. This applies particularly in a cluster environment.

"State Manager Service" on page 347
The portal State Manager Service is the access point for managing the navigational state of the portal. The navigational state represents the current view of portal resources as displayed to a user.

"Virtual Portal Configuration Service" on page 352
The Virtual Portal configuration service (WP VirtualPortalConfigService) enables you to specify properties for the default virtual portal and for specific virtual portals.

Administrator Unique Names Mapping Service:

Administration portlets and themes create URL links to other administration portlets and pages. If these links were hardcoded, they would no longer be usable if you changed the unique names of these pages. Therefore a service for obtaining those unique names is provided in the portal Administrator Unique Names Mapping Service. This service contains properties with key-value pairs that map internal keys to the actual unique names that are assigned to the referenced pages.

In the WebSphere Integrated Solutions Console, the portal Administrator Unique Names Mapping Service is listed as **WP AdminUniqueNamesMappingService**.

Note: If you change the unique name of a portal administration page by using the Manage Unique Names portlet, you also need to update that name in the properties. This is required so that the theme and administration portlets still function.
The available mappings are defined as follows:

```
# ------------------------------------------------ #
# Portal administration page unique names mapping #
# ------------------------------------------------ #
# Internal key           unique name               #
# ------------------------------------------------ #
#
#CONTENT_LAYOUT          = ibm.portal.Content
#APPEARANCES             = ibm.portal.Appearance
#MANAGE_PAGES            = ibm.portal.Manage Pages
#UNIQUE_NAMES            = ibm.portal.Custom Unique Names
#ASSIGN_ROLES            = ibm.portal.Resource Permissions
#PROPERTIES_PORTLET      = ibm.portal.Page Properties
#MY_FAVORITES            = wps.My Favorites
#ORGANIZE_FAVORITES      = wps.Organize Favorites
#SET_PERMISSIONS         = ibm.portal.Locks
#MANAGE_LOG              = ibm.portal.Enable Tracing
#MY_PORTAL               = ibm.portal.Home
#ADMINISTRATION          = ibm.portal.Administration
#PAGE_CUSTOMIZER         = ibm.portal.Page Customizer
#PORTLET_MANAGER         = ibm.portal.Web Modules
#MANAGE_MY_PORTLETS      = ibm.portal.Portlets
```

```
#MANAGE_MY_PORTLET_APPS  = ibm.portal.Applications
#MANAGE_WEBSERVICES      = ibm.portal.Web Services
#IMPORTXML               = ibm.portal.Import XML
#SEARCH_CENTER           = ibm.portal.Search Center
#VIRTUAL_PORTAL          = ibm.portal.Virtual Portal
#LOGIN                   = wps.Login
#SELFCARE                = wps.Selfcare
#APP_PROPERTIES          = ibm.portal.Template and Application Properties
#APP_PARAMETER           = ibm.portal.Template Parameters
#APP_ROLES               = ibm.portal.Application Roles
#APP_TEMPLATES           = ibm.portal.Templates
#APP_MEMBERSHIP          = ibm.portal.Application Membership
#APP_CATALOG             = ibm.portal.Catalog
#APP_LAYOUT              = ibm.portal.Template and Application Layout
#PZN_PICKER_PAGE         = ibm.portal.Personalization.Picker
```

Examples of where these unique names are used are: Theme links for New Page, Edit Page, and Assign Permissions.

Cache Manager Service:

The portal Cache Manager Service is responsible for managing the different caches used in WebSphere Portal Express.

In the WebSphere Integrated Solutions Console, the portal Cache Manager Service is listed as **WP CacheManagerService**.

The portal provides two different types of caches: **shared** and **non-shared**:

Shared caches
The shared caches are cluster aware. This means that deleting an element from the cache on one cluster node results in deleting that element from the corresponding cache instances on all other nodes. This ensures that frequently changing data are kept consistent over the whole cluster installation.

Non-shared caches
The non-shared caches are used for data where cluster awareness is of no concern. This avoids unnecessary network communication overhead.

Plan well ahead and apply special care when modifying these properties. There are two levels of properties:

cacheglobal properties
They specify the default setting which is to be used for all caches unless explicitly overridden by the corresponding cache instance property.

cacheinstance.*cacheidentifier* properties
They are used to override a global setting, for example the size of the cache, for a specific instance of a cache.

Changing some or all of these properties can dramatically improve or impair portal performance. Therefore it is recommended not to change the shared setting for any cache unless the consequences are absolutely understood and agreed. If you want to determine the optimal values for the `size`, `lifetime`, `admit-threshold` and `replacement` properties, monitor the cache properties during the staging phase of your portal installation. Use the Tivoli Performance viewer, that is the WebSphere Application Server PMI client (PMI = Performance Monitoring Interface) to find the optimal settings for your environment.

The properties for the Cache Manager Service for both shared and non-shared caches are listed in the following:

cacheglobal.enabled = [true | false]
cacheinstance.*cacheidentifier*.enabled = [true | false]
Use this property to control whether caching is enabled or not. Use this property with care !

cacheglobal.size = *number*
cacheinstance.*cacheidentifier*.size = *number*
Use this property to define the number of elements that can be put into the cache before eviction takes place. The eviction uses a "near LRU" algorithm.

cacheglobal.shared = [true | false]
cacheinstance.*cacheidentifier*.shared = [true | false]
Use this property to define whether a cluster-aware cache is to be used or not.

cacheglobal.lifetime = *number*
cacheinstance.*cacheidentifier*.lifetime = *number*
Use this property to specify the lifetime of elements in the cache in seconds. When the specified lifetime is up, elements are not discarded from the cache immediately. They are evicted when the next element is inserted. Specifying `-1` means an infinite lifetime. In this case no timeout is applied and the cache entry is never evicted.

randomizePercent = *number*
cacheinstance.*cacheidentifier*.randomizePercent= *number*

Use this property to randomize cache entry lifetimes to some extent. If all entries in a cache have the same lifetime, this can result in high loads on the database when reloading entries, as large amounts of entries are evicted at the same time.

Specify the value for this property as a numeric value given in percent. For example, a value of `25` means that all cache entry lifetimes are up to 25% more or less than the default lifetime (given by the lifetime parameter). No cache entry will have a lifetime less than 50% of the default value, no matter how large you specify the value for this property. By default no value is specified. In this case lifetimes are not randomized, and all cache entries have the default lifetime.

If you set the default lifetime property to infinite by the value `-1` , the lifetime randomization setting is not applied, even if you specify a value for the `randomizePercent` property.

You can view the actual randomized lifetime of a cache entry by enabling tracing for class `com.ibm.wps.services.cache.AbstractCache`.

You can set the following additional properties for non-shared caches. (Setting them for shared caches does no harm, they will be ignored.)

cacheglobal.replacement= [aggressive | moderate | conservative]

cacheinstance.*cacheidentifier*.replacement= [aggressive | moderate | conservative]
Controls the eviction algorithm behavior.

cacheglobal.admin-threshold = *number*

cacheinstance.*cacheidentifier*.admin-threshold = *number*
Admittance threshold. Use this to keep unwanted entries from the cache. An

entry is cached only if it is put into the cache as often as specified by the value for this property. If you want each entry to be cached, set this to zero (`0`).

The cache identifiers are described in the following:

com.ibm.wps.pe.deployedresources
Use this cache instance to cache servlet configuration information and the database representation of all web modules stored in the database.

com.ibm.wps.pe.portletregistry
Use this cache instance to cache the database representation of all portlets stored in the database.

com.ibm.wps.pe.portletdefinition
Use this cache instance to cache the database representation of all portlet applications stored in the database.

The following caches are for social rendering and the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework:

com.ibm.workplace.wcm.pzn.plr.BeanListCache
The bean list cache caches the bean list Java objects that the Digital Data Connector plug-ins return. The DDC plug-ins control the cache key generation for the individual entries and whether the bean lists are automatically removed from the cache during user login. By default, this cache is enabled.

Note: Single entries of this cache can have a size of several MB. Therefore, the default number of cache entries for this cache is much lower than the default of other portal caches. When you use the bean list cache, closely monitor the cache and tune it as required. You might also consider the size of individual cache entries and how to influence it. For more information, read *Configuring the maximum number of items loaded from Connections*.

com.ibm.workplace.wcm.pzn.plr.xml.DocumentCache
The document cache is used by the generic XML DDC plug-in for caching the Document Object Model (DOM) objects for individual source URIs. This cache specifically caches the DOMs for associated item attributes. If an individual associated item attribute is flagged as `shared` in the list-rendering profile, the cache entries are shared between different users. Such shared documents do not get invalidated on user login. Documents that are loaded through non-shared associated item attributes are cached separately per user. These cache entries are automatically invalidated during login. By default, this cache is enabled.

com.ibm.workplace.wcm.pzn.plr.json.DocumentCache
The document cache is used by the generic JSON DDC plug-in for caching the Document Object Model (DOM) objects for individual source URIs. This cache specifically caches the DOMs for associated item attributes. If an individual associated item attribute is flagged as `shared` in the list-rendering profile, the cache entries are shared between different users. Such shared documents do not get invalidated on user login. Documents that are loaded through non-shared associated item attributes are cached separately per user. These cache entries are automatically invalidated during login. By default, this cache is enabled.

com.ibm.workplace.wcm.pzn.plr.ListRenderingCache
The list-rendering cache caches the markup that a specific appearance component generates for a specific bean list instance. If you enable this cache, the updates in the appearance component might not become visible

immediately, as updates to the corresponding IBM Web Content Manager design components do not invalidate this cache. In general, the entries in this cache are invalidated together with the corresponding bean list objects in the bean list cache that is listed earlier. As a result, it is good practice to disable this cache on authoring systems and enable it on delivery systems.

To use this cache, you must use the `ListRenderingCache` rendering plug-in to instrument the Web Content Manager design components that are involved in the markup generation for this cache. For more information, read *Using the list-rendering cache*.

For more information about these caches, read *Digital Data Connector cache tuning*.

Related tasks:

"Configuring the maximum number of items loaded from Connections" on page 2123
You can define a value for the maximum number of social objects that you want the IBM Connections to return when data for a list of social objects is requested.

Related reference:

"Digital Data Connector cache tuning" on page 3297
To improve performance, you can tune the caches for the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework.

Common Component Configuration Service:

You can use the Common Component Configuration service to configure the behavior of the common components framework, the enabler widget container, and the client-side APIs.

In the WebSphere Integrated Solutions Console, the portal Common Component Service is listed as **WP CommonComponentService**. It provides the following configuration properties:

cc.multipart.enabled = true|false
Use this property to specify whether multipart requests can be used for batch processing. This property applies to using the enabler client-side APIs. The portal theme uses this property primarily during bootstrap when it loads remote data.

cc.multipart.correlatehosts = true|false
Use this property to specify whether you want hosts in multipart requests to be correlated.

cc.multipart.pageLoadOptimization
This property is optional. You can use it for user interface components to modify page loading for optimization if required. If you do not specify a value for this property, it defaults to `false`.

cc.multipart.pageLoadOptimizationTheme
This property is optional. You can use it for user interface components to modify theme loading for optimization if required. If you do not specify a value for this property, it defaults to `false`.

cc.multipart.pageLoadOptimizationAppWidgets
This property is optional. You can use it for user interface components to modify widget loading for optimization if required. If you do not specify a value for this property, it defaults to `false`.

cc.theme.context
Use this property to specify the context root of the default theme.

cc.theme.id
Use this property to specify the ID of the default common component theme.

cc.enabler.sandboxenabled
Use this property to specify whether the sandbox is enabled. If you have widgets that are loaded in a sandbox, the widgets are loaded inside an iFrame. Loading them inside an iFrame keeps them from interacting with the other JavaScript and DOM resources on the same page.

cc.enabler.subdomains
Use this property to specify an array of strings with the names of the subdomains that are used to create sandboxed widgets. iFrames that load sandboxed iWidgets use subdomains to create an alternative domain rather than the default one so that cross-domain browser security settings are put in place. Use this property to specify the subdomains that you set up that can be resolved correctly through DNS.

cc.enabler.serverdomain
Use this property to specify the name of the server domain that you use with the sandbox.

cc.enabler.useridattribute
Use this property to specify the user registry attribute that uniquely identifies users.

cc.enabler.user.displaynameattribute
Use this property to specify the user registry attribute that is used to display the name of users. The user registry attribute is the attribute on the object that represents a user in the user registry. For example, LDAP or other, that refers to the user's name to display.

cc.enabler.groupidattribute
Use this property to specify the user registry attribute that uniquely identifies groups.

cc.enabler.group.displaynameattribute
Use this property to specify the user registry attribute to is used to display the name of groups.

cc.enabler.acceptedPagesParentNode
Use this property to specify the navigation node that becomes the parent of shared pages that users accept. However, the shared pages parameters were deprecated in WebSphere Portal Express 8.5.

cc.enabler.remote-cache-expiry
Use this property to specify the number of seconds for the remote expiration time for new pages. The amount of time specifies how long cache entries for newly created pages can be cached.

cc.page.autowiredefaultenabled
Use this property to specify whether widgets are autowired automatically.

cc.isDebug = true|false
Use this property to specify whether you want debugging to be enabled or not.

cc.traceConfig
Use this property to specify an array of strings that are used to set client side tracing on user interface components. Use the exact syntax of a JavaScript array, including brackets and double or single quoted values for each item. Example: `["com.ibm.mashups.enabler.*","com.ibm.pb.*"]`

cc.productname = *product name*
Use this property to specify the product name; user interface components such as themes can use the value from this property. Themes or other user interface components can use this property to display a global brand name. For example, in a default portal installation this property is used to specify the product display name WebSphere Portal Express.

cc.theme.autoEditNewPages = true|false
Use this property to specify whether you want blank pages to go into edit mode automatically (`true`) or not (`false`).

cc.theme.alwaysRefreshOnPageSave
Use this property to specify whether you want the page always to refresh when it is saved (`true`) or not (`false`).

The following list gives the navigational state parameters. Use them to set the configuration for the navigational state model. By default the `w` (width), `h` (height), and `st` (state) attributes are persisted to persistence store and are not included in the URL. The page identifier (PID) of the current page and the space identifier (SID) of the current space are added to the URL. In general all navigational state parameters that are not defined in the parameter `navstate.persistence.pstore` are added to the URL. This way all widget navigational state parameters are added to the URL.

cc.enabler.navstate.persistence.url = ['sid','pid']
Use this property to specify an array of strings that specify which attributes of the navigation state are stored in the URL.

cc.enabler.navstate.persistence.pstore = ['w','h','st']
Use this property to specify an array of strings that specify which attributes of the navigation state are stored in the persistence store.

cc.enabler.navstate.persistence.url.limit = 10
Use this property to specify the limit on how many widgets can store their navigation state in the URL.

cc.enabler.navstate.persistence.url.splimit = 10
Use this property to specify the limit on how many shared parameter sets can be stored through encoding in the URL.

Adding new parameters

You can add new parameters by prefixing the key with `dyn:`, for example: `dyn:com.ibm.mashups.`*`someKey`* `=` *`someValue`*. The new name-value pair must be of type string. Restart the server after you add the new parameter.

Configuration Service:

The portal Configuration Service is responsible for collecting the most essential configuration data of the IBM WebSphere Portal Express engine.

In the WebSphere Integrated Solutions Console, the portal Configuration Service is listed as **WP ConfigService**.

Notes:

- Many of the properties that are listed here are set by the installation procedure. Therefore, plan ahead and apply special care when you modify these properties.

- WebSphere Portal Express and the WebSphere Integrated Solutions Console also provide a CP Configuration Service; this service has properties for tagging and rating only.
- The Configuration Service also includes the configuration properties for WSRP services. They are listed and described in the context of the respective WSRP topics for which they are relevant.

was.home = (${WAS_INSTALL_ROOT})
This path is the absolute path to the installation directory of WebSphere Application Server.

wps.home = (${WPS_INSTALL_ROOT})
This path is the home (or install) directory of the WebSphere Portal Express.

redirect.login = (true)
Turns on user-defined redirection after successful login. If a URL is specified under `redirect.login.url` listed later, that URL is used as the URL for the redirection. If no URL is specified, the portal determines the default page for the current user and sends a redirect to that page in the protected portal area.

redirect.login.ssl = (false)
Turns on SSL in the system-defined redirection after successful login. If no URL is specified for the property `redirect.login.url` listed in the following, the redirect URL uses HTTPS for the protocol.

redirect.login.url [optional] = <none>
Specifies the URL for redirection after successful login. If no URL is specified, the portal determines the default page for the current user and sends a redirect to that page in the protected portal area. This setting does not affect implicit logins, such as single sign-on with LTPA token or through an external security manager.

redirect.login.authenticated.url [optional] = <none>
Specifies the URL for redirection after the first access to a protected page when the user is authenticated by an external security manager (TAI) and a portal session does not exist yet. If no URL is specified, the portal either displays the protected page that was originally requested, or, if session resume is enabled, the last page that the user accessed in the previous session.

redirect.logout = (false)
Turns on user-defined redirection after successful logout. If a URL is specified under `redirect.logout.url` listed later, that URL is used as the URL for the redirection. If no URL is specified, the portal determines the default page in the public portal area and sends a redirect to that page.

redirect.logout.ssl = (false)
Turns on SSL in system-defined redirection after successful logout. If no URL is specified, the redirect URL uses HTTPS.

redirect.logout.url = <none>
Specifies the URL for redirection after successful logout. If no URL is specified, the portal determines the default page in the public portal area and sends a redirect to that page.

ldapserviceattributename.attribute [optional] = (*uid*)
Use this property to determine that portal workflow integration uses a dedicated user attribute when individual users on WebSphere Process Server are identified. Set this property to the user attribute that is used by WebSphere Process Server during task authorization. WebSphere Process Server uses the J2EE principal name for this purpose. By default the J2EE principal name maps to the `uid` user attribute in most LDAP servers, except for Domino servers.

Domino LDAP servers use the `cn` attribute by default, therefore for such a configuration set the `ldapserviceattributename.attribute` to the value `cn`. This property is optional.

multiple.realms.enabled = (false)
Multiple Realms Support properties to allow login with uid@realm.

multiple.realms.login.default.realm = <none>
Multiple Realms Support properties to allow login with uid@realm.

multiple.realms.user.dn.template = <none>
Multiple Realms Support properties to allow login with uid@realm.

host.name = <none>
The default is that no value exists for `host.name.`. In this case, portal URLs start with the host name of the incoming request. If you want the host name in URLs be static, you enter a host name here. For example, in a cluster installation you can enter the name of the network traffic dispatcher here. If a host name is entered, this entry is used to create the portal URLs.

host.port.http = <none>
The HTTP port (normally 80).

host.port.https = <none>
The HTTP-SSL port (normally 443).

security.css.protection = (true)
This property determines whether Cross-Site-Scripting security protection is turned on. The default is `true` for enabling the protection.

redirect.commands = (false)
Specifies that a portal command is followed with an HTTP redirect. This way URLs can be bookmarked. Using this feature results in a certain performance. Therefore, it must be used only if needed.

uri.context.path = (/wps)
The context path under which the portal is running.

uri.context.path.facade = (/wsrp)
The context path for the additional WAR file that is used as a facade web application for your WSRP implementation. With this path, you can use Secure Socket Layer (SSL) with Client Authentication for WSRP and simultaneously use other means of authentication for the portal, for example form based authentication. This separation is required as J2EE allows only for one authentication mechanism per WAR file.

uri.home.public = (/portal)
The servlet context of the portal engine for public (or anonymous) pages, that is, pages that users can view without entering a user ID or password.

uri.home.protected = (/myportal)
The servlet context portal engine for protected (or personal) pages, that is, pages that users can view only by entering a user ID and password.

uri.home.doc = (/doc)
The servlet context of the portal engine for the documentation area.

uri.home.substitution = (false)
Determines whether a public URL must be translated to a protected URL if a user session exists.

Important: To preserve the original behavior and design assumptions of WebSphere Portal Express URLs, set the value of the **uri.home.substitution** property to `true`.

Uri.home.substitution controls the behavior of WebSphere Portal Express when a user who is already logged on to the IBM WebSphere Application Server environment on which WebSphere Portal Express is running uses a `/portal` URL to access WebSphere Portal Express.

The original default behavior of WebSphere Portal Express when a logged-in user made a `/portal` request and the **uri.home.substitution** property was either not set or set to `false` was to log out the user and redirect them to the login page. Setting the **uri.home.substitution** property to `true` changes the behavior so that WebSphere Portal Express translates the public URL to a protected URL by redirecting the user to a `/myportal` URL version of the same request without logging out the user. This behavior is how most users want WebSphere Portal Express to function.

A setting in WebSphere Application Server security, called **use available authentication**, affected the behavior of WebSphere Portal Express URLs. The **use available authentication** setting is now set to `true` by default in WebSphere Application Server. When set to `true`, this setting directs WebSphere Application Server to build a security context for requests to unprotected URLs, specifically the `/portal` URL, if possible. More specifically, WebSphere Application Server builds this security context when valid credentials such as an LtpaToken are recognized on the inbound request. In this case, a request to the `/portal` URL by a logged in user does not automatically log out the user and redirect them to the login page. Instead, the request is processed, but in an inconsistent manner. Many things appear to work properly as if the user was recognized as logged in. However, some subtle functional errors might occur, specifically when the rendering of the response embeds secondary requests to the WebSphere Portal Express **contenthandler** function. Therefore, to achieve the most correct operation, set the **uri.home.substitution** property in the **WP ConfigService Resource Environment Provider** to `true`.

Setting the **uri.home.substitution** property to `true` ensures that even when a request to the `/portal` URL is forwarded by WebSphere Application Server with a security context, WebSphere Portal Express still redirects the user to a `/myportal` version of that same URL. This behavior maintains the original design assumption of using two URL entry points into WebSphere Portal Express, one for anonymous access and one for authenticated access.

Note: If you want to preserve the original behavior of WebSphere Portal Express when **uri.home.substitution** is not set or is set to `false`, see the property **logout.user.onpublic** and the following technote, Default triggers for implicit logouts changed in WebSphere Portal v8.

wsrp.resourceproxy.basic.auth.credentialslot = <none>
On a WSRP Consumer portal, you can use this property to specify a credential vault slot that contains the user ID and password credentials. The resource proxy servlet uses the credentials from the credential vault slot when resources that are protected by HTTP basic authentication are fetched. The user ID and password are sent to all remote resources that are referenced in the markup of the remote WSRP portlet.

wsrp.resourceproxy.no.header.forwarding = <none>
On a WSRP Consumer portal, you can use this property to specify the list of HTTP headers that are not forwarded from the client request in addition to the

host header and cookie headers. The host header and cookie headers are never forwarded independent of how this property is set.

Persistent session properties

Use these properties to configure session persistence for users. For more information about persistent session state and its possible options, see the topics about *Configuring user session persistence*.

persistent.session.level = (0)

Determines the level on which the persistent session must operate. If you set this property to a value of 3, this setting does not affect implicit logins, such as single sign-on with LTPA token or through an external security manager.

persistent.session.option = (0)

Determines whether the user gets the option to resume the session. If you set this property to 0, the level setting for the property **persistent.session.level** is applied during login, and the user has no choice whether to resume the previous session or not. If you give users the resume option by setting this property to 1, you must configure the persistent session preservation level by setting the property **persistent.session.level** to 1 or 2.

timeout.resume.session = (false)

Determines whether resuming the session after a session timeout requires user authentication. The default value is false. If this property is set to false and the user tries to continue working after a session timeout, the portal shows an error message that states the session is timed out and the user must log in again. If you set this property to true, the portal ignores the session timeout and does not show the error message. The user can resume the previous session without authentication and continue to work. In both cases, the previous session is resumed according to the setting of the persisted.session.level property.

session.security.use.errorcode = (true)

Use this property to specify whether the portal does a redirect or displays an HTTP error, if session security support is enabled for the portal server and the user in the session does not correspond to the authenticated user in the request. Session security support is a hardening feature of WebSphere Application Server. You can activate it for each application server in the WebSphere Integrated Solutions Console under the **Web Container Settings** > **Session Management** section. If this session security support is active, the application server checks for each authenticated request whether the user who owns the current session matches the user who originated the request. For example, this authentication can be determined by the LTPA token. The portal service configuration property specifies how the portal behaves, if it detects a mismatch between the session user and the authenticated user.

If you set this property to true, the portal returns the HTTP error code that you define by the property session.security.errorcode listed later. This typically results in an appropriate error message to be displayed.

If you set this property to false, you can specify a redirect URL by using the property session.security.redirecturl listed later. For example, you can redirect to a specific error page, which is then displayed to the user.

By default this property is set to true.

For more information about session security support in general, see the appropriate version of the *WebSphere Application Server information center* for your installation.

session.security.errorcode = (409)
Use this property to specify the HTTP error code that is returned if all of the following conditions apply:

1. Session security support is enabled in the WebSphere Application Server.
2. The property `session.security.use.errorcode` listed earlier is set to `true`.
3. A mismatch of the user in the session and the authenticated user is detected.

You must specify a valid HTTP error code. The default is error code 409.

session.security.redirecturl = <none>
Use this property to specify the redirect URL to which portal redirects if all of the following conditions apply:

1. Session security support is enabled in the WebSphere Application Server.
2. The property `session.security.use.errorcode` listed earlier is set to `false`.
3. A mismatch of the user in the session and the authenticated user is detected.

If the property `session.security.use.errorcode` listed earlier is set to `false`, you must specify a value for this property. This property has no default.

portal.session.protection = (true)
Use this property to specify that, for each authenticated portal request, portal checks whether the user in the portal session matches the calling user of the current request. If this portal check results in a mismatch, the portal invalidates the existing session and creates a new one for the calling user to make sure that both identities match. The portal provides this hardening feature, which is independent of the session security support that is provided by WebSphere Application Server. By default this property is set to `true`, therefore the portal checks by default.

portal.enable.filtering = (true)
This flag determines whether the portal must use Portal Filtering or not. The default is `true`.

portlet.url.find = <none>
URL that is used for find and set in global settings portlet.

portlets.unauthorized.visible = (false)
Determines what a user sees whether they are not authorized to view a portlet.

portletcontainer.std.custom.windowStates = <none>
This property defines custom window states that are handled by the portal. This action allows portlets to specify custom window states as defined in the Java Portlet Specification 1.0. The portal allows portlets to generate URLs and so start other portlets with a custom window state if both of the following preconditions apply:

- The started portlet specifies a custom window state in its deployment descriptor (`portlet.xml`).
- That window state is registered by using this property.

The property value is a comma-separated list of custom window states. For example, `portletcontainer.std.custom.windowStates = winState1, myWinState`.

allow.derived.titles = (true)
Determines whether the title and description of derived pages can be redefined by users. If the value is set to `false`, titles and description of pages can be changed only on non-derived pages.

wps.mappingurl.portal_url_identifier = (/!ut/p)
This property determines an identifier for Portal URLs. For the specification of the format of this property, refer to the topic about URL mapping.

Note: With WebSphere Portal Express Version 8.5, URL mappings are deprecated.

wps.mappingurl.enabled = (true)
This property determines whether URL mapping is enabled or not. Possible values are `true` to enable URL mapping, or `false` to disable URL mapping. The default value is `true`.

Note:

- With WebSphere Portal Express Version 8.5, URL mappings are deprecated.
- When you create a URL mapping or create or modify a page, make sure that URL mappings and friendly URLs in your portal do not match, partially overlap, or otherwise interfere with each other. For example, do not use strings such as `home`, `ibm`, `ibm.com`, and do not use strings that are used as URL mappings or friendly URLs in your portal already. Otherwise, several browser redirect loops might occur, sometimes without an error message. To determine such strings, create an export from your portal by using the XML configuration interface and scan the exported XML result output file for the string that you want to use for your URL mapping or for your friendly URL.

wps.mappingurl.invalid = (false)
This property determines how the portal responds to a URL mapping that contains path information. Specify one of the following two values:

true
If you set this property to `true` and the portal gets a request for a URL mapping that contains path information, the portal returns either an HTTP 404 error or redirects the user to the default portal page.

false
This value is the default value. If you set this property to `false` and the portal gets a request for a URL mapping that contains path information, the portal responds as defined by the property `friendly.pathinfo.enabled`.

Notes:

- With WebSphere Portal Express Version 8.5, URL mappings are deprecated.
- The property `friendly.pathinfo.enabled` applies to both friendly URLs and URL mappings.
- The property `state.decoding.fallback` is not applied to URLs that the portal interprets as URL mappings or friendly URLs. If you use friendly URLs or URL mappings, consider setting the parameters `state.decoding.fallback`, `wps.mappingurl.invalid`, and `friendly.pathinfo.invalid` in a consistent way. This action can help provide a consistent user experience. Example: If you set `state.decoding.fallback = false`, consider setting `wps.mappingurl.invalid = true` and `friendly.pathinfo.invalid = true`.

navigation.portletmenu.mode = (0)
The `navigation.portletmenu.mode` property defines in which way portlet

menus are integrated in the overall portal navigation menu structure. Portlet menus are navigation parts that are provided by the portlet itself. They can be added as a subtree to the navigation menu item that references the page in which the portlet is found. This property has the following three options:

0 Disabled: Portlet menus are not displayed in the navigation menu at all. This value is the default value.

1 Current selection: Only the portlet menus of the portlets that are found on the currently selected page are added under the navigation menu item for that page.

2 Everything: The portlet menus of all portlets on all pages are added under the appropriate navigation menu items in the navigation tree.

navigation.expansion.defaultstate = (false)
This value determines whether the nodes in the navigation tree are expanded or collapsed by default. The default is `false`, which means that the nodes are collapsed. Some exceptions apply; for example, the Portal Administration navigation tree is expanded by default.

Note: Setting this value to true does not affect Web 2.0 themes, as the expansion state is not returned from the portal REST service.

page.reload.interval = (0)
This value defines the page reload interval for unauthenticated users. Use it to specify the interval in minutes after which the portal page hierarchy must be reloaded for an unauthenticated user. The reload respects the most current access control settings for that user. If this value is set to zero, no automatic reload occurs during the session.

wsrp.caching.enabled = (true)
Use this property to enable or disable WSRP markup caching. The default for this property is `true`. This value means that WSRP markup caching is enabled, if no value is specified for this property. For more information, see the topic about *WSRP Markup Caching*.

friendly.enabled = (true)
This property determines whether friendly URL names can be set for portal pages in the Manage Pages portlet. The default value is `true`. If you set this property to `true`, you can add friendly URLs for portal pages in the Manage Pages portlet. "Friendly" means that you can use a name that is concise and easy to remember to address a specific portal page. To add a friendly URL for a portal page, click the **Edit Page Properties** icon for the page for which you want to add a friendly URL. You can then give your portal users that URL, and they can access that page by entering the URL in the Address field of their browser.

Note: When you create a URL mapping or create or modify a page, make sure that URL mappings and friendly URLs in your portal do not match, partially overlap, or otherwise interfere with each other. For example, do not use strings such as `home`, `ibm`, `ibm.com`, and do not use strings that are used as URL mappings or friendly URLs in your portal already. Otherwise, several browser redirect loops might occur, sometimes without an error message. To determine such strings, create an export from your portal by using the XML configuration interface and scan the exported XML result output file for the string that you want to use for your URL mapping or for your friendly URL.

If this property is set to `true`, you can use the property `friendly.redirect.enabled` listed later to determine whether a redirect must be sent if the incoming URL did not contain the friendly URL prefix of the addressed page.

`friendly.redirect.enabled = (true)`

Use this property to determine whether a redirect must be sent if the incoming URL did not contain the friendly URL prefix of the addressed page. This property does not take any effect if friendly URLs are disabled by setting the property `friendly.enabled` to `false`. Valid values for this property are as follows:

`true`

Set this property to `true` if you use an External Security Manager in your portal deployment that is configured to protect URLs based on their prefixes. This value is the default value of this property.

`false`

If you set this property to `false`, no redirect is sent in the previous case.

CF03 **`friendly.pathinfo.validation.redirect.onsuccess.enabled = (true)`**

This key specifies whether portal sends required friendly URL redirects if the path information of an incoming friendly URL is valid. Specify one of the following two values:

`true`

This value is the default value. If you set this property to `true` and portal gets a request for a friendly URL that contains path information, portal sends a required friendly URL redirect as if **`friendly.redirect.enabled`** was set to `true`. A required friendly URL redirect is only suppressed if the response indicates that the path information does not identify an available content item to ensure that the configured HTTP status code is sent.

`false`

If you set this property to `false`, portal sends friendly URL redirects as defined by the property **`friendly.redirect.enabled`**.

`friendly.pathinfo.invalid = (false)`

This property determines how the portal responds to a friendly URL that contains path information. Specify one of the following two values:

`true`

If you set this property to `true` and the portal gets a request for a friendly URL that contains path information, the portal returns either an HTTP 404 error or redirects the user to the default portal page. The portal response depends on the setting of the property `state.decoding.fallback`.

`false`

This value is the default value. If you set this property to `false` and the portal gets a request for a friendly URL that contains path information, the portal responds as defined by the property `friendly.pathinfo.enabled`.

Note: The property `state.decoding.fallback` is not applied to URLs that the portal interprets as URL mappings or friendly URLs. If you use friendly URLs or URL mappings, consider setting the parameters `state.decoding.fallback`, `wps.mappingurl.invalid`, and `friendly.pathinfo.invalid` in a consistent way. This action can help provide a consistent user experience. Example: If you set `state.decoding.fallback = false`, consider setting `wps.mappingurl.invalid = true` and `friendly.pathinfo.invalid = true`.

friendly.pathinfo.enabled = (true)

This property determines whether URL mappings and friendly URLs can contain path information to a content item as part of the URL. Specify one of the following two values:

true

This value is the default value. If you set this property to `true` and the portal gets a request for a URL that contains path information, the portal respects that path information and takes the user to the specified portal page.

Note: The property `friendly.pathinfo.enabled` applies to both friendly URLs and URL mappings.
Support for path information in friendly URLs also requires that the property `friendly.enabled` is set to `true` and the property `friendly.pathinfo.invalid` is set to `false`.
Support for path information in URL mappings also requires that the property `wps.mappingurl.enabled` is set to `true` and the property `wps.mappingurl.invalid` is set to `false`.

false

If you set this property to `false` and the portal gets a request for a URL that contains path information, the portal ignores the path information and takes the user only to the requested page.

CF03 **friendly.pathinfo.validation.errorCode = (404)**

This key specifies the HTTP status code that the portal returns if the path information of a friendly URL cannot be resolved to a content item for the requested page. You can specify one of the following values:

404

The default value. This HTTP status code tells a caller, such as a search crawler or web browser, that no content is found for the friendly URL. The missing content might be temporarily or permanently missing.

410

This HTTP status code informs a caller, such as a search crawler or a web browser, that the resource for the friendly URL is no longer available. This missing resource is permanently gone.

Portal can identify conditions that require a different HTTP status code than the one you configure by using **friendly.pathinfo.validation.errorCode**. For example, friendly URL redirects require the HTTP status code 302. To support the most common use cases, see the topic *Preventing friendly URL redirects for invalid friendly URLs for web content*.

CF03 **friendly.pathinfo.validation.errorTextProvider**

This key specifies the text provider of the localized HTTP status message to send as well as the configured HTTP status code. If you configure a text provider and a request URL has invalid path information, portal responds with a blank page that displays only the HTTP status code and the corresponding localized message that is specified by the text provider. The value of this parameter must be the ID of an implementation of the `com.ibm.workplace.wcm.api.plugin.textprovider.TextProvider` interface. To use the default messages of WebSphere Portal Express, specify the text provider with the ID `PathInfoValidationTextProvider`. If you implement a custom text provider, make sure that it supports message keys that are

composed of the prefix HTTP_STATUS_MESSAGE_ and the configured HTTP status code, for example: HTTP_STATUS_MESSAGE_404.

Important: Portal ignores this setting if you also specify the **friendly.pathinfo.validation.errorURI** property or page parameter.

CF03 **friendly.pathinfo.validation.errorResourceBundle**
This key specifies a Java resource bundle as an alternative to implementing a custom text provider. If you configure a Java resource bundle and a request URL has invalid path information, portal responds with a blank page displays only the HTTP status code and the corresponding localized message from the Java resource bundle. The value of this setting must be the fully qualified name of the Java resource bundle. If you provide a custom Java resource bundle, make sure that it contains message keys that are composed of the prefix HTTP_STATUS_MESSAGE_ and the configured HTTP status code, for example: HTTP_STATUS_MESSAGE_404.

Important: Portal ignores this setting if you also specify the **friendly.pathinfo.validation.errorURI** property or page parameter. Portal also ignores this setting if you set the value of the **friendly.pathinfo.validation.errorTextProvider** property or page parameter to a custom text provider ID.

CF03 **friendly.pathinfo.validation.errorURI**
This key specifies the piece of content URI that portal resolves if the request URL has invalid path information. The value of this parameter must be a piece of content URI that portal can resolve, for example:

nm:oid:*unique_page_name*
This navigation model URI redirects the request to a specific portal page based on the unique name of the target page.

custom:resolutionserviceuri
This custom implementation of the com.ibm.portal.resolver.ResolutionService interface resolves invalid path information to a dynamically determined navigational state. When portal resolves the piece of content URI, the content path that failed the portlet validation is passed to the resolution service as the **wcmContentPath** parameter.

CF03 **friendly.pathinfo.validation.errorContentPath**
This key specifies the full content path that portal sets as public Web Content Manager context of the resolved page if the request URL has invalid path information. Web Content Viewer portlets on the resolved page that are configured to listen to other portlets can then render the content with the specified path. The value of this setting must be the path of a content item that is available to users, for example: /Web Content/home/human_resources/health/topic_not_found.

CF03 **friendly.pathinfo.validation.enabled = (false)**
This key specifies whether portal validates the path information of friendly URLs. Specify one of the following two values:

true
If you set this property to true and portal gets a request for a friendly URL that contains path information, portal validates that path information. If it does not identify an available content item, portal responds based on

its configuration and the configuration of the resolved page. For more information, see *Configuring the validation of friendly URLs for web content*.

false
This value is the default value. If you set this property to false and portal gets a request for a friendly URL that contains path information, portal responds based on the properties **friendly.pathinfo.enabled** and **friendly.pathinfo.invalid**. For more information, see *Enabling the validation of friendly URLs for web content*.

friendlyname.uniqueness.enforcement = (true)
This property determines whether the portal enforces that new friendly names are unique across existing non-private sibling nodes. The default value is `true`. The enforcement does not include derived pages with an inherited friendly name and siblings that are moved in by a personalization rule.

com.ibm.wps.resolver.servlet.AbstractServlet.enableWebDAV[optional]=(true)
This property specifies whether the WebDAV feature is enabled in WebSphere Portal Express. By default, this property is set to the value `true`, by which WebDAV is enabled. To disable WebDAV, specify the value `false`. To re-enable WebDAV, specify the value `true`.

portlet.iwidget.markup.prefetching = (true)
This property determines whether the markup of portlets on pages in Client-side rendering mode must be loaded together with the markup for the portal page. The default value is `true`. This property defines the default markup prefetching behavior for pages that are configured to use the Client-side rendering mode. The default behavior can be overridden on a per portlet basis by declaring the same property as a portlet init property in the deployment descriptor file (`portlet.xml`) of the portlet. To disable portlet markup prefetching by default, set the value of this property to `false`. In this case, the markup of portlets on pages in Client-side rendering mode is fetched by using separate HTTP requests.

portlet.enable.transcoding = (true)
Determines whether transcoding is enabled.

portlet.automaximize = (false)
If you set this value to `true`, the portlet window is maximized when a portlet is set into edit, configure, or help mode.

proxy.enable.app.config = (false)
If you set this property to `true`, the Ajax proxy ignores all `proxy-config.xml` files inside portlets.

content.topology.writelock.timeout = *milliseconds* (default=25000)
This setting controls the maximum wait time to obtain a writable model before a timeout warning. To add or change the settings, open **Resource Environment Providers** in the WebSphere Integrated Solutions Console. Restart the portal server after you make your changes.

content.topology.writelock.dump = true|false (default=false)
This setting controls if a Java core memory dump is written in a timeout event for debugging. To add or change the settings, open **Resource Environment Providers** in the WebSphere Integrated Solutions Console. Restart the portal server after you make your changes.

com.ibm.wps.filestore.JCRWebdavTreeModelFactory.cacheClearOnRestart = true|false (default=true)
This setting defines whether the file cache content is invalidated and fetched again after server starts or not. The default value is `true`.

***actual.SSO.tokenUrl* = *your_URL_for_SAP_integration* (no default)**
This property is optional. Use it to specify a referenced property of SAP integration. Change the property name according to your chosen reference in the SAP integration page properties. Specify the URL for SAP integration as the value.

enable.default.social.object.resolution.mode.request.param = (true)|false
This property is optional. The default setting is `true`. If you set this property to `false`, the parameter `ibm.portal.default.social.object.resolution.mode` is disabled. This setting influences how social object links in social lists are resolved. For more information, see *Configuring globally how social object links are resolved*.

content.topology.writelock.dump = true|false (default=false)
This setting controls if a Java core memory dump is written in a timeout event for debugging. To add or change the settings, open **Resource Environment Providers** in the WebSphere Application Server administrative console. Restart the Portal Server after you make your changes.

com.ibm.wps.filestore.JCRWebdavTreeModelFactory.cacheClearOnRestart = true|false (default=true)
This setting defines whether the file cache content is invalidated and fetched again after server starts or not. The default value is `true`.

proxy.cv.slot.regex = *your regular expression with allowed slot IDs*
This property is optional. You can use it to define a subset of available slots in the Credential vault to which you want to limit the access of outbound HTTP connections. For details, read *Authenticating outbound HTTP connections*.

state.decoding.fallback [=true]
Use this property to control how the portal responds to requests for URLs that it cannot decode. Set it to one of the following two values:

true
This value is the default value. If you set this parameter to `true`, the portal renders the default or home page. This action is the fallback solution in scenarios with portal site visitors.

false
If you set this parameter to false, then the portal serves an HTTP 404 error to requests that it cannot decode. This action can be the preferred solution for other scenarios.

Note: The property `state.decoding.fallback` is not applied to URLs that the portal interprets as URL mappings or friendly URLs. If you use friendly URLs or URL mappings, consider setting the parameters `state.decoding.fallback`, `wps.mappingurl.invalid`, and `friendly.pathinfo.invalid` in a consistent way. This action can help provide a consistent user experience. Example: If you set `state.decoding.fallback = false`, consider setting `wps.mappingurl.invalid = true` and `friendly.pathinfo.invalid = true`.

search.service.suppress_automatic_creation = (false)
Use this property to determine whether the automatic creation of search services and search collections is suppressed. Specify one of the following two values:

false
To not suppress the automatic creation of search services and search collections, set this property to `false`. This is the default value.

true
To suppress the automatic creation of search services and search collections, set this property to `true`.

x-method-override.enabled = (false)
Use this property to specify whether you want to have PUT and DELETE requests simulated by tunneling that is by using POST requests instead. To enable this type of tunneling, set this property to `true`. If you set the property `x-method-override.enabled` to `true`, then the Config Service considers the `x-method-override` request header, when a request comes in. Whether to send this header is a decision of the HTTP client. By default, this property is set to `false`, and tunneling is disabled.

wcm.pages.enabled = (true)
This property specifies whether web content pages are enabled. The default value is `true`.

wcm.config.seedlist.version = (1.0)
This property specifies the version of the search seedlist format that is used by the portal. Search seedlist format 1.0 is the only supported search seedlist format, so the default and only supported value is `1.0`.

wcm.config.seedlist.servletpath = (/seedlist)
This property specifies the path to the servlet that generates the search seedlist. The default value is `/seedlist`.

delete.empty.portlet.locales = (false)
This property specifies whether the portal deletes the `localedata` element for a portlet after you set the locale to an empty value.

digest.seed

In WebSphere Portal Express, all resources that are served through the *contenthandler* framework, for example through the entries `/wps/contenthandler` and `/wps/mycontenthandler`, contain a digest token in their URLs. This digest token controls the cacheability of the resources by encoding request dependencies of the resource in the digest. Different dependencies result in a different digest, therefore a different URL, and a different entry in HTTP caches is generated. The digest computation algorithm also takes a seed value into account. This seed is a constant value, identical for all resources on a server. You can control this seed value by setting the *digest.seed* property in the WP ConfigService. You can control the seed value to make sure that all resources served through the *contenthandler* framework get fresh URLs, so eventual cache hits are avoided.

"Portlet Response headers" on page 311
The portlet response headers are part of the portal Configuration Service.

Related concepts:

CF03 "How to enable the validation of friendly URLs for web content" on page 2046
Learn about the properties and values that are required in WebSphere Portal Express Configuration Service Resource Environment Provider to validate friendly URLs for web content.

CF03 “Configuring the validation of friendly URLs for web content” on page 2046
After you enable the validation of friendly URLs for web content, you can choose from various configuration options. These options enable you to specify how the portal responds to friendly URLs that contain path information that does not identify an available content item. Learn about the parameter combinations you can specify and how the portal response varies based on these combinations.

CF03 “How to prevent friendly URL redirects for invalid friendly URLs for web content” on page 2051
If the validation of friendly URLs for web content is enabled and the path information of an incoming friendly URL is not valid, portal responds with the HTTP status code as defined by the portal Configuration Service property and page parameter **friendly.pathinfo.validation.errorCode**. However, depending on the configuration, portal does not always send the configured HTTP status code. Portal can identify conditions that require a different HTTP status code.

Related tasks:

“Configuring globally how social object links are resolved” on page 2127
You can include one or many attributes of social objects in the design component that defines the visual design of your social list. Among other features, social objects have different resolvable links that enable users to open details views of the social objects or the community to which the social objects belong. If you plan to add these links to your social list, you can decide how you want the social objects and their home community to be resolved when users click the corresponding links. IBM WebSphere Portal Express can either resolve the links in the context of the portal itself, or redirect the user to the IBM Connections user interface. You can globally configure how these types of links of social objects are resolved for the users. To do so, you use a setting in the WP Connections Integration Service resource environment provider in the WebSphere Integrated Solutions Console.

“Authenticating outbound HTTP connections” on page 3031
You can protect the access to the remote host by an authentication mechanism.

Portlet Response headers:

The portlet response headers are part of the portal Configuration Service.

portletcontainer.response.headers.additionallyNotAllowed = <none>
There is a predefined set of response header fields that you cannot use in portlet response fields. These unallowable header fields are listed later in this topic. In addition to these predefined header fields you can define additional fields that are then also not allowed to be included into a portlet response header. If you specify more than one field, you must separate the values by commas.

portletcontainer.response.headers.forceAllowed = (none)
This property enables you to re-enable the usage of the header fields that are by default prohibited header fields. If you specify more than one field, you must separate the values by commas.

The following list shows the header fields of the HTTP 1.1 (RFC 2616) specification that are by default **not** allowed to be set:

6.2 Response Header Fields:

The response-header fields allow the server to pass additional information about the response that cannot be placed in the Status-Line. These header

fields give information about the server and about further access to the resource identified by the Request-URI.

- Accept-Ranges (Section 14.5)
- Location (Section 14.30)
- Proxy-Authenticate (Section 14.33)
- Server (Section 14.38)
- Vary (Section 14.44)
- WWW-Authenticate (Section 14.47)

7.1 Entity Header Fields:

Entity-header fields define meta information about the entity-body or, if no body is present, about the resource identified by the request. Some of this meta information is **optional**; some might be **required** by portions of this specification.

- Allow (Section 14.7)
- Content-Encoding (Section 14.11)
- Content-Language (Section 14.12)
- Content-Length (Section 14.13)
- Content-Location (Section 14.14)
- Content-MD5 (Section 14.15)
- Content-Range (Section 14.16)
- Content-Type (Section 14.17)
- Expires (Section 14.21)
- Last-Modified (Section 14.29)

4.2 Message Headers:

HTTP header fields, which include general-header (section 4.5), request-header (section 5.3), response-header (section 6.2), and entity-header (section 7.1) fields, follow the same generic format as that given in Section 3.1 of RFC 822 [9]. Each header field consists of a name followed by a colon (**:**) and the field value. Field names are **case-insensitive**.

CP Configuration Service for tagging and rating:

The CP Configuration Service provides the properties for tagging and rating.

Usage notes for this topic:

1. The CP Configuration Service has properties for tagging and rating only. WebSphere Portal Express also has a portal Configuration Service (without "CP") that provides general portal configuration properties. The WebSphere Integrated Solutions Console lists that service as WP ConfigService.
2. The settings for the properties listed here apply to tagging and rating portal wide. Some of these properties have corresponding parameters that you can apply to individual tag or rating widget instances by the code that calls the widget instance. A setting applied to an individual widget instance overwrites the portal wide setting specified in the CP Configuration Service property listed here. For more details, read the topics about tagging and rating and the parameter reference topics for the widgets for tagging and rating.
3. In the following lists of properties, the values given in parentheses are the default values. Properties marked with **<none>** have no default values.

4. After you modify properties of one of the portal configuration services in the WebSphere Integrated Solutions Console, restart the portal server for your updates to take effect.
5. The parameters listed here are specific to the inline tag and rating widgets that were introduced with IBM WebSphere Portal Express Version 8.5. The dialog and inline widgets of earlier portal versions were deprecated with portal V 8.5. For information about the earlier widgets and their parameters, read the appropriate topics in the WebSphere Portal Express Version 8.0 product documentation.

   "General properties for tagging and rating"
   View the general properties for tagging and rating.

   "Properties for the tag widget" on page 315
   View the properties for the tag widget.

   "Properties for the rating widget" on page 317
   View the properties for the rating widget.

Related reference:

"Parameter reference for the tag and rating widgets" on page 1330
You can configure each of the portal tagging and rating features to determine the look and functionality of these features. To do so, you configure the tag and rating widgets.

Related information:

IBM WebSphere Portal V 8 Product Documentation

General properties for tagging and rating:

View the general properties for tagging and rating.

Note: The parameters listed here are specific to the inline tag and rating widgets that were introduced with IBM WebSphere Portal Express Version 8.5. The dialog and inline widgets of earlier portal versions were deprecated with portal V 8.5. For information about the earlier widgets and their parameters, read the appropriate topics in the WebSphere Portal Express Version 8.0 product documentation.

com.ibm.wps.cp.tagging.isTaggingEnabled = (true)
com.ibm.wps.cp.rating.isRatingEnabled = (true)
Use these properties to enable or disable tagging or rating in the portal user interface. The default value is `true`.

com.ibm.wps.cp.tagging.contextMenu.isTaggingEnabled = (true)
com.ibm.wps.cp.rating.contextMenu.isRatingEnabled = (true)
Use these properties to enable or disable the page menu entries for tagging and rating. The default value is `true`.

com.ibm.wps.cp.tagging.portletContextMenu.isTaggingEnabled = (true)
com.ibm.wps.cp.rating.portletContextMenu.isRatingEnabled = (true)
Use these properties to enable or disable the portlet menu entries for tagging and rating. The default value is `true`.

com.ibm.wps.cp.default.feed.expiration = (600000)
Use this property to configure the default feed expiration time. Specify a value in milliseconds. The default is 600000 milliseconds = 10 minutes.

com.ibm.wps.cp.tagging.tagspace.TimeStampRange = (500)
Use this property to specify the range in which you want the time stamps of a tag space to be considered equal in milliseconds. Setting this property to a low value might affect performance.

com.ibm.wps.cp.tagging.normalization.displayNormalizedNames = (false)
Use this property to define whether the normalized tag names are shown in the tag cloud. Accepted values are `false|true`. The default value is `false`. Example: If there are three tags named `cot?`, `cote`, and `c?te`, the values for this property have the following meaning:

false If this property is set to the default `false`, all three tags show up separately in the tag cloud with a count of 1 each. Search suggestions for `c` are `cot?`, `cote`, and `c?te`; search suggestions for `co` are `cot?` and `cote`; search suggestion for `c?` is only `c?te`.

true If you set this property to `true`, all three tags are normalized to the tag name `cote`. Only this tag name `cote` is shown in the tag cloud or suggested by the search service.

com.ibm.wps.cp.tagging.normalization.typeAhead = (nonnormalized)
Use this property to determine which tag names the type-ahead feature **for tag creation** shows to the user. Accepted values are `nonnormalized|normalized`. The default value is `nonnormalized`. Example: If there are three tags: `cot?`, `cote`, and `c?te`, the values for this property have the following meaning:

nonnormalized
If this property is set to the default `nonnormalized`, the suggestions from the type-ahead feature are as follows:

- If a user types the characters `c?`, the type-ahead service suggests `c?te`.
- If a user types the characters `co`, the type-ahead service suggests `cot?` and `cote`.

normalized
If you set this property to `normalized`, the suggestions from the type-ahead feature are as follows:

- If a user types the characters `c?` or `co`, the type-ahead service shows all three tags as suggestions: `c?te`, `cot?`, and `cote`.

Note: The type-ahead feature works only with the dialog tag widget of the default tagging user interface of portal versions earlier than V 8.5. With WebSphere Portal Express V 8.5, the tag and rating widgets of earlier portal versions are deprecated.

com.ibm.wps.cp.filter.tagging.blacklist = (false)
Use this property to exclude unwanted tags, that is tags that might be seen as offending. If you enable this property, the blacklist filter checks every term that a user enters as a tag, before it is applied and stored. If the term that is used is listed on the blacklist, the portal does not allow this tag to be used and responds with an error message.

com.ibm.wps.cp.filter.tagging.whitelist = (false)
Use this property to check every term that users want to use as a tag before it is applied and stored. If the term that the user types is not listed on the whitelist, the portal does not allow this tag to be used and responds with an error message. This way operators can specify a controlled vocabulary from which users need to chose when they want to tag resources.

com.ibm.wps.cp.tagging.validation.regex = [^\<\>\(\)\[\]:]{1,50}
Use this property to specify a regular expression that the portal uses to validate the tag names that users type. The default is `[^\<\>\(\)\[\`

]:]{1,50}"}. As a result, users can type tag names that are from 1 to 50 characters long and do not contain any angled brackets, parentheses, brackets, or colons: < > ( ) [ ] :.

Limitation note: Angled brackets (< and >) are not allowed within tag names. Therefore, no matter how you customize the regular expression, angled brackets are not accepted.

com.ibm.wps.cp.rating.average.expected.averagerating = (3.0)
Use this property to specify the expected rating average per rated item. An item can be a single resource, or a search scope, such as all resources of type A or category B.

com.ibm.wps.cp.rating.averagel.expected.numberofratings = (10)
Use this property to specify the expected number of ratings per item. This value influences the credibility of the rating instances. The higher this value is, the more the average value is influenced by the value of the property com.ibm.wps.cp.rating.average.expected.averagerating.

com.ibm.wps.cp.rating.maxratingvalue = (5)
Use this property to specify the maximum value that a rating can have. The default is 5.

Related reference:

"Properties for the rating widget" on page 317
View the properties for the rating widget.

"Rating widget parameter reference" on page 1334
You configure rating widget instances by setting the Javascript parameters listed here.

Related information:

IBM WebSphere Portal V 8 Product Documentation

Properties for the tag widget:

View the properties for the tag widget.

Note: The parameters listed here are specific to the inline tag widget that was introduced with IBM WebSphere Portal Express Version 8.5. For information about the earlier widgets and their parameters, read the appropriate topics in the WebSphere Portal Express Version 8.0 product documentation.

com.ibm.wps.cp.tagging.inline.countsEnabled = (false)
Use this property to specify whether the count of each community tag is displayed. The count shows how often the tag was applied by users. The default value is false. To have the count of each community tag that is displayed, specify true. The count is then displayed in parentheses.

com.ibm.wps.cp.tagging.inline.customLabel
Use this property to specify a non-localized custom label to describe the displayed tags. If you do not want any further labeling or if you want to keep the label short, you can specify an empty string. This property has no default value.

com.ibm.wps.cp.tagging.inline.customMessageNoTags
Use this property to specify a non-localized custom label that you want to be displayed if no tags are available. This case can occur when users did not assign tags to the piece of content yet. Non-localized means that the label does not change with the browser language. This property has no default value.

com.ibm.wps.cp.tagging.inline.maxResults = (5)
Use this property to specify the number of tags that are shown per resource. The default value is 5.

com.ibm.wps.cp.tagging.inline.order = (DESC)
Use this property to specify the order direction for displaying the tags. This property is related to the property com.ibm.wps.cp.tagging.dialog.orderMetric listed later. Specify one of the following values:

DESC
This value is the default value. It specifies descending order. For example, when the default TAG_SPACE_COUNT_REVERSE_NAME is specified for the orderMetric property, tags with the highest count and the lowest character in the alphabet are listed first.

ASC
This value specifies ascending order. For example, if the order metric property is specified as orderMetric = TAG_SPACE_COUNT, tags with the lowest count are listed first.

com.ibm.wps.cp.tagging.inline.orderMetric = (TAG_SPACE_COUNT_REVERSE_NAME)
Use this property to specify the order metric for the order by which the tags are displayed. To determine the actual order, use the property com.ibm.wps.cp.tagging.dialog.order . The default value is TAG_SPACE_COUNT_REVERSE_NAME. This default means that tags are shown first by the tag count, with resources with more tags shown before resources with fewer tags, then, if resources have the same number of tags, alphabetically. Specify one of the following values:

- TAG_SPACE_COUNT_REVERSE_NAME. This value is the default value.
- TAG_SPACE_NAME
- TAG_SPACE_COUNT
- TAG_SPACE_CREATION_DATE
- TAG_SPACE_LAST_MODIFIED_DATE
- TAG_SPACE_COUNT_NAME

For more details, read the class com.ibm.portal.cp.Constants.OrderMetric in the portal Javadoc and to the *Inline tag widget parameter reference* in this documentation.

com.ibm.wps.cp.tagging.inline.privateTaggingEnabled = (false)
Use this parameter to enable private tagging. To do so, specify the PERSONAL_PRIVATE scope for the tags that you want to show in this inline widget. The default value is false.

com.ibm.wps.cp.tagging.inline.showDialogLauncher = (true)
Use this property to control whether a plus (**+**) sign for starting the corresponding dialog widget is shown. The default is true.

Note: This property applies only in case of Dojo based inline tagging widgets.

com.ibm.wps.cp.tagging.inline.tagClickActionMode = (TAG_CENTER)
Use this property to determine what happens when a user clicks a tag. Specify one of the following values:

TAG_CENTER
With this value, the widget redirects the user to the tag center. This value is the default value.

PUBLIC_RENDER_PARAMETER
With this value, the widget shows a public render parameter with the tag name.

Related concepts:

"The tag widget" on page 1311
Users can use the tag widget to view, apply, and update tags that were applied to a resource.

Related reference:

"Tag widget parameter reference" on page 1331
You configure specific tag widget instances by setting the JavaScript parameters that are listed here.

Related information:


IBM WebSphere Portal V 8 Product Documentation

Properties for the rating widget:

View the properties for the rating widget.

Note: The parameters listed here are specific to the inline rating widget that was introduced with IBM WebSphere Portal Express Version 8.5. For information about the earlier widgets and their parameters, read the appropriate topics in the WebSphere Portal Express Version 8.0 product documentation.

com.ibm.wps.cp.rating.inline.customLabel
Use this property to specify a non-localized custom label to describe the displayed ratings. This property has no default.

com.ibm.wps.cp.rating.inline.numStars = (5)
Use this property to specify the number of stars or asterisks of which a rating consists. Specify a positive numeric value. The default value is `5`. Do not specify a value larger than the value specified for the property `com.ibm.wps.cp.rating.maxratingvalue` in the WP CP configuration service for tagging and rating. For more information about this property, read *General properties for tagging and rating*.

com.ibm.wps.cp.rating.inline.privateRatingEnabled = (false)
Use this parameter to enable private rating. To enable private rating, set this parameter to `true`. The default value is `false`.

com.ibm.wps.cp.rating.inline.ratingScope = (COMMUNITY_PERSONAL_PUBLIC)
Use this property to specify the scope of ratings that you want to show in this inline widget. Specify one of the following values:

```
COMMUNITY_PERSONAL_PUBLIC|PERSONAL_PUBLIC|PERSONAL_PRIVATE
```

If you do not specify a value, the property defaults to `COMMUNITY_PERSONAL_PUBLIC`.

com.ibm.wps.cp.rating.inline.ratingDesc
Use this property to specify the rating description that you want to show in this rating widget. By default, this property is not part of the rating widget. If you add this property to the rating widget, the default value is `ALL`. Specify one of the following values:

RATING_VALUE
With this value, the rating description includes only the numerical rating value.

TOTAL_NO_RATING
With this value, the rating description includes only the total number of ratings assigned.

ALL With this value, the rating description includes both the numerical rating value and the total number of ratings. This is the default value if you add the `com.ibm.wps.cp.rating.inline.ratingDesc` property to the rating widget.

NONE
If you do not specify a value, the rating description is not displayed at all.

Related concepts:

"The rating widget" on page 1315
Users can use the rating widget to view, apply, and update ratings that were applied to a resource.

Related reference:

"General properties for tagging and rating" on page 313
View the general properties for tagging and rating.

"Rating widget parameter reference" on page 1334
You configure rating widget instances by setting the Javascript parameters listed here.

Related information:

IBM WebSphere Portal V 8 Product Documentation

Content Access Service:

Portlets can access content from remote systems that are located on the other side of a firewall by invoking the portal Content Access Service. If you configure properties of the Content Access Service, these settings applies only to the portlets that call this service.

In the WebSphere Integrated Solutions Console, the portal Content Access Service is listed as **WP ContentAccessService**.

You can configure the properties of the portal Content Access Service at either of the following locations:

- Under the **WP PortletServiceRegistryService** in the WebSphere Integrated Solutions Console.
- In the property file `PortletServiceRegistryService.properties` under the items beginning with `com.ibm.wps.pe.pc.legacy.service.ContentAccessServiceImpl`.

com.ibm.wps.pe.pc.legacy.service.ContentAccessServiceImpl.no.proxy.for =
Specifies host names for which ContentAccessServices does not use a proxy, even if a proxy is configured. Values must be separated by semicolon (`;`). Wildcards are not supported.

Example: `com.ibm.wps.pe.pc.legacy.service...no.proxy.for =localhost;127.0.0.1`

com.ibm.wps.pe.pc.legacy.service.ContentAccessServiceImpl.protocol.handlers =
Assigns additional URL protocol handlers that Java uses to handle connections to various URL protocols. Values must be separated by a vertical bar (`|`). The default is usually sufficient, as it supplies a handler for HTTPs URLs.

Example:
`com.ibm.wps.pe.pc.legacy.service...ServiceImpl.protocol.handlers = com.ibm.net.ssl.internal.www.protocol`

Proxy protocol and port properties

The following properties allow you to specify proxy protocol and port settings for different protocols. You must specify for each protocol the name and port number of the proxy servers that you use. The general format is as follows:

com.ibm.wps.pe.pc.legacy.service.ContentAccessServiceImpl.proxy.http.host = *hostname*
Specifies an HTTP proxy host for HTTP URLs.

com.ibm.wps.pe.pc.legacy.service.ContentAccessServiceImpl.proxy.http.port = *port number*
Specifies the port for the HTTP proxy. If this is not specified, 80 is used as the default value.

com.ibm.wps.pe.pc.legacy.service.ContentAccessServiceImpl.proxy.https.host
Specifies an HTTP proxy host for HTTPs URLs. The proxy must support CONNECT requests, otherwise known as 'tunneling' requests.

com.ibm.wps.pe.pc.legacy.service.ContentAccessServiceImpl.proxy.https.port
Specifies the port for the HTTP proxy. If this is not specified, 80 is used as the default value.

com.ibm.wps.pe.pc.legacy.service.ContentAccessServiceImpl.socks4.host
Specifies a SOCKS V4 proxy host for any URL.

com.ibm.wps.pe.pc.legacy.service.ContentAccessServiceImpl.socks4.port
Specifies the port. If this is not specified, 1080 is used as the default value.

com.ibm.wps.pe.pc.legacy.service.ContentAccessServiceImpl.socks5.host
Specifies a SOCKS V5 proxy host for any URL.

com.ibm.wps.pe.pc.legacy.service.ContentAccessServiceImpl.socks5.port
Specifies the port. If this is not specified, 1080 is used as the default value.

com.ibm.wps.pe.pc.legacy.service.ContentAccessServiceImpl.proxy.auth.enabled
Specifies whether authentication should be tried for proxied connections. This applies to the proxy server, not to the origin server from which the Content Access Service is fetching. Also, this only applies to HTTP proxy (with settings from proxy.http.* and proxy.https.*) and SOCKS proxy (with settings from proxy.socks4.* and proxy.socks5.*).

com.ibm.wps.pe.pc.legacy.service.ContentAccessServiceImpl.proxy.auth.credentialslot
Specifies whether proxy authentication should be used for connections that use a proxy server. You must provide the user ID and password in a credential slot of the portal credential vault. You must also specify the name of this slot in the content access service configuration. The credential must have the type `UserPasswordPassive`. Proxy authentication applies to the proxy server, not to the origin server from which the ContentAccessService is fetching. Also, this only applies to HTTP proxy (with settings from proxy.http.* and proxy.https.*) and SOCKS proxy (with settings from proxy.socks4.* and proxy.socks5.*).

If no proxy host is set, WebSphere Portal Express tries to load all URLs directly. If no port is set, the default port for HTTP (80) is used. Alternatively, you can socksify the TCP/IP stack of your system. Examples:

The name of the HTTP proxy host:

```
com.ibm.wps.pe.pc.legacy.service.ContentAccessServiceImpl.proxy.http.host = host.somewhere.ibm.com
```

The name of the HTTP proxy port:

```
com.ibm.wps.pe.pc.legacy.service.ContentAccessServiceImpl.proxy.http.port = 80
```

The name of the tunneling HTTPs proxy host:

```
com.ibm.wps.pe.pc.legacy.service.ContentAccessServiceImpl.proxy.https.host = securehost.somewhere.ibm.com
```

The name of the HTTPs proxy port:

```
com.ibm.wps.pe.pc.legacy.service.ContentAccessServiceImpl.proxy.https.port = 443
```

Data Store Service:

WebSphere Portal Express uses a database to store configuration data for pages, clients, markup, and all other resources. The Data Store Service is responsible for managing the data source of the portal as configured while installing WebSphere Portal Express.

In the WebSphere Integrated Solutions Console, the portal Data Store Service is listed as **WP DataStoreService**.

Under normal circumstances there should not be a need to modify any of the configuration properties in the Data Store service. The Data Store service properties are listed in the following:

scheduler.cleanup.enabled = (true)
Determines whether deletion of portal pages is performed later by the scheduled cleanup service, or immediately after the user completes the deletion task. This affects the deletion of portal pages and all their dependent resources, such as components and portlet instances. The default is `true`, which means that deletion of portal pages is delayed and performed by the cleanup service.

Note: Even if this property is set to `true` and delayed cleanup, deleted pages are no longer visible to users immediately after deletion.

For details about this property and how to schedule the cleanup service, see the topic about Delayed cleanup of deleted portal pages.

datasource.machineid
The value for this property is equivalent to the MAC address of the server. It consists of a string of 12 hexadecimal figures.

Note: Do not change the value for this property.

The following properties are domain specific properties. They are paired. The last three pairs are analog to the first pair. The possible valid values listed under the first property *xxx*`.datasource.dbms` of the first pair can also be specified for the first property of the following pairs.

Note: Do not assign the same schema name twice for database domains that reside in the same database instance. For example, if the release database domain resides in a database named `DB1` and uses the schema `SCHEMA1`, no other domain in the same database instance can use that same schema name `SCHEMA1`. This restriction applies to domains that are in the same database instance only. Using the same schema name more than once in different database instances of the same database management system is no problem.

rel.datasource.dbms = *your_DBMS*
Use this property to specify the database management system (DBMS) for the release database domain. The default value is . Valid values are listed in the following table. They are also valid for the property *xxx*.datasource.dbms properties in the following three property pairs.

*Table 34. Valid values for xxx.datasource.**dbms** properties*

DBMS used	DBMS value for *xxx*.datasource.dbms properties
IBM DB2 Universal Database for z/OS	DB2_ZOS
IBM DB2 Universal Database Enterprise Server Edition	DB2
IBM DB2 Universal Database for i	DB2_ISERIES
Oracle Enterprise Edition	ORACLE
Microsoft SQL Server Enterprise Edition	SQLSERVER2005

rel.datasource.schema = (*RELEASE*)
Use this property to specify the schema that is used for database objects in the release database domain.

cust.datasource.dbms = *your_DBMS*
Use this property to specify the database management system for the customization database domain. The default value is . For valid values see the property rel.datasource.dbms listed earlier.

cust.datasource.schema = (*CUSTOMIZATION*)
Use this property to specify the schema that is used for database objects in the customization database domain.

comm.datasource.dbms = *your_DBMS*
Use this property to specify the database management system for the community database domain. The default value is . For valid values see the property rel.datasource.dbms listed earlier.

comm.datasource.schema = (*COMMUNITY*)
Use this property to specify the schema that is used for database objects in the community database domain.

jcr.datasource.dbms = *your_DBMS*
Use this property to specify the database management system for the JCR database domain. The default value is . For valid values see the property rel.datasource.dbms listed earlier.

jcr.datasource.schema = (*JCR*)
Use this property to specify the schema that is used for database objects in the JCR database domain.

The following property specifies the **database domain tracking daemon** setting:

domain.tracker.wait = (1000)
Use this property to specify the time for which the domain tracking daemon waits for a response by the database domain until it polls again. The value is specified in milliseconds. The default value is 1000 (milliseconds), which is equivalent to 1 second.

Note: This daemon does not poll continuously, but only in case of errors. Therefore increasing this value will not reduce normal database traffic.

For further information about data sources and their configuration, see the WebSphere Application Server documentation.

Deployment Service:

The portal Deployment Service provides services for accessing the configuration properties that are required for the portlet deployment. The portlet deployment component is responsible for the integration of portlets into the portal. It handles the correct deployment of portlet applications and their WAR files into WebSphere Portal Express and WebSphere Application Server. It uses the WebSphere Application Server management services for the physical deployment and management of WAR files in the WebSphere Application Server. Management of WAR files includes installing, removing, redeploying, starting, and stopping portlet applications.

In the WebSphere Integrated Solutions Console, the portal Deployment Service is listed as **WP DeploymentService**.

Notes:

1. The WebSphere Portal Express configuration is separated into two types:
   - Deployed configuration. This type of configuration is read-only. The deployed configuration is always read from the file `portlet.xml`.
   - Administrative configuration. This type of configuration is read and write.

   The deployed configuration can be modified by administrative changes, for example, by using administrative portlets or the XML configuration interface. The administrative configuration is never overwritten by changes to the deployed configuration.
2. Portlet applications appear in the Enterprise Application list on the WebSphere Integrated Solutions Console. However, do not manage them from outside the portal. Instead, manage them by using the portal administration portlets or the XML configuration interface of the portal. You recognize web applications that comprise a portlet application by their administrative name, also called the display name. It is shown in the WebSphere Integrated Solutions Console. You can identify the name of such a portlet application by a portal-specific identifier prefix `PA_<name>`. This identifier is appended to the name. An example for such a name is `PA_WPS_Welcome`. The name in turn is derived from the name of the WAR file when the portlet application is installed. You can change this administrative name with an update of the portlet application.

In the following list of properties, the values that are given in parentheses are the default values.

was.admin.host = (*localhost*)
The WebSphere Application Server administrative host name. This property is used to adapt to the WebSphere Application Server bootstrap host name, if the default is not applicable.

use.admin.user = (true)
Use this key to select between two user authentication mechanisms for the portal Portlet Deployment Manager to authenticate with the WebSphere Application Server administrative services when portal security is enabled. Specify one of the following two possible values:

true
Use a single preset shared user ID for all portal administrative users who issue WAR deployment requests. This value is the default value. This user

ID is a separate user ID that is common for all users who have the access rights to install or manage applications. You must register this user ID with WebSphere Integrated Solutions Console User Administrator rights.

false
Use the actual user ID by which the administrator issues the WAR deployment request. Every portal user with portlet deployment rights must be added to the WebSphere Integrated Solutions Console User list with administrator rights. Alternatively, you can add the complete group of portal administrators to the WebSphere Integrated Solutions Console Group administrator rights.

was.notification.timeout = (900)
Use this property to specify the timeout value (in seconds). It specifies how many seconds the deployment tasks waits for an application server event during the management of WAR files. This value may have to be increased on large portal installations.

portletapp.starting.weight = (100)
Use this property to specify the value for the starting weight of the portlet applications (war files). To ensure the correct initialization sequence, this value must be higher than the starting weight of the portal itself.

portletapp.shared.library.list
This property defines a list of library references that are added to each deployed WAR file during deployment. You can specify multiple references by separating them with commas (**,**). The library references must already be defined in the application server, and the JAR files must already be deployed at the location that is assigned in the reference definition.

portletapp.reload.enabled = preserve
Use this property to define the value for the reload property of the deployed WAR file. This property can have the following values:

true
Specify this value to enable reloading mode for all WAR files. Use this value only for portlet development and portlet debugging purposes, but not for production environments.

false
Specify this value to disable reloading mode for all WAR files. This value is the default value.

preserve
When you specify this value, the setting from the file `ibm-web-ext.xmi` is applied, if available.

The default setting is `false`.

Note: Do not enable reloading in a production environment. Enable reloading only for portlet development and portlet debugging purposes.

discard.config.interval = (60)
This property defines the minimum time interval for which the configuration service workspace that is used during WAR file deployment is kept. After this time expires, the workspace is discarded when the portal runs the next deployment task. The unit of measure is minutes. Valid values are listed in the following, together with their meaning:

-1 Never discard the workspace.

0 Always discard the workspace immediately after the action that required the workspace was completed.

> 0 (numerical value greater than 0)
Time interval (in minutes) for which a workspace is retained before it is discarded. It is then rebuilt for the next deployment task.

Notes:

- When you set this property, use good judgment. The proper use of this setting must be a compromise between performance and workspace consumption for the following reasons:
  - Discarding the workspace frequently has a negative impact on deployment performance. The larger your portal installation is, the longer it takes to discard and rebuild the workspace to save the configuration changes during WAR file deployment.
  - However, retaining a workspace keeps the `wp_xxx` temporary directories in the WebSphere Application Server `wstemp` directory. Therefore, the temporary space that they occupy in the file system grows every time a WAR file is deployed and every time the portal is restarted.
- The configuration service workspace is not discarded immediately after expiry of the time interval that you set. The cleanup is done the next time that a deployment operation is called. It checks for expired changes and discards the workspace that they occupy. If further deployment operations occurred *after* the last time that the timer interval expired and the workspace was released, the changes in the last allocated workspace remain in the file system even on portal shutdown. Nevertheless, the previous cleanup reduces the volume of occupied disk space to only those temporary files that were processed after the last cleanup interval.

CF06 The following property protects deployment settings that users modified by using the WebSphere Integrated Solutions Console during WAR update through web module administration in the portal.

protect.deployment.attributes = (false)
Use this property to decide whether you want to have existing web module attributes of a deployed portlet WAR file overwritten with future portal updates or not. Portal deployment functions observe the values that are defined for this service during updates of a WAR file. However, a user can later modify the attributes of a deployed WAR by scripting or by using the WebSphere Integrated Solutions Console. For example, the user can modify the sequence by which class loaders are loaded for an individual WAR file. You can use this property to protect such modifications and prevent them from being overwritten by the portal web module deployment function. Valid values are as follows:

false
If you do not want to protect WAR file modifications, specify `false`. This value is the default value.

true
To protect modifications to WAR files, specify `true`.

File location definitions: The following values define file locations. All these settings have default values. Enable or modify them only if the defaults are not appropriate.

delete.temp.files = (true)
This property determines whether temporary files that were created during deployment in the directory *application.repository.dir.name*/temp are deleted or kept. The default is true, which means that the files are deleted. Change the value to false only for debugging purposes so that you can view the content of the temporarily expanded WAR files. When you complete debugging, change the value back to true and delete the directories manually. If you change the value to false, be aware that the hard disk drive space required by the temporary directory grows with each WAR file that you add or update.

shorten.deployment.names = (true)
Use this key to enforce shorter file names during deployment. Some platforms, such as Windows impose a limit to the length of a file path. File paths that are too long can cause deployment to fail if the resulting path is too long.

deployment.names.limit = (21)
This value is the threshold value for portlet application file and display names. Longer names are shortened if required.

The following setting is for debug purposes only. Enable it only when instructed to do so by support personnel.

deployment.debug.log.times = (false)
This setting is for debug purposes only. Enable it only when instructed to do so by support personnel.

HTTP Client Service:

Several components of the portal need to open HTTP or HTTPs connections to other resources. The portal HTTP Client Service provides a central point for configuration properties to these outbound connections. You can set global properties for the SSL configuration and proxy server usage.

In the WebSphere Integrated Solutions Console, the portal HTTP Client Service is listed as **WP HTTPClientService**.

Notes:

1. These properties do not currently replace all individual portlet proxy configuration properties. To set the proxy properties for specific portlets, consult the documentation for each portlet for how to modify their specific properties.
2. Some functional components of the portal can overwrite each of the settings if the component configuration differs from the global value. The following describes the global settings only; if a component allows you to set component specific properties, these are described in the respective component documentation.

global.ssl.configuration = (NodeDefaultSSLConfig)
Use this property to specify the name of an SSL configuration to be used for secure communication as defined in the WebSphere Application Server security configuration.

global.sso.domain = domain name
Use this property to specify the domain that starts with a dot, for example .a.com and denotes the range of hosts to which single-sign on cookies, such as LTPA, are forwarded from a client request. If the property is not set, single sign-on cookies are not forwarded to any remote host.

global.proxy.http.host = host name
Use this property to specify a proxy host for HTTP URLs. If no proxy host is set, the portal tries to load all HTTP URLs directly.

global.proxy.http.port = port number
Use this property to specify the port for the HTTP proxy. If no value is specified, 80 is used as the default value.

global.proxy.https.host = host name
Use this property to specify a proxy host for HTTPs URLs. If no proxy host is set, WebSphere Portal tries to load all HTTPs URLs directly.

global.proxy.https.port = port number
Use this property to specify the proxy port for HTTPs URLs. If no value is specified, 443 is used as the default value.

global.proxy.auth.credentialslot = slot name
Set this property if you want proxy authentication to be used for connections that use a proxy server. You must provide the user ID and password in a credential slot of the portal credential vault. You then specify the name of this credential slot in this property. The credential must have the type `UserPasswordPassive`. Proxy authentication applies to the proxy server only, not to the target server of the outbound connection.

global.proxy.excludehost = host name
Use this property to specify a particular host for which no proxy connection is used, even if a proxy is configured. You can set this property multiple times. Specify one setting for each host that is excluded from proxy connections.

Live Object Service:

You can use the Live Object configuration service to configure the behavior of the live object framework.

isLOFServiceRequired = true | false
Use this property to specify whether the theme loads the live object service code or not. It is up to the theme code to enforce this property. The portal default theme supports this property. The default value is `true` .

true The theme loads the live object framework service code. This is the default value.

false The theme does not load the live object framework service code.

isDynamicLoading = true | false
Use this property to specify whether the live object handlers are loaded dynamically or statically. The default value is `false` . The setting of this property influences the effect of the properties for specific handlers.

false If you set this property to `false` , the handlers are loaded statically. This means that the handler javascript code is loaded as part of the static file of the live object framework service at the time when the service is requested. This is the default. This setting increases the size of the initial download content, but reduces the number of requests if the live object framework is used on the page.

true If you set this property to `true` , the handlers are loaded dynamically if the appropriate microformat is part of the page. This means that handlers such as click-to-action (C2A) and Person Card are loaded by the semantic service, if DOM (document object model) nodes exist in the page that match the appropriate criteria. After you set this property to `true` , changing the properties for

specific handlers has no effect. For example, if you set this property to `true` and the property `isPersonCardHandlerRequired` is set to `false` , the semantic service loads the Person Card handler dynamically if a `vcard` node exists in the page markup. This setting reduces the size of downloaded content on first page access, but it might increase the number of requests required to render the page.

isPersonCardHandlerRequired = true|false
Use this property to specify whether the Person Card handler is included in static content. The default value is `true` .

Note: This property is only considered if the property `isDynamicLoading` is set to the value `false` .

true The Person Card handler is loaded.

false The Person Card handler is not loaded, and the Person Card does not show, even if `vcard` nodes are on the page.

isActionHandlerRequired
Use this property to specify whether the Person Card action handler is included in static content. The default value is `true` .

Note: This property is only considered if the property `isDynamicLoading` is set to the value `false` .

true The Person Card action handler is loaded.

false The Person Card action handler is not be loaded and `vcard` does not show extension menu actions, even if they are on the page.

isC2AHandlerRequired
Use this property to specify whether the click-to-action (C2A) handler is included in static content. The default value is `true` .

Note: This property is only considered if the property `isDynamicLoading` is set to the value `false` .

true The C2A handler is loaded.

false The C2A handler is not loaded and click-to-action does not work, even if C2A nodes are on the page.

Loader Service:

The portal Loader Service is responsible for dynamically loading class files. The service does so by looking up a given class name in different packages. Upon loading the respective class file, an instance of that class is returned.

In the WebSphere Integrated Solutions Console, the portal Loader Service is listed as **WP LoaderService**.

The Loader Service loads class files in four categories: commands, and supporting classes for screen templates, skin templates, and theme templates. To optimize the efficiency, the implementation of the service is free to cache loaded class files or instances and return a cached instance. That means that the implementation of such classes must be thread safe.

In cases where additional or alternative commands are required, the following configuration property can be modified:

command.path
Use this property to specify the package prefix or prefixes in which commands are searched.

Localizer Service:

The portal Localizer Service provides access to the configured default locale and the system default locale. It also provides a list of supported bidirectional languages. Giving the system default locale is necessary because `Locale.getDefault()` is set to the default.

In the WebSphere Integrated Solutions Console, the portal Localizer Service is listed as **WP LocalizerService**.

Although the locale is set during installation time, you can later change the locale by modifying the following properties in the Localizer Service:

locale.default.language
The language of the locale, for example, EN or PT.

locale.default.country
The country or region code of the locale, for example, US or BR.

locale.default.variant
The variant code of the locale.

The default language must be supported by WebSphere Portal Express. If you leave all three properties without a specified value, the system locale is used as the default locale.

All properties are case-insensitive. The ISO standard ISO-639 is used for the language codes of most languages. For Hebrew the old language code `iw` is used. The ISO standard ISO-3166 is used for the country/region codes.

Model WebDAV Service:

The WP Model WebDAV configuration service provides parameters that the portal uses during theme creation. Changing the values for these parameters only affects future theme instances, but leaves existing theme instances unchanged.

theme.resourceroot.default
This parameter specifies the resource root for the creation of theme folders.

theme.contextroot.default
This parameter specifies the context root for the creation of theme folders.

skin.resourceroot.default
This parameter specifies the resource root for the creation of skin folders.

skin.contextroot.default
This parameter specifies the context root for the creation of skin folders.

Navigator Service:

The portal Navigator Service allows you to specify a number of settings; among these are properties for cache scope and cache expiration. Depending on your configuration, you might be able improve your performance by modifying these properties.

In the WebSphere Integrated Solutions Console, the portal Navigator Service is listed as **WP NavigatorService**.

For more detailed information about page caching for improved performance, see the section about tuning the caching of your portal.

The following list gives properties that influence remote caching:

public.session
Use this property to specify whether an anonymous user always has a public session. This may be useful when a portlet requires a session for anonymous users. the default value is `false` . To enable public sessions for pages that anonymous users can view without logging in, set this property to `true`.

The setting of `public.session` influences the remote cache scope for public pages. If `public.session` is set to `true`, then the cache scope is set to non-shared (private). If `public.session` is set to `false`, then the cache scope is set to shared (public).

Note: Setting `public.session` to `true` might reduce performance.

public.expires
Use this property to specify the cache expiration time (in seconds) for caches outside of WebSphere Portal Express and for unauthenticated pages only. These caches must adhere to the HTTP 1.1 specification (RFC 2616). The `public.expires` key specifies the time after which HTTP caches should drop the response. You can further restrict this time by the `remote.cache.expiration` key.

This value is used as a maximum value for the cache expiration time and as a global default value for unauthenticated pages. If you also set the property `remote.cache.expiration` to a value greater than or equal to zero (`0`), the smaller one of the two values is used.

WebSphere Portal Express calculates and aggregates the remote cache information, that is the scope and expiration time, by a number of properties contributed by themes, pages, and portlets besides the properties described here. Therefore WebSphere Portal Express can do any of the following internally while processing a request:

- Reduce the cache lifetime
- Reduce the cache scope, for example, from public (shared) to private (non-shared)
- Switch off the overall cachability of pages.

Therefore this value might not be static for all responses resulting from requests to unauthenticated pages.

The response of WebSphere Portal Express sets the following header fields:

- The `Expires` header with the expiration time added to the system date and time.
- The `Cache-Control : max-age =` header with the expiration time as its parameter.

The default value specified for this property is `60` seconds. If no value is specified, WebSphere Portal Express defaults the value to `60` seconds.

remote.cache.expiration
This property specifies the maximum cache lifetime of a page, both public and private, in seconds. Use this property to specify a global value for the

expiration of pages in remote caches. Setting this value to zero (`0`) switches caching off in remote caches. If the legacy setting is not available, this property is used for authenticated and unauthenticated pages. If the legacy setting is available, then the smaller of the two values is used for unauthenticated pages only. In this case the `remote.cache.expiration` property is used for authenticated pages in general. If theme, composition, and portlets contribute remote cache information, then the global settings also contribute to the information. In this case the lowest of the values of all contributors is used, including the global settings.

The default value for this property is `60` seconds. If no value is specified, WebSphere Portal Express defaults the value to zero (`0` seconds).

`remoteCacheInfo.response.header.vary`
This property specifies the HTTP headers that force a proxy to cache different variants of the same URL. Use this property to specify a comma separated list of HTTP header fields to which WebSphere Portal Express should refer in its vary field of the generated HTTP response. This is required to ensure that proxy caches can invalidate entries in their cache if the specified header fields do not match from request to request. The default for this property is `User-Agent` .

`public.cache-control`
This property specifies the HTTP headers that force a proxy to cache different variants of the same URL. Use this property to specify a comma separated list of HTTP header fields to which WebSphere Portal Express should refer in its vary field of the generated HTTP response. This is required to ensure that proxy caches can invalidate entries in their cache if the specified header fields do not match from request to request. The default for this property is `no-cache` .

`private.cache-control`
This property specifies the value that is set for the `cache-control` HTTP header field when the portal generates a response in request for private pages. This header field controls the behavior of all caching mechanisms along the request-response chain. The default for this property is `no-cache` .

Portal Security Services:

WebSphere Portal Express provides several configuration services for authentication, Portal Access Control, and Portal User Management (PUMA).

"Authentication Service" on page 331
The portal Authentication Service contains the configuration properties for portal authentication. Authentication means that users identify themselves in order to gain access to the system. Usually they do this by a user ID and password.

"Credential Vault Service" on page 332
You can use the portal Credential Vault Service to configure Vault Adapter implementations that are used by the Credential Vault Service to store credential secrets.

"Portal Access Control Services" on page 333
WebSphere Portal Express provides several configuration services for Portal Access Control.

"Puma Store and Validation Services" on page 343
The following topics list and describe the configuration services for Portal User Management (PUMA): these are the Puma Store Service and the Puma Validation Service.

Authentication Service:

The portal Authentication Service contains the configuration properties for portal authentication. Authentication means that users identify themselves in order to gain access to the system. Usually they do this by a user ID and password.

In the WebSphere Integrated Solutions Console, the portal Authentication Service is listed as **WP AuthenticationService**.

authentication.execute.portal.jaas.login = (false)
Use this property to enable or disable the execution of the portal JAAS login:

false
Disables the execution of the portal JAAS login. This is the default. Disable this property only if you have no JAAS Login Modules defined for the portal application login configuration.

true
Enables the execution of the portal JAAS login. You can enable this property if you have JAAS Login Modules defined for the portal application login configuration.

This is related to performance.

authentication.isLoginUrlActive = (true)
Use this property to enable or disable the automatic login URL in Portal.

Note: This is a `java.lang.Boolean` property.

true
Enables the automatic login URL. This is the default.

false
Disables the automatic login URL

Use the following properties to define the custom filters in the various authentication filter chains in the portal. Each of these properties takes a list of the fully qualified class names of the custom filter implementations, separated by colons (**:**) or semicolons (**;**). For concept information about authentication filters, read the topic about *Configuring authentication filters*.

login.explicit.filterchain = <none>
Use this property to specify the custom filters for the filter chain that is triggered for an explicit login by user name and password. The classes listed in this property must implement the interface `com.ibm.portal.auth.ExplicitLoginFilter`.

login.implicit.filterchain = <none>
Use this property to specify the custom filters for the filter chain that is triggered for an implicit login, that is if the user is already authenticated to WebSphere Application Server but has no portal session yet. The classes listed in this property must implement the interface `com.ibm.portal.auth.ImplicitLoginFilter`.

logout.explicit.filterchain = <none>
Use this property to specify the custom filters for the filter chain that is triggered for an explicit logout. The classes listed in this property must implement the interface `com.ibm.portal.auth.ExplicitLogoutFilter`.

logout.implicit.filterchain = <none>
Use this property to specify the custom filters for the filter chain that is

triggered for an implicit logout, that is if the user got a session timeout. The classes listed in this property must implement the `interface com.ibm.portal.auth.ImplicitLogoutFilter`.

sessiontimeout.filterchain = <none>
Use this property to specify the custom filters for the filter chain that is triggered directly after an idle timeout of the session occurred. The classes listed in this property must implement the interface `com.ibm.portal.auth.SessionTimeoutFilter`.

sessionvalidation.filterchain = <none>
Use this property to specify the custom filters for the filter chain that is triggered for every request before the action handling and rendering is processed. The classes listed in this property must implement the interface `com.ibm.portal.auth.SessionValidationFilter`.

filterchain.properties = <none>
Use an arbitrary set of properties according to the previous pattern to specify properties for any of your custom filters. The property value is then available to the specified filter class in the `SecurityFilterConfig` object passed to its `init` method.

Related concepts:

"Configuring authentication filters" on page 265
The portal authentication filters are a set of plug-in points. You can use them to intercept or extend the portal login, logout, session timeout, and request processing by custom code, for example to redirect users to a specific URL.

Credential Vault Service:

You can use the portal Credential Vault Service to configure Vault Adapter implementations that are used by the Credential Vault Service to store credential secrets.

In the WebSphere Integrated Solutions Console, the portal Credential Vault Service is listed as **WP VaultService**.

General Credential Vault Service properties

You can set the following general configuration properties Credential Vault Service:

systemcred.dn
Specifies the Distinguished name (DN) of the vault administrative user. All system credentials are stored under the user's account. This property is set to the portal administrative user by default.

export.userDN
This is the user DN value of the XML Access user that is allowed to import/export secrets via the XML Configuration interface. This is usually the same user DN string as defined in the same configuration file under the property `systemcred.dn`. This user needs authority to use the XML Configuration interface and has to be used during the import/export. Otherwise an import/export of credential secrets is not possible.

export.cipher
The cipher used during export for encryption. This cipher has to be available via Java JCE in the WebSphere Portal Express system. The default value is AES.

export.keyLength
Number of bits used as key length for the cipher. The default value is 128 .

export.enforceSSL
This field controls whether credential import and export must be done via secured HTTP connection or not. If you set this property to `true` , credential import and export must be done via secured HTTP connection. If you set this property to `false` , it is allowed to import and export credentials also via an unsecured HTTP connection. The default value is `true` .

Vault Adapter specific properties

By default, two Vault Adapter implementations are available: `default-release` and `default-customization`. Those Vault Adapters store credential secrets in the portal server data store. For each implementation, define a unique string type, a class name, and a domain. Optionally, you can specify a configuration file, managing resources, and a read only flag.

You can define the following properties for each Vault Adapter Implementation Type. To be able to differentiate the properties for each type, the properties are in the format `vault.`*`type.key`* . Replace *type* by the Vault Adapter Implementation Type, and replace *key* by the key. The following list shows the properties that you can append:

class
Use this property to specify the Vault Adapter Implementation Class Name, but without the `.class` extension. This property is mandatory.

config
Use this property to specify the path of a configuration file that your adapter may need . This property is optional.

domain = (rel)
Use this property to specify the database domain where the segment and slot configuration data is stored. In the special case of the `DefaultVault`, this also specifies where the actual credentials are stored. This property is mandatory. Possible values are all available database domains as specified in the Data Store Service. The default value is `rel` ; this specifies the release domain.

manageresources = (false)
Use this property to specify whether the Vault Adapter should create and delete resources. This property is optional.

Note: If you set this property to `true`, the adapter must have internal support to manage resources. If you omit this property, it will default to `false` .

readonly = (true)
Use this property to specify whether the underlying vault for this adapter should be considered read only. This property is optional.

Note: If you set this property to `true`, the `manageresources` property is ignored. If you omit this property, it will default to `true` .

Portal Access Control Services:

WebSphere Portal Express provides several configuration services for Portal Access Control.

"Access Control Data Management Service" on page 334
The Access Control Data Management Service contains the configuration

properties for Portal Access Control. The domain short names have to correspond to the domain names that are defined for the portal Data Store Service.

"External Access Control Service" on page 336
The portal External Access Control Service is responsible for collecting authorization data from external security managers, such as Computer Associates eTrust SiteMinder or IBM Security Access Manager.

"Auditing Service" on page 340
With the Auditing Service, you can log a set of events in to a separate audit log file. All events are organized in groups. For example, the logging events **User created** and **User deleted** are grouped and must be turned on or off together.

Access Control Data Management Service:

The Access Control Data Management Service contains the configuration properties for Portal Access Control. The domain short names have to correspond to the domain names that are defined for the portal Data Store Service.

In the WebSphere Integrated Solutions Console, the portal Access Control Data Management Service is listed as **WP AccessControlDataManagementService**.

The following set of properties is mandatory for each database domain that contains resources that need to be protected by Portal Access Control:

accessControlDataManagement.domain.*domain_short_name*.adminuser = *full distinguished name of the administrative user for this domain*
Use this property to define the administrative user. As the value specify a full distinguished name that corresponds to a valid entry in the associated user repository. This property is mandatory.

accessControlDataManagement.domain.*domain_short_name*.admingroup = *full distinguished name of the administrative group for this domain*
Use this property to define the administrative group. As the value specify a full distinguished name that corresponds to a valid entry in the associated user repository. This property is mandatory.

accessControlDataManagement.domain.*domain_short_name*.virtualresource = *name of the virtual root resource of this domain*
This property specifies the virtual root resource. The value is the name of a virtual resource that actually exists in the domain and represents the root of the protected resource hierarchy in this domain. This property is meant for internal use only; do not change its value.

The administrative user and group are granted administrator roles on the full hierarchy of protected resources starting from the virtual root resource of the domain defined with the third setting. These roles are granted in addition to the portal roles of the user or group and therefore not displayed in the Access Control portlets. A valid set of values to these properties could for example look like the following:

```
accessControlDataManagement.domain.rel.adminuser=uid=Bob,o=Your Company
accessControlDataManagement.domain.rel.admingroup=cn=Admins,o=Your Company
accessControlDataManagement.domain.rel.virtualresource=PORTAL
accessControlDataManagement.domain.cust.adminuser=uid=Bob,o=Your Company
accessControlDataManagement.domain.cust.admingroup=cn=Admins,o=Your Company
accessControlDataManagement.domain.cust.virtualresource=PORTAL
accessControlDataManagement.domain.comm.adminuser=uid=Bob,o=Your Company
accessControlDataManagement.domain.comm.admingroup=cn=Admins,o=Your Company
accessControlDataManagement.domain.comm.virtualresource=PORTAL
```

```
accessControlDataManagement.domain.jcr.adminuser=uid=Bob,o=Your Company
accessControlDataManagement.domain.jcr.admingroup=cn=Admins,o=Your Company
accessControlDataManagement.domain.jcr.virtualresource=PORTAL
```

The following additional properties of the Access Control Data Management Service are optional:

accessControlDataManagement.enableNestedGroups = (true)
Use this setting to determine whether the group membership of groups is exploited at all by the Portal Access Control component. Supported values are: `true` and `false`. The default is `true`.

accessControlDataManagement.enableTargetResourceGroupInheritance = (false)
Use this setting to determine whether the group membership of groups is exploited by the Portal Access Control component for permission enforcement on users or groups. If you specify `false`, you can only get permissions on user groups via roles on the groups and on users via roles on the direct groups of which the user is a member. Supported values are: `true` and `false`. The default is `false`.

accessControlDataManagement.reorderRoleNames = (false)
Use this setting to determine whether the role name contains the unique name or the title of the resource on which the role was created. Specify `true` when you use an external authorization provider, such as IBM Security Access Manager, as this makes it easier to find the role names. Supported values are: `true` and `false`. The default is `false`.

accessControlDataManagement.externalizeAllRoles = (false)
This property is only applicable for externalization of resources through the user interface. the default value is `false`. If the property is set to `false` and a resource is externalized, then the following things happen:

1. The resource and all descendants of this resource that are not private and not externalized so far are externalized.
2. The roles and role mappings that exist on all resources that were identified in the previous step 1 are written into the external security manager object space.
3. For the root resource that was chosen to be externalized, a role mapping for the Administrator role for the executing user is created in the external security manager object space.

If this property is set to `true`, then in addition to the previous three steps, roles are created in the external security manager object space for all action sets for the root resource that have not already been created in steps 2 and 3.

accessControlDataManagement.createAdminMappingXMLAccess = (true)
This property is only applicable for externalization of resources through the XML Configuration Interface. If the property is set to false and a resource is externalized the following happens:

1. The resource will be externalized.
2. The roles and role mappings on the resource are written into the external security manager object space.

If the property is set to `true`, then in addition to the two previous steps, a role mapping for the Administrator role is created for the executing user in the external security manager object space.

Connecting to the user repository during startup

If you want the portal to wait and retry connecting to the underlying user repository, and if it is not available during portal startup, change the following two properties. This might be necessary in scenarios where the user repository is only available in a certain time frame after the initialization of the portal startup. As the domain administrative users and groups have to be resolved, the portal cannot start without connecting to the user repository. The service startup performs the specified number of attempts to connect to the user repository, each time waiting for the specified time interval before starting the next attempt. If none of the attempts is successful, the service startup quits with an exception.

accessControlDataManagement.ldapFailoverNumberOfAttempts = (1)
Use this property to specify how many times the service startup attempts to connect to the user repository. The default is 1 (once).

accessControlDataManagement.ldapFailoverInterval = (60)
Use this property to specify how long the service startup waits until it retries to connect to the user repository. This value is specified in seconds. The default is 60 seconds.

External Access Control Service:

The portal External Access Control Service is responsible for collecting authorization data from external security managers, such as Computer Associates eTrust SiteMinder or IBM Security Access Manager.

In the WebSphere Integrated Solutions Console, the portal External Access Control Service is listed as **WP ExternalAccessControlService**.

In the portal External Access Control Service, you can modify the configuration properties listed in the following. However, plan well ahead and apply special care when modifying these properties.

General properties of the External Access Control Service

These properties are used for general purposes of the External Access Control Service.

externalaccesscontrol.ready = (false)
This property indicates whether the configuration in this file has been configured to connect to the External Security Manager. The default value is false .

externalaccesscontrol.server = WebSphere_Portal
externalaccesscontrol.application = WPS
externalaccesscontrol.cell = cell
Role name representations are qualified with a context built by these three properties. For example, the **Administrator@External_Access_Control/xxx/xxx** is represented as follows:

Security Access Manager: Protected object space entry

```
/WPSv6/Administrator@External_Access_Control/xxx/xxx/WPS/WebSphere_Portal/cell
```

eTrust SiteMinder:

```
resource/subrealms under Domain: WebSphere Portal v8
/cell/WebSphere_Portal/WPS/Administrator@External_Access_Control/xxx/xxx
```

Access Manager configuration

Use the following properties to configure the connection between WebSphere Portal Express and your Tivoli Access Manager.

externalaccesscontrol.pdroot = (/WPSv6)
After you completed the `AMJRTE` and `SrvSslCfg` configuration tasks, the following directives are required to allow WebSphere Portal Express to use Tivoli Access Manager as an External Security Manager. Provide the root of your Protected Object Space for Portal Server entries.

externalaccesscontrol.pduser = *sec_master*
externalaccesscontrol.pdpw = *passw0rd*
Use these properties to provide an administrative user ID and password with adequate rights in Tivoli to create, delete, modify the objects in the Protected Object Space. You can use the WebSphere Application Server PropFilePasswordEncoder utility to mask the password. Using PropFilePasswordEncoder will remove any comments and uncommented properties. Therefore create a back up copy of this file for future reference. Example for IBM i Linux Windows:

```
AppServer_root/bin/PropFilePasswordEncoder
wp_profile_root/PortalServer/config/properties/ExternalAccessControlService.properties
        externalaccesscontrol.pdpw
```

Example for :

Note: This command should be typed *on one line* in a command line window.

externalaccesscontrol.pdurl=file:///${WAS_INSTALL_ROOT}/java/jre/PdPerm.properties
Use this property to specify the URL location of the Access Manager properties file for AMJRTE. This URL must be in the format `file:///` *directory_path_to_properties_file* . HTTP URLs are not supported.

externalaccesscontrol.createAcl = (true)
This property is optional. Use this property to specify whether Access Control Lists (ACLs) are created in Access Manager for roles that are stored externally. The default is `true`. If this property is set to `false`, the Access Manager administrator will be responsible for all ACL linkages between Security Access Manager and WebSphere Portal Express. Possible values for this property are:

true
A Security Access Manager ACL will be created for **every** WebSphere Portal Express resource. This is the default.

false
No ACLs will be created for portal objects.

externalaccesscontrol.pdactiongroup = ([WPS])
externalaccesscontrol.pdAction = (m)
These properties are optional. Use these properties to specify the action group and the customized actions to map to portal role membership. If these items do not exist, they will be created at startup. The values previously given are the default values.

Computer Associates eTrust SiteMinder policy server information

Use the following properties to configure the connection between WebSphere Portal Express and your Policy Server.

externalaccesscontrol.domainname = WebSphere Portal V 8
Use this property to specify the domain name that is to be created in the eTrust SiteMinder administrative GUI. All realms and sub-realms will be created under this domain. This domain will be created when starting WebSphere Portal Express.

externalaccesscontrol.scheme = (Basic)
Use this property to specify the scheme that is to be to associated with the realms. You must define this scheme in eTrust SiteMinder before starting WebSphere Portal Express. The default value is `Basic`.

externalaccesscontrol.agentname = *wpsagent*
externalaccesscontrol.agentsecret = *passw0rd*
Use these properties to specify the agent name and secret to establish a run time connection with eTrust SiteMinder. The agent should be a web agent with a static shared secret, so that Web Agents later than Version 4.6 of WebAgents should enable the property `supports 4.x agents` on the eTrust SiteMinder web agent. You can use the WebSphere Application Server PropFilePasswordEncoder utility to mask the password.

Note: Using PropFilePasswordEncoder removes all comments and all properties that are commented out. Therefore make sure you create a back up copy of this file for future reference *before* using the PropFilePasswordEncoder utility.

An example of masking the password is:

```
AppServer_root/bin/PropFilePasswordEncoder wp_profile_root/PortalServer/
config/properties/ExternalAccessControlService.properties
externalaccesscontrol.agentsecret
```

Note: Type this command *on one line* in a command line window.

externalaccesscontrol.admin = *siteminder*
externalaccesscontrol.password = *passw0rd*
Use these properties to specify the administrative user ID and password for a user who can create, delete, and modify eTrust SiteMinder objects that are used to represent WebSphere Portal Express roles. This user ID must have sufficient access to domain level objects in eTrust SiteMinder. You can use the WebSphere Application Server PropFilePasswordEncoder utility to mask the password.

Note: Using PropFilePasswordEncoder removes all comments and all properties that are commented out. Therefore make sure you create a back up copy of this file for future reference *before* using the PropFilePasswordEncoder utility.

An example of masking the password is:

```
AppServer_root/bin/PropFilePasswordEncoder wp_profile_root/PortalServer/
properties/ExternalAccessControlService.properties
externalaccesscontrol.password
```

externalaccesscontrol.userdir = (User Directory 1)
Use this property to specify the User Directory that is associated with the domain. You can configure the failover for user directories in the eTrust SiteMinder administrative GUI. The user directory must exist before you start WebSphere Portal Express.

externalaccesscontrol.failOver = (false)
Use this property to specify whether the ESM subsystem should switch to

another Policy Server if it cannot contact the current one. Possible values are `true` and `false`. You can specify this property as either `externalaccesscontrol.failOver` or as `externalaccesscontrol.failover` .

Note: It is important that this value and the number of Policy Server IP addresses that are specified by the `servers` property are carefully coordinated. If you specify multiple Policy Server addresses on the `servers` property, and this property is set to `false`, then the Computer Associate's Agent API will follow round-robin load balancing, by distributing or spraying requests between the configured Policy Servers. This may be appropriate for a TAI which is only doing `read` operations from the Policy Server(s), but not for `write` operations . If you have multiple servers defined in the `externalaccesscontrol.servers` property (following next), set `failOver` to `true` .

`externalaccesscontrol.servers = server1,server2, . . .`
Use this property to specify the IP addresses of all the Policy Servers. Multiple addresses need to separated by commas. An example is: `servers=10.0.0.1,10.0.0.2` .

Note: If you have multiple servers defined in the `externalaccesscontrol.servers` property, set the `failOver` property to `true` .

You can define the following properties for each server. In order to differentiate the properties for each server, specify the keys in the format `Server IP address.key=value` . The defaults are assumed for any keys that you omit. The available keys are as follows:

`accountingPort = (44441)`
The accounting port for the Policy Server. The default is 44441.

`authenticationPort = (44442)`
The authentication port for the Policy Server. The default is 44442.

`authorizationPort = (44443)`
The authorization port for the Policy Server. The default is 44442.

`connectionMax = (10)`
The maximum number of connections which the authorization service may make to this Policy Server. The default is 10.

`connectionMin = (1)`
The initial number of connections which the authorization service will establish with this Policy Server. The default is 1.

`connectionStep = (1)`
The number of connections that are to be allocated if the authorization service runs out of connections to the Policy Server. The default is 1.

`timeout = (20)`
The connection timeout in seconds. The default is 20.

An example for server 10.0.0.1 is as follows:

```
10.0.0.1.accountingPort=44441
10.0.0.1.authenticationPort=44442
10.0.0.1.authorizationPort=44443
10.0.0.1.connectionMax=30
10.0.0.1.connectionMin=10
10.0.0.1.connectionStep=5
10.0.0.1.timeout=60
```

Auditing Service:

With the Auditing Service, you can log a set of events in to a separate audit log file. All events are organized in groups. For example, the logging events **User created** and **User deleted** are grouped and must be turned on or off together.

In the WebSphere Integrated Solutions Console, the portal Auditing Service is listed as **WP AuditService**.

The section about available events lists and describes the events that are available for auditing.

The audit log output is written to the audit log file. No other log messages are written to this file. For an explanation of the contents of the audit log file, refer to the section about the audit log file.

Auditing service configuration

By default, the audit log service is disabled. Therefore, the service is loaded, but does not register any event listeners for audit logging. The auditing service configuration is controlled by the Auditing Service.

audit.service.enable = (false)
This parameter is the global switch. Set this parameter to `true` to turn on the service. Set this parameter to `false` to turn off the service. The default setting is false.

The actual log file access of the service can be configured by using the following property:

audit.logFileName = log/audit_*$create_time*.log
This property defines the location and the name of the audit log file. The placeholder *$create_time* is replaced by a time stamp during file name generation. A second placeholder *$APPSERVER_NAME* is used for a vertical cluster configuration to make the log file name unique. Example:

```
audit.logFileName = log/audit_$APPSERVER_NAME_$CREATE_TIME.log
```

With the auditing service, you can have the transaction ID written to the audit log file. The project ID can also be written to the audit log file. As these IDs can be long and might not be required in every environment, you can disable the inclusion of the IDs.

audit.showTransactionID.enable = (true)
Use this property to disable transaction IDs in the audit log. Change the value to `false`. The default value is `true`.

audit.projects.enable = (true)
Use this property to disable project IDs in the audit log. Change the value to `false`. The default value is `true`.

You determine the events that you want to be logged by enabling the appropriate properties as required. Set the events that you want to enable to the value `true`. The following groups of events are defined:

```
audit.groupEvents.enable
audit.userEvents.enable
audit.portletEvents.enable
audit.roleEvents.enable
audit.roleBlockEvents.enable
```

```
audit.ownerEvents.enable
audit.resourceEvents.enable
audit.externalizationEvents.enable
audit.userInGroupEvents.enable
audit.webModuleEvents.enable
audit.domainAdminDataEvents.enable
audit.designerDeployServiceEvents.enable
audit.impersonationEvents.enable
audit.taggingEvents.enable
audit.ratingEvents.enable
audit.projectPublishEvents.enable
audit.vanityURLEvents.enable
```

The default value for all of these properties is `false`. That means that no events are logged by default, even if you turned on the service by setting the property `audit.service.enable` to `true`.

To enable one or more groups of events, change the default value of the appropriate `audit.`*`eventGroup`*`.enable` property to `true`.

Available events

This list shows the events that you can log with the auditing service. They are listed by the groups in which they are available. If you enable one group, all events in that group are logged.

Table 35. Groups of events for the audit logging service

Audit logging group	Audit logging event	Meaning of the event
`audit.groupEvents`	Group created event	A new user group was created via portal UIs.
`audit.groupEvents`	Group modified event	A user group was modified via portal UIs.
`audit.groupEvents`	Group deleted event	A user group was deleted via portal UIs.
`audit.userEvents`	User created event	A new user was created via portal UIs.
`audit.userEvents`	User modified event	A user was modified via portal UIs.
`audit.userEvents`	User deleted event	A user was deleted via portal UIs.
`audit.portletEvents`	Portlet Application created event	A new web module or portlet application was created via portal UIs.
`audit.portletEvents`	Portlet Application modified event	A web module or portlet application was modified via portal UIs.
`audit.portletEvents`	Portlet Application deleted event	A web module or portlet application was deleted via portal UIs.
`audit.roleEvents`	Role assigned event	A portal role was assigned to a user. The user was given the specified type of access permission on all resources that are affected by this role. For example, it can be EDITOR on Page1.
`audit.roleEvents`	Role unassigned event	A portal role was unassigned from a user. The user no longer has the specified access rights on the resources that are affected by this role. For example, the user is no longer EDITOR on Page1.
`audit.roleBlockEvents`	Role block modified event	The portal role block information of a resource was changed. The event message contains a list of blocked and non-blocked roles on the resource. As roles can either be inherited or propagated, there are two separate lists for inheriting roles and propagating roles. If propagating role blocks was changed, the list for inheriting roles is empty and vice versa.
`audit.ownerEvents`	Resource owner modified event	The owner of a resource was changed.
`audit.resourceEvents`	Resource created event	A new resource was registered. This event is triggered when the resource is registered in Portal Access Control.

Table 35. Groups of events for the audit logging service (continued)

Audit logging group	Audit logging event	Meaning of the event
audit.resourceEvents	Resource modified event	A registered resource was modified. This event is triggered if the resource is modified in Portal Access Control.
audit.resourceEvents	Resource deleted event	A registered resource is no longer registered in Portal Access Control. This event usually happens when a resource is deleted.
audit.externalizationEvents	Resource externalized event	A resource was externalized. This event means that access permissions to this resource are no longer controlled by Portal Access Control, but by an external Access Manager. For example, it might be Security Access Manager.
audit.externalizationEvents	Resource internalized event	A resource was internalized. It is now controlled by Portal Access Control and no longer by an external Access Manager.
audit.userInGroupEvents	User added to group event	A user was added to a group. The user is now a member of this group and therefore inherits access rights from the group.
audit.userInGroupEvents	User who is removed from group event	A user was removed from a group. The user is no longer a member of that group and does no longer have the inherited access rights.
audit.webModuleEvents	Web Module started event	A new web module was started.
audit.webModuleEvents	Web Module stopped event	An installed web module was stopped.
audit.domainAdminDataEvents	Domain administration data initialized event	The administrative data for a domain, such as administrative user, administrative group, and virtual root resource, was initialized during the start of portal. For the lifetime of the current portal process, this user and group have administrative permissions on the domain resource hierarchy, starting from the virtual root resource. For more information, refer to the Access Control Data Management Service. This event is always thrown for each defined domain during the server start. Because the system causes this event, no user is logged.
audit.designerDeployServiceEvents	Component installed event	A portlet application was created by using IBM Lotus Component Designer.
audit.designerDeployServiceEvents	Component modified event	A portlet application that is created by using IBM Lotus Component Designer was modified.
audit.designerDeployServiceEvents	Component uninstalled event	A portlet application that is created by using IBM Lotus Component Designer was deleted.
audit.impersonationEvents	Impersonation started event	A user started impersonation with another user.
audit.impersonationEvents	Impersonation ended event	A user ended impersonation with another user.
audit.impersonationEvents	Impersonation attempted with no permission event	A user tried to impersonate another user but has no permission.
audit.vanityURLEvents	Vanity URL created	A user created a vanity URL.
audit.vanityURLEvents	Vanity URL modified	A user modified a Vanity URL.
audit.vanityURLEvents	A user deleted a Vanity URL	Vanity URL deleted A user deleted a Vanity URL.
audit.tagEvents	Tag created	A user created a tag.
audit.tagEvents	Tag deleted	A user deleted a tag.
audit.ratingEvents	Rating created	A user created a rating
audit.ratingEvents	Rating deleted	A user deleted a rating.
audit.projectPublishEvents	Project created	A user created a project.
audit.projectPublishEvents	Project published	A user published a project.
audit.projectPublishEvents	Project removed	A user removed a project.

Audit log file

The audit log file contains one audit log message per line. All log messages start with a time stamp, followed by the optional transaction ID, the message code, and the event message. Each event message contains the following information:

- The user ID of the user who triggered the audit event
- Additional information about the event itself.

Events for actions that run in a transaction are written to the log file when the transaction is committed. If the transaction is rolled back, no event messages are written to the log file.

Events for actions that do not run in a transaction are written to the log immediately. In such cases, it is not guaranteed that the related action was completed successfully.

Puma Store and Validation Services:

The following topics list and describe the configuration services for Portal User Management (PUMA): these are the Puma Store Service and the Puma Validation Service.

"Puma Store Service"
The portal Puma Store Service contains the configuration properties for the Portal User Management (PUMA).

"Puma Validation Service" on page 344
The portal PUMA Validation Service contains the configuration properties for the validation component of PUMA.

Puma Store Service:

The portal Puma Store Service contains the configuration properties for the Portal User Management (PUMA).

In the WebSphere Integrated Solutions Console, the portal Puma Store Service is listed as **WP PumaStoreService**.

Properties for both the Portal User Management and the PUMA SPI

The following properties configure both the Portal User Management and the PUMA SPI:

store.puma_default.user.fbadefault.filter =
Defines the default search attribute for users. Usually this is the same as the Relative Distinguished Name (RDN) attribute of the LDAP. Depending on your environment, it might be a different attribute. The value for this property should correspond to the value of one of the following properties in `wkplc.properties`, depending on your security configuration:

- **federated.ldap.loginProperties**
- **standalone.ldap.loginProperties**

store.puma_default.group.fbadefault.filter =
Defines the default search attribute for groups. Usually this is the same as the RDN attribute of the LDAP. Depending on your environment, it might be a different attribute. The value for this property should correspond to the value of one of the following properties in `wkplc.properties`, depending on your security configuration:

- **federated.ldap.loginProperties**
- **standalone.ldap.loginProperties**

store.puma_default.user.base.attributes =
Defines the attribute subset that portal loads during direct user lookups, for

example at Login. Attributes that are not defined in this list are loaded by a separate request to the backend user store.

store.puma_default.user.minimum.attributes =
Defines the attribute subset that portal loads during attribute searches for users. Attributes that are not defined in this list are loaded by a separate request to the backend user store.

store.puma_default.group.minimum.attributes =
Defines the attribute subset that portal loads during attribute searches for groups. Attributes that are not defined in this list are loaded by a separate request to the backend user store.

store.puma_default.userManagement.cacheMode = (true)
Defines whether Puma uses a cache or not. The default for this property is `true`.

store.puma_default.logDuplicateKeyExceptions = (true)
Use this property to determine whether the Data Store component writes DuplicateKeyException error messages out to the portal log or not. This does not influence the error handling: With either setting, the exceptions are handled without an error to the portal system.

true
This is the default. If this property is set to `true` or if the property is not set at all, the exception error messages are written to the log. The exceptions are handled without error to the portal.

false
If you set this property to `false`, the error messages are **not** written out to the log. The exceptions are handled without error to the portal.

Properties for the Portal User Management only, but not the PUMA SPI

The following properties configure only the Portal User Management, but not the PUMA SPI:

store.puma_default.puma.commonname = ({0} {1})
The **Registration / Edit My Profile** portlet can generate the common name (CN) of a user automatically. This property defines how the CN is generated. You can define dynamic and static parts. Dynamic parts are added by using {*X*}, where *X* stands as a reference number to the `puma.commonname.X` that defines the attribute that you want to place here. Dynamic parts can only be user attributes that are available and valid. The default is `{0} {1}`.

store.puma_default.puma.commonname.parts =
Defines the number of dynamic parts in the common name.

store.puma_default.puma.commonname.X =
The user attribute for dynamic part *X*. *X* must be between `0` and `puma.commonname.parts -1`. The default is `puma.commonname.0 = givenname` and `puma.commonname.1 = sn`.

Puma Validation Service:

The portal PUMA Validation Service contains the configuration properties for the validation component of PUMA.

In the WebSphere Integrated Solutions Console, the portal Puma Validation Service is listed as **WP ValidationService**.

Properties for user validation

user.YOURATTRIBUTE.min = (1)
Defines the minimum number of characters that is allowed for the defined *YOURATTRIBUTE*. The default is 1.

user.YOURATTRIBUTE.max = (60)
Defines the maximum number of characters that is allowed for the defined *YOURATTRIBUTE*. The default is 60.

user.YOURATTRIBUTE.charset = (ascii)
Defines the character set against which characters are validated. Supported values are ascii and unicode. The default is ascii.

user.YOURATTRIBUTE.extra_chars = (-._)
Defines extra special characters which are not in the supported character set, but should be treated as valid. By default, the dash, period, and underscore are valid: - . _

Note: The YOURATTRIBUTE portion of the property needs to be spelled in uppercase. The following sections show example sets of properties with attributes. They follow the same pattern as the set described in this section. The example properties are set to the default values.

Settings for the attribute user.fbadefault.filter defined in the Puma Store Service

The following example set of properties shows the settings for the attribute user.fbadefault.filter defined in the Puma Store Service:

user.UNIQUEID =
For this property, specify the value of the user.fbadefault.filter attribute that is defined in the Puma Store Service.

user.UNIQUEID.min = 1

user.UNIQUEID.max = 60

user.UNIQUEID.charset = ascii

user.UNIQUEID.extra_chars = -._

Properties for group validation

The following example set shows the settings for the attribute group.fbadefault.filter defined in the Puma Store Service:

group.RDN=
For this property specify the value of the group.fbadefault.filter attribute that is defined in the portal Puma Store Service. For more information see the topic about the portal Puma Store Service.

group.RDN.min = 1

group.RDN.max = 200

group.RDN.extra_chars = -,_

Properties for password validation

Unlike the properties listed earlier, the properties for password validation do not require any uppercase spelling.

password.min_characters = 5

password.max_characters = 60

password.charset = ascii

password.extra_chars = -._

Portlet Container Service:

The portal Portlet Container service provides properties for portlet filtering.

In the WebSphere Integrated Solutions Console, the portal Portlet Container Service is listed as **WP PortletContainerService**.

legacy.portlet.enable.filtering = (true)
This property determines whether or not portlet filtering is used.

Project Identification Service:

The Project Identification Service provides access to the identifier for a currently selected project in IBM Web Content Manager. Projects enable you to make changes to a set of items and publish those changes at the same time.

All changes to items occur either within the scope of a project or outside of a project. Changes made to item in the project result in draft items that do not affect the live content. Changes made outside of a project affect the live content.

A project's scope applies to each request, so that the request either completely executes within the scope of a particular project or completely outside the project's scope. For a given request, you cannot switch between projects during request processing.

Because a request is associated with a thread, the project identifier is also associated with the thread. The project service `com.ibm.portal.services.project.ProjectIdentificationService` returns this thread-specific project identifier. Note that the service does not define how to associate a project identifier with a thread; this is handled during URL generation.

Example:

```
InitialContext ctx = new InitialContext();
ProjectIdentificationService piService = (ProjectIdentificationService)
                                         ctx.lookup(ProjectIdentificationService.JNDI_NAME);
ObjectID projectID = piService.getProjectID();
```

Registry Service:

The portal Registry Service loads and caches a small number of objects that are regularly accessed in the engine. This improves performance. However the trade off is that the cached objects can be stale compared to their database counterparts. This applies particularly in a cluster environment.

If the age of those objects causes a problem, try reducing the refresh rate for the respective entities.

In the WebSphere Integrated Solutions Console, the portal Registry Service is listed as **WP RegistryService**.

The following list describes the Registry Service properties:

default.interval = (1800)
The default interval for refreshing a bucket. The amount is specified in seconds, for example `default.interval = 1800`.

bucket.<bucket-name>.class
The type of class that the bucket with the given name is caching.

bucket.<bucket-name>.reload [optional = true]
This property controls whether or not the bucket with the given name is reloaded in frequent intervals.

bucket.<bucket-name>.interval = (default.interval)
The length of the reload interval for the bucket with the given name. If no value is set, the default.interval setting is used.

bucket.<bucket-name>.sorted [optional = false]
This property controls whether or not the bucket with the given name needs to keep the cached objects in a sorted order. The sorting order is determined by the objects themselves.

The bucket names are described in the following:

theme
The theme bucket is used to cache the database representation of all themes stored in the database.

language
The language bucket is used to cache the database representation of all languages that are stored in the database.

skin
The skin bucket is used to cache the database representation of all skins stored in the database.

language
The language bucket is used to cache the database representation of all languages stored in the database.

client
The client bucket is used to cache the database representation of all clients stored in the database.

markup
The markup bucket is used to cache the database representation of all markups stored in the database.

State Manager Service:

The portal State Manager Service is the access point for managing the navigational state of the portal. The navigational state represents the current view of portal resources as displayed to a user.

In the WebSphere Integrated Solutions Console, the portal State Manager Service is listed as **WP StateManagerService**.

The portal State Manager Service holds the following properties:

preprocessors =
(com.ibm.wps.state.preprocessors.selection.StandardPortalSelectionImpl)
This property specifies a list of one or more preprocessors that are used. It can take multiple values.

Notes:

1. If you want to add your own custom preprocessors in the WebSphere Integrated Solutions Console, you must first enter the default values in the following sequence and then append your custom preprocessors to the end of the list. The reason is as follows:
   - If you specify a value for this parameter, that value overwrites the default value.
   - The default value is mandatory. Therefore, you cannot replace it by a different value.
   - The following preprocessors must be arranged in order, as for requests they are processed in that order.
2. The required syntax is (*classname* (, *classname*) *) 1 .

The default value is as follows:

```
preprocessors = com.ibm.wps.state.preprocessors.urlmapping.URLMappingPreProcessor,
                com.ibm.wps.resolver.friendly.preprocessors.FriendlyPreProcessor,
                com.ibm.wps.resolver.portal.ResolvedPreprocessor,
                com.ibm.wps.state.preprocessors.selection.StandardPortalSelectionImpl,
                com.ibm.wps.state.preprocessors.selection.FragmentSelectionImpl,
                com.ibm.wps.state.preprocessors.selection.ResourceSelectionImpl,
                com.ibm.wps.state.preprocessors.eclipse.ExtensionPreProcessor,
                com.ibm.wps.state.preprocessors.portlet.RequestParameterMerger
```

Of the default values given, the following two selection preprocessors are alternative options. They process the page that the user selected. All other preprocessors are for portal internal use only and must not be changed.

Note: Both of the following selection preprocessors are mutually exclusive. They cannot be used in combination with each other.

com.ibm.wps.state.preprocessors.selection.StandardPortalSelectionImpl
This value implements the standard portal selection behavior, which prefers displaying pages over displaying labels. Therefore, if a user selects a label, the portal displays a page under that label, rather than the label itself with the message that says that there is no content available. (In this case the page that is displayed is the last page that the user selected under this label, or if that page is not available, the first available page under the label.) This value is the default value.

com.ibm.wps.state.preprocessors.selection.SimpleSelectionImpl
This value implements a simple selection strategy; it always displays the element that the user selected, regardless of whether the user selects a label or a page. If the user selects a label, the portal displays that label with the message that there is no content available. You can replace this value for the previously listed default value.

To prevent loss of the language information of users' browser session, you can also add the following preprocessor to the list of preprocessors:

com.ibm.wps.state.preprocessors.locale.CookieSupportedLanguagePreProcessor
This preprocessor creates a backup copy of the locale information that is found in the navigational state of the portal page to a cookie. A users's choice of language is lost when the navigational state is cleared. For example, the language information is lost if users use bookmarks to friendly URLs for navigation or if the navigational state is cleared intentionally. You can use this preprocessor to preserve the user's language choice. The language is then persisted to the next page that the user selects. When then user selects a different language, the portal updates the

information in the cookie accordingly. You can also determine the maximum lifetime of the cookie that holds the language information. To do so, specify the following property in the State Manager Service:

com.ibm.wps.state.preprocessors.locale.CookieSupportedLanguagePreProcessor.cookie = (-1)

Specify an integer value. The value is interpreted as the number of seconds until the cookie is invalidated. A negative value, for example -1, means that the cookie is not deleted until the browser session is finished, for example by closing the web browser window. The default value for this property is -1, by which the cookie is not invalidated until the end of the browser session. Examples:

com.ibm.wps.state.preprocessors.locale .CookieSupportedLanguagePreProcessor.cookie.maxage=30

The cookie is active for 30 seconds after the last request.

com.ibm.wps.state.preprocessors.locale .CookieSupportedLanguagePreProcessor.cookie.maxage=-1

The cookie is active while the browser window remains open.

com.ibm.wps.state.preprocessors.selection.StandardPortalSelectionImpl.selection.fallback

Use this property to specify what happens if a user requests a page that does not exist, for example by selecting a bookmark for a page that was deleted. Specify one of the following values:

true

This value is the default value. With this value, the portal takes the user to the default fallback page, for example the home page.

false

With this value, the portal gives an HTTP 404 error.

Use this property together with the `state.decoding.fallback` property in the portal WP configuration service. Set the values for the two properties in a consistent way.

keymanager.lru.size = (*integer*)

Use this property to specify the history expiration limit of portal pages that users select. The number that you specify defines the minimum number of different pages that are selected by the user after which the portal can discard the render parameters of a page. (The decision whether the render parameters of the page are discarded depends on the expiration policy of the internal cache that stores the render parameters of those pages.) If the user returns to a page after the user selects the specified number of other pages *and* if the render parameters of that page expired, the portal displays that page in its default state.

You can specify by which circumstances the render parameters of a page are stored or discarded:

1 Each time that the user selects a different page, the render parameters of the portlets on the previously selected page can be discarded.

A positive integer

Specify the required number of pages. The render parameters of a page can be discarded after the user selected that number of other pages.

0 Render parameters are always stored in the portal session memory and never discarded.

Do not specify a value less than zero (0). Negative values are considered to be not valid.

"URL normalization for search of portal pages by external search engines"
You can configure the normalization of the URL of your portal. URL normalization is required to enable external search engines to crawl the content of your portal.

URL normalization for search of portal pages by external search engines:

You can configure the normalization of the URL of your portal. URL normalization is required to enable external search engines to crawl the content of your portal.

For this purpose, URL normalization runs the following actions:

- It removes all elements from a portal page URL that are used for portal internal purposes. For example, it removes actions that are coded into the URL and change the portal state.
- It reduces the portal page URL to those elements that are required for a crawler to read the URL and crawl the portal page.

You can use the following properties to configure the normalization of the URL of your portal.

com.ibm.wps.state.outputmediators.OutputMediatorFactory.normalization_xsl_file =

(UrlNormalization_MIN.xsl)
Use this property to specify the XSL style sheet file that defines the transformation that you want to use to normalize the portal URL. This property needs to be set all on one line and concatenated. The default value is `UrlNormalization_MIN.xsl`. The following two files are available to allow for a minimum or maximum transformation:

UrlNormalization_MIN.xsl
This XSL style sheet contains the states for `portlet-mode`, `window-state`, `renderparameters`, `selection`, and `locale` in the normalized URL. This transformation represents the minimum set of states that must be defined in the URL. All other states are removed from the URL. This value is the default.

UrlNormalization_MAX.xsl
This XSL style sheet contains the states for `portlet-mode`, `window-state`, `renderparameters`, `selection`, `solo`, `locale`, and `screen-template`. This maximum transformation represents the set of states that can be defined in a normalized URL for a web crawler. All other states are removed from the URL.

The meaning of the different states that are listed for the minimum and maximum normalization style sheets is as follows:

portlet-mode
Portlet modes allow a portlet to display a different user interface, depending on the task that the user performs with the portlet. A portlet has five modes of display: `view`, `help`, `edit`, `edit_defaults`, `config`.

window-state
Portlet states allow users to change how the portlet window is displayed within the portal. Users can choose from three different states: maximized, minimized, normal.

renderparameters
Parameters set to render a portal page.

selection
Defines the selected portal page.

solo
A portlet can also be displayed in solo state. Solo state hides the portal theme elements, such as a banner, page navigation, or toolbar.

locale
Defines the language in which the page is presented.

screen-template
Defines the screen that is used on the portal page.

theme-template
Defines the theme that is used on the portal page.

You can also set up your own URL normalization. You can implement a URL normalization that is different from the URL normalization that is provided by the two XSL style sheets that come with the portal. To do so, create your own XSL style sheet and set it as the value for the URL normalization parameter:

1. Here is an example for creating your own XSL style sheet:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
     <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>

     <xsl:template match="text()">
     </xsl:template>

     <!-- Traverse through the tree starting at the root element -->
     <xsl:template match="root">
          <xsl:copy>
               <xsl:copy-of select="@*"/>
     <!-- Search for the state node with the attribute type = navigational -->
               <xsl:apply-templates select="state[@type='navigational']"/>
          </xsl:copy>
     </xsl:template>

     <!-- Selection of all states which should stay coded in the URL -->
     <!-- Allowed States: portlet-mode, window-state, renderparameters (param, value, text),
        selection, solo, locale , screen-template -->
     <xsl:template match="state">
          <xsl:copy>
               <xsl:copy-of select="@*"/>
               <xsl:apply-templates select=" . . . "/>
. . .
          </xsl:copy>
     </xsl:template>

. . .
. . .
. . .

</xsl:stylesheet>
```

2. Set the name of the new XSL style sheet as the value for the URL normalization parameter:

```
com.ibm.wps.state.outputmediators.OutputMediatorFactory.normalization_xsl_file =
     UrlNormalization_Your_Own_Style_Sheet_File_Name.xsl
```

Virtual Portal Configuration Service:

The Virtual Portal configuration service (WP VirtualPortalConfigService) enables you to specify properties for the default virtual portal and for specific virtual portals.

Property syntax

There are several ways that you can define properties for virtual portals:

URL context for the virtual portal
Property format:
`context.`*`virtual_portal_context`*`.property.`*`property_name`*`.`*`property_value`*

Example: `context.vp1.property.property1.true`

Host name of the virtual portal
Property format:
`hostname.`*`virtual_portal_hostname`*`.property.`*`property_name`*`.`*`property_value`*

Example: `hostname.vp1.example.com.property.property1.true`

The default keyword
The default keyword is used to identify the default virtual portal. The default virtual portal has no dedicated URL context or host name.

Property format: `default.`*`property_name`*`.`*`property_value`*

Example: `default.property1.true`

The global keyword
The global keyword is used as a fallback property if there no more specific property defined for a virtual portal.

Property format: `global.`*`property_name`*`.`*`property_value`*

Example: `global.property1.true`

Replace the following variables when defining specific properties:

`property_name`
The name of the property that you are defining.

`property_value`
The value of the property that you are defining.

`virtual_portal_context`
The URL context of the target virtual portal.

`virtual_portal_hostname`
The host name of the target virtual portal.

Note: You can determine the URL context and the host name of the target virtual portal with the Manage Virtual Portals administration portlet.

Evaluation order

The portal evaluates properties for virtual portals in the following order and returns the appropriate value:

1. The portal checks for a specific property defined for a virtual portal by the **`context`**, **`hostname`**, or **`default`** keyword. The value of the property is returned.

Important: If you define conflicting values for a virtual portal with different properties, the results can be unpredictable. For example, this issue can occur if you define one value with the **context** keyword and a different value with the **hostname** keyword. To avoid problems, use only one value.

2. If no specific property is defined for a virtual portal, the portal checks for a property defined by the **global** keyword. The value of the property is returned.
3. If no global property is defined, the value `null` is returned.

Web Content Manager service configuration:

Configuration services for IBM Web Content Manager contain settings for the general operation of the web content system, including settings for messaging, pre-rendering, and searching.

Note: To view or change settings in these configuration services, the preferred method is to use the WebSphere Integrated Solutions Console.

"Web Content Manager configuration service"
The Web Content Manager configuration service (WCM WCMConfigService) defines common configuration settings for Web Content Manager.

"Web Content Manager messaging service" on page 357
The Web Content Manager messaging service (WCM MessagingService) defines settings for enabling the Java messaging services for web content.

"Web Content Manager pre-rendering service" on page 357
The Web Content Manager pre-rendering service (WCM PrerenderService) defines settings that affect how web sites are pre-rendered.

"Web Content Manager search service" on page 359
The Web Content Manager search service (WCM SearchService) defines settings that control how web content is crawled and indexed for searching.

"Personalization service" on page 360
The Web Content Manager Personalization service (WCM PersonalizationService) defines settings that control how web content is used by Personalization.

Web Content Manager configuration service:

The Web Content Manager configuration service (WCM WCMConfigService) defines common configuration settings for Web Content Manager.

api.use.dn
This property specifies whether calls to the Web Content Manager API accept or return the common name (cn) or distinguished name (dn) for user lookups. A value of `true` indicates that the distinguished name is used.

Default value: `false`

defaultLibrary
The name of the default library used for rendering and the API. This is used if the URL or context does not contain a library.

Default value: `Web Content`

default.SiteArea
The name of the default site area to use if the URL does not contain a site area.

Default value: `SiteArea`

connect.businesslogic.module
A list of the modules that are run as part of Web Content Manager on the

portal. Some modules are run as part of the content server itself, while others are only accessed through a URL command. The following modules can be defined:

- `web`: Core module for processing requests for web content. This module is required, so do not remove this module from the list.
- `default`: Core module for processing requests for web content. This module is required, so do not remove this module from the list.
- `ajpe`: Core module for processing requests for web content. This module is required, so do not remove this module from the list.
- `custom`: Core module used to enable custom workflow actions. This module is required, so do not remove this module from the list.
- `syndication`: Core module for managing syndication. This must be enabled on both the syndicator and subscriber servers.
- `itemdispatcher`: Core module used by syndication to send the requested item to a subscriber. This must be enabled on a syndicator server.
- `synd`: Core module for syndication. This must be enabled on a syndicator server.
- `subs`: Core module for subscribing to a syndicator. This must be enabled on a subscriber server.
- `mail`: Core module used for sending email from the email workflow action.
- `plutouploadfile`: Core module used by the authoring portlet to transfer files from the user's computer to the web content system.
- `plutodownloadfile`: Core module used by the authoring portlet to transfer files from the web content system to the user's computer.
- `refreshallitems`: Module to touch all items in a specified library. This will force all items to be saved. This module is accessed through a URL command.
- `unlocklibrary`: Module to unlock a specified library. This module is accessed through a URL command.
- `ajpecatselect`: Module used to update the profile information for the user making the request. This module is accessed through a URL command.
- `memberfixer`: Module to identify or change member IDs between environments with different LDAP topologies. This module is accessed through a URL command.
- `workflowenablement`: Module to enable workflow on content types that do not currently have workflow enabled. This module is accessed through a URL command.
- `clearversions`: Module to clear the version history of an item. This module is accessed through a URL command.
- `clearhistory`: Module to clear the history of an item. This module is accessed through a URL command.

Default value: `web, mail, default, ajpe, ajpecatselect, memberfixer, workflowenablement, itemdispatcher, plutouploadfile, plutodownloadfile, synd, subs, syndication, refreshallitems, unlocklibrary, custom, data, clearversions, clearhistory`

`connect.moduleconfig.syndication.inittasks`

Indicates whether automatic syndication is enabled. This property should be set the same on both the syndicator and the subscriber. If set to "false" , automatic syndication is not enabled. If set to "true", automatic syndication is enabled.

Default value: `true`

`cmpnt.htmlEncodeDefault`
Indicates whether HTML encoding occurs for text in components.

Default value: `true`

`active.content.filtering.enable`
Indicates whether active content filtering is enabled or disabled.

Default value: `true`

`nestedGroupLookup.disabled`
The **`accessControlDataManagement.enableNestedGroups`** setting does not apply to Web Content Manager. Web Content Manager continues to request a user's full hierarchical group membership and therefore prevents environments using Web Content Manager from completely disabling nested groups. This setting will control whether Web Content Manager will perform a nested group lookup with a value of true indicating to disable nested group lookups and a value of false indicating to perform nested group lookups.

Default value: `false`

`recentitems.size`
Specifies the maximum number of recent items to store, up to a maximum value of `100`.

Default value: `10`

`versioningStrategy.Default`
Specifies the default versioning behavior. Possible values include `always`, `never`, or `manual`.

Default value: `always`

`versioningStrategy.AuthoringTemplate`
Specifies the versioning behavior used for authoring templates. Possible values include `always`, `never`, or `manual`.

Default value: `always`

`versioningStrategy.Component`
Specifies the versioning behavior used for components. Possible values include `always`, `never`, or `manual`.

Default value: `always`

`versioningStrategy.Content`
Specifies the versioning behavior used for content items. Possible values include `always`, `never`, or `manual`.

Default value: `always`

`versioningStrategy.PresentationTemplate`
Specifies the versioning behavior used for presentation templates. Possible values include `always`, `never`, or `manual`.

Default value: `always`

`versioningStrategy.SiteArea`
Specifies the versioning behavior used for sites. Possible values include `always`, `never`, or `manual`.

Default value: `always`

versioningStrategy.Taxonomy
Specifies the versioning behavior used for taxonomy items. Possible values include `always`, `never`, or `manual`.

Default value: `always`

versioningStrategy.Workflow
Specifies the versioning behavior used for workflow items. Possible values include `always`, `never`, or `manual`.

Default value: `always`

resource.maxUploadSize
Specifies the maximum size in megabytes (MB) of individual files uploaded in file, image, rich text, and HTML components.

Default value: `16`

resourceserver.maxCacheObjectSize
Specifies the maximum size in kilobytes (KB) of resources to be cached by the resource server module.

Default value: `300`

resourceserver.cacheExpiryDate
Specifies the expiry date of resources cached by the resource server module.

Default value: `REL 1M`

user.cache.enable
Indicates whether the cache holds user object in the web content system.

Default value: `false`

admin.delete.error.percent.threshold
The error percent threshold for a given type when deleting a library. This value is specified as an integer from `0` to `100`. If the threshold level is reached for any type, the library deletion task is stopped.

Note: If the threshold is set to `100`, the task ignores the error rate. If the threshold is set to an unsupported value, such as a value less than `0` or greater than `100`, the threshold is set to the default value of `40` percent.

Default value: `40`

deployment.subscriberOnly
Indicates whether this instance of Web Content Manager will only be subscribed to by other servers and will never itself syndicate content to other servers. If this property is set to `true`, all item gatherers are deleted and the item changed task is not added to the scheduler. This improves performance and is recommended for production machines that are subscribe-only servers.

Default value: `false`

deployment.itemChangedTaskDelay
Specifies the number of seconds to use as the syndication interval, with a minimum of `0` seconds and a maximum of `65536` seconds. A value of `0` will prevent syndication from occurring. The shorter the interval, the sooner an update can be sent, but because frequent syndication can affect performance on servers with large amounts of content, a longer interval might be required.

Default value: `30`

wcm.transaction.timeout
Web Content Manager uses a default transaction timeout of 120 seconds even when the value is changed in the server.

Default value: 120

Web Content Manager messaging service:

The Web Content Manager messaging service (WCM MessagingService) defines settings for enabling the Java messaging services for web content.

topic.publishing.enabled
This must be set to true to enable message generation and delivery.

Default value: `false`

items.topic.publishing.enabled
Indicates whether item topics are published. This must be set to true to enable messages for item state changes.

Default value: `true`

syndication.topic.publishing.enabled
Indicates whether syndication topics are published. This must be set to true to enable messages for the status of syndication.

Default value: `true`

prerender.topic.publishing.enabled
Indicates whether pre-render topics are published. This must be set to true to enable messages for the status of pre-rendering.

Default value: `true`

items.topic.name
The JNDI name of the JMS topic for status changes.

Format for item state changes: jms/IWKTopics/Items

Format for syndication state changes: jms/IWKTopics/Syndication

Format for pre-rendering state changes: jms/IWKTopics/PreRender

Default value: None

topic.connection.secure

Set to true to enable secure topic connections. You must also set username and password.

Default value: `false`

topic.connection.secure.username

Username for secure topic connections.

topic.connection.secure.password

A clear text or encoded password for secure topic connections. The password can be encoded using the *PropFilePasswordEncoder* task.

Web Content Manager pre-rendering service:

The Web Content Manager pre-rendering service (WCM PrerenderService) defines settings that affect how web sites are pre-rendered.

prerender.extended.support.enabled
Indicates whether prererendering supports JSP and PZN requests.

Default value: `false`

prerender.authenticator.classname
Indicates the authenticator used to make a connection to the default login URL for the portal, when an external security manager is not installed.

Default value:
`com.aptrix.cacher.authentication.WCMDefaultPrerenderAutthenticator`

prerender.authenticator.credentials.classname
The credentials used by the authenticator specified by the `prerender.authenticator.classname` property.

Default value:
`com.aptrix.cacher.authentication.DefaultPrerenderPropertiesCredentials`

prerender.default.authenticator.credentials.username
Indicates the user name used by the DefaultPrerenderPropertiesCredentials authenticator. If you are using a custom credential provider, this property is not required.

Default value: *portal_admin_id*

prerender.default.authenticator.credentials.password
Indicates the password used by the DefaultPrerenderPropertiesCredentials authenticator. If you are using a custom credential provider, this property is not required. The password can be encoded using the PropFilePasswordEncoder utility provided with WebSphere Application Server.

Default value: *portal_admin_password*

prerender.default.isSecure
Indicates whether the server URLs should be formatted with secure HTTP (`https://...`) or unsecured HTTP (`http://...`).

Default value: `false`

prerender.default.hostName
The host name of the server performing prerendering. The value is typically represented as a WebSphere Application Server variable (for example, `${WCM_HOST}`).

Default value: `${WCM_HOST}`

prerender.default.hostPort
The port number for server performing prerendering. The value is typically represented as a WebSphere Application Server variable (for example, `${WCM_PORT}`).

Default value: `${WCM_PORT}`

prerender.default.portalContext
The context root for the Web Content Manager web application (for example, `/wps/wcm`). The value is typically represented as a WebSphere Application Server variable (for example, `${WCM_WPS_CONTEXT_ROOT}`).

Default value: `${WCM_WPS_CONTEXT_ROOT}`

prerender.default.portal.servlet.authenticatedContext
The authenticated context root for the portal (for example, `/myportal`). The value is typically represented as a WebSphere Application Server variable (for example, `${WCM_WPS_CONTEXT_ROOT}/${WCM_WPS_PERSONALIZED_HOME}`).

Default value: ${WCM_WPS_CONTEXT_ROOT}/${WCM_WPS_PERSONALIZED_HOME}

prerender.default.portal.servlet.unauthenticatedContext
The unauthenticated context root for the portal (for example, /portal). The value is typically represented as a WebSphere Application Server variable (for example, ${WCM_WPS_CONTEXT_ROOT}/${WCM_WPS_DEFAULT_HOME}).

Default value: ${WCM_WPS_CONTEXT_ROOT}/${WCM_WPS_DEFAULT_HOME}

prerender.default.wcm.servlet.authenticatedContext
The default secured path to the Web Content Manager servlet.

Default value: ${WCM_CONTEXT_ROOT}/myconnect

prerender.default.wcm.servlet.unauthenticatedContext
The default unsecured path to the Web Content Manager servlet.

Default value: ${WCM_CONTEXT_ROOT}/connect

Web Content Manager search service:

The Web Content Manager search service (WCM SearchService) defines settings that control how web content is crawled and indexed for searching.

SearchService.DateFormatString
Indicates the date format to use when entering dates in search forms and for displaying search results. Enter a supported Java date format string. If this property is not set, then the default format is MMM dd yyyy HH:mm:ss z.

Default value: None

SearchService.RecrawlInterval
Indicates the interval in hours between crawling of new web content search sources.

Default value: 4

SearchService.BrokenLinksExpirationAge
Indicates the expiration age in days for broken links in new web content search sources.

Default value: 1

SearchService.MetaFields
Specifies additional elements to crawl when searching for metadata. The format for this property is elementName,key1. To specify more than one metadata field maps, use the following format: elementName1,key1;elementName2,key2;elementName3,key3. For example: metaText,meta.

- elementName is the name of element you want to search for metadata. Any valid element with that name in a searchable site area or content item will be crawled.
- key is the "key" that is specified in an element tag used as part of a search element design. In the previous example, the key of meta has been used. To render the content of the metaText element in a search element design, you would use the following tag: <Element context="autoFill" type="content" key="meta"/>.

Notes:

- Only text elements and short text elements can be searched.
- Only site areas that have been configured to be searchable will be crawled.

Default value: None

SearchService.SearchSeed.ExcludeFileAttachments
Indicates whether resource component attachments are included in the search results. If this property is set to `false`, the files stored in file resource elements in content items can also be searched. Files stored in file resource elements in a site area can also be searched as long as a default content item has been selected.

Default value: `false`

SearchService.SearchSeed.excludeExtensions
Set this to a list of file name extensions that you want to exclude from search results. Separate each item in the list with a comma. Any file resource component attachments and file resource element attachments that have these extensions are excluded from search results. For example, `SearchService.SearchSeed.excludeExtensions=avi,mpeg,zip`.

Default value: None

SearchService.DefaultResultPageSize
Specifies the default number of items displayed per page for new web content search components.

Default value: `10`

SearchService.Siapi.IIOP_URL
The IIOP URL created for the default search service.

Default value: None

SearchService.Siapi.EJB
The EJB name for the JNDI associated with the default search service.

Default value: None

SearchService.Siapi.SOAP_URL
The SOAP URL created for the default search service.

Default value: None

SearchService.Siapi.PSE.Type
The type of search service used for the default search service.

Default value: None

SearchService.DefaultCollectionName
The default web content search collection created during installation.

Default value: `WebContentCollection`

SearchService.DefaultSeedPageSize
Specifies the number of items displayed per page for the Web Content Manager search seed servlet.

Default value: `200`

Personalization service:

The Web Content Manager Personalization service (WCM PersonalizationService) defines settings that control how web content is used by Personalization.

rulesEngine.user.nestedGroupLookup
The accessControlDataManagement.enableNestedGroups setting does not apply to Personalization. Personalization continues to request a users full hierarchical group membership and therefore prevents environments using

Personalization from completely disabling nested groups. This setting will control whether Personalization performs a nested group lookup with a value of true indicating to perform nested group lookups and a value of false indicating to disable nested group lookups.

Default value: `true`

Changing ports

You can change the IBM WebSphere Portal Express ports values after installation if there are port conflicts with other cells on the system.

Before you begin

WebSphere Portal Express is installed.

Note: The starting port parameter is required for a successful completion of the **`modify-ports-by-startport`** task. When you specify a start port, this port becomes the base for assigning port values. The code increments this value as each port is assigned, which means that the WebSphere Portal Express ports are assigned incrementally starting with the port defined with the **`modify-ports-by-startport`** task.

Procedure

1. Run the following task from the *wp_profile_root*`/ConfigEngine` directory to generate the `WebSphere_Portal_PortMatrix.txt` file:

   Note: The port file is in the *wp_profile_root*`/ConfigEngine/log` directory. It lists the WebSphere Portal Express (WebSphere_Portal) ports for your installation.

   - Linux : `./ConfigEngine.sh list-server-ports-by-name -DServerName=WebSphere_Portal -DWasPassword=`*`password`*
   - Windows: `ConfigEngine.bat list-server-ports-by-name -DServerName=WebSphere_Portal -DWasPassword=`*`password`*
   - IBM i: `ConfigEngine.sh list-server-ports-by-name -DServerName=WebSphere_Portal -DWasPassword=`*`password`*
2. Stop the WebSphere_Portal server.
3. Run the following command to change the starting port number:
   - Linux : `./ConfigEngine.sh modify-ports-by-startport -DWasPassword=`*`password`* `-DModifyPortsServer=`*`WebSphere_Portal`* `-DStartPort=`*`starting port number`*
   - Windows: `ConfigEngine.bat modify-ports-by-startport -DWasPassword=`*`password`* `-DModifyPortsServer=`*`WebSphere_Portal`* `-DStartPort=`*`starting port number`*
   - IBM i: `ConfigEngine.sh modify-ports-by-startport -DWasPassword=`*`password`* `-DModifyPortsServer=`*`WebSphere_Portal`* `-DStartPort=`*`starting port number`*
4. Run the following command to change ports by using the port file:

   Note: Sample port files are available on the Setup disc. The following information is an example of a port file although the port values are different based on your environment:

   ```
   BOOTSTRAP_ADDRESS=10035
   SOAP_CONNECTOR_ADDRESS=10025
   ORB_LISTENER_ADDRESS=10034
   SAS_SSL_SERVERAUTH_LISTENER_ADDRESS=10041
   CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS=10036
   ```

```
CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS=10033
WC_adminhost=10042
WC_defaulthost=10039
DCS_UNICAST_ADDRESS=10030
WC_adminhost_secure=10032
WC_defaulthost_secure=10029
SIP_DEFAULTHOST=10027
SIP_DEFAULTHOST_SECURE=10026
IPC_CONNECTOR_ADDRESS=10037
SIB_ENDPOINT_ADDRESS=10028
SIB_ENDPOINT_SECURE_ADDRESS=10038
SIB_MQ_ENDPOINT_ADDRESS=10040
SIB_MQ_ENDPOINT_SECURE_ADDRESS=10031
```

- Linux : `./ConfigEngine.sh modify-ports-by-portsfile -DWasPassword=`*`password`* `-DModifyPortsServer=`*`WebSphere_Portal`* `-DPortsFile=`*`full path to ports file`*
- Windows: `ConfigEngine.bat modify-ports-by-portsfile -DWasPassword=`*`password`* `-DModifyPortsServer=`*`WebSphere_Portal`* `-DPortsFile=`*`full path to ports file`*
- IBM i: `ConfigEngine.sh modify-ports-by-portsfile -DWasPassword=`*`password`* `-DModifyPortsServer=`*`WebSphere_Portal`* `-DPortsFile=`*`full path to ports file`*

5. Restart the WebSphere_Portal server.

Related tasks:

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

Completing the portal URI change started during installation

If you changed the context root on the **Configuration for IBM WebSphere Portal: Profile configuration details: Advanced** pane during installation, there are more steps to take to complete the change.

About this task

IBM WebSphere Portal Express and Web Services for Remote Portlets are installed with a default URI. You can change this URI after installation to better suit the requirements of your organization.

Notes:

- To change the WebSphere Portal Express URI: When you specify the context root, do not specify a value that is the same as a directory that exists in a portlet WAR directory. For example, you set the WebSphere Portal Express context root to `/images`. There is a portlet with the directory structure `/myPortlet.ear/myPortlet.war/images`. This issue might cause a conflict if the portlet encodes URI references to resources in its own `/images` directory. In this situation, the portlet would be unable to display images. WebSphere Portal Express looks for the image resources according to its own context root path instead of the directory path that is specified by the portlet WAR file.
- For changing the URI of a WSRP Producer portal: Changing the WSRP Producer context root does not require that you redeploy all portlets. Run the **`modify-servlet-path`** configuration task only.

Important: With Version 8, the URI of the context root for the WSRP Producer is `/wps/wsrp`. Before Version 8, this context root was `/wsrp`. If you migrated from an earlier version, you still might have WSRP Consumers that attempt to access the WSRP Producer with the previous context root (`/wsrp`). You can correct this issue in one of the following ways:

- Modify the context root for the WSRP Producer to `/wsrp`. This change enables the Consumers to access the Producer without requiring further changes to the Consumers.
- Update the configuration of the WSRP Consumers to use the new context root (`/wps/wsrp`).

- If you use IBM Web Content Manager Syndication, the Syndicators and Subscribers servers that refer to this Portal instance must be updated with the modified URI. Log on to the WebSphere Portal Express syndicating to this instance. Click the **Administration menu** icon. Then, click **Portal Content** > **Syndicators**. Click the edit icon of the Syndicator you want to edit. Update the URL with the new context root information. Then, log on to the WebSphere Portal Express subscribing to this instance. Click the **Administration menu** icon. Then, click **Portal Content** > **Subscribers**. Click the edit icon of the subscriber you want to edit. Update the URL with the new context root information.

Procedure

1. If necessary, start the WebSphere_Portal server in a stand-alone environment or the deployment manager and node agent in a clustered environment.
2. For combined cumulative fix 7 or earlier: Complete the following steps on the Deployment Manager server:
   a. Log on to the Deployment Manager WebSphere Integrated Solutions Console.
   b. Go to **Security** > **Global security**.
   c. Click **Trust association** in the Web and SIP security section.
   d. Click **Interceptors** in the Additional Properties section.
   e. Click **com.ibm.portal.auth.tai.HTTPBasicAuthTAI**.
   f. Edit the **urlBlackList** and **urlWhiteList** parameters with the new context path, for example:
      - **urlBlackList**: */wpsmodified*/myportalmodified*
      - **urlWhiteList**: */wpsmodified*/mycontenthandler*
   g. Click **Apply**.
   h. Save all changes.
   i. Log out of the Deployment Manager WebSphere Integrated Solutions Console.
3. Complete the following steps if you are using an external web server, such as an HTTP Server:
   a. Choose one of the following options that are based on your WebSphere Portal Express environment:

Table 36. `configurewebservername` command options

WebSphere Portal Express environment	Steps
Stand-alone configuration	Complete the following steps in a stand-alone configuration: 1. Copy the following script from the `plugin_root/bin` directory of the web server to the *wp_profile_root*`/bin` directory on your WebSphere Portal Express server: • Linux : `./configure`*`webservername`*`.sh` • Windows: `configure`*`webservername`*`.bat` • IBM i: `configure`*`webservername`*`.sh` where *webservername* is the web server definition name you defined previously when you configured the HTTP Server for WebSphere Portal Express, for example: `configurewebserver1.bat`. 2. Run the following command, from the *wp_profile_root*`/bin` directory: • Linux : `./configure`*`webservername`*`.sh` • Windows: `configure`*`webservername`*`.bat` • IBM i: `configure`*`webservername`*`.sh`
Idle standby configuration	Complete the following steps in a idle standby configuration: 1. Copy the following script from the `plugin_root/bin` directory of the web server to the `dmgr_profile/bin` directory on your Deployment Manager server: • Linux : `./configure`*`webservername`*`.sh` • Windows: `configure`*`webservername`*`.bat` • IBM i: `configure`*`webservername`*`.sh` where *webservername* is the web server definition name you defined previously when you configured the HTTP Server for WebSphere Portal Express, for example: `configurewebserver1.bat`. 2. Run the following command on the Deployment Manager server: • Linux : `./configure`*`webservername`*`.sh` • Windows: `configure`*`webservername`*`.bat` • IBM i: `configure`*`webservername`*`.sh`

b. Regenerate the web server plug-in in WebSphere Application Server. If you are using a remote web server, copy the generated `plugin-cfg.xml` file to the remote server.

   Important: Do not complete these steps if you are changing only the WSRP Producer URI.

c. Restart the web server.

d. Restart the WebSphere_Portal server.

4. For combined cumulative fix 7 or earlier: Complete the following steps to update the registered Application URI entries in the JCR.ICMSTJCRNODEREGISTER table:

   Cluster note: In a clustered environment, complete these steps on the primary node only.

   a. Stop the WebSphere_Portal server.
   b. Back up the database.
   c. Prior to CF04, start the WebSphere_Portal server. CF04 Starting with CF04, do not restart the WebSphere_Portal server.
   d. Complete the following steps to unregister the node types:
      - Open the `ibmcontentwcm.registernodetypes` file, which is in the `/WebSphere/PortalServer/wcm/prereq.wcm/config/nodetypes/` directory.
      - Change `<registerAction action="register"/>` to `<registerAction action="deregister"/>`.
      - Save your changes.
      - Run the following task:
        - Linux : `./ConfigEngine.sh action-register-wcm-nodetypes -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`*
        - Windows: `ConfigEngine.bat action-register-wcm-nodetypes -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`*
        - IBM i: `ConfigEngine.sh action-register-wcm-nodetypes -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`*
   e. Complete the following steps to register the node types:
      - Open the `ibmcontentwcm.registernodetypes` file, which is in the `/WebSphere/PortalServer/wcm/prereq.wcm/config/nodetypes/` directory.
      - Change `<registerAction action="deregister"/>` to `<registerAction action="register"/>`.
      - Update all lines that contain the `<ApplicationURI name="wps/mypoc/?view=auth&uri=wcm:oid:"/>` content.

        Change the name of the attribute value to reflect the new **`WpsContextRoot`** value that is found in the `wkplc.properties` file. For example, if the original value for **`WpsContextRoot`** was `wps` and the new value is `wp8`, change the lines to `<ApplicationURI name="wp8/mypoc/?view=auth&uri=wcm:oid:"/>`.
      - Run the following task:
        - Linux : `./ConfigEngine.sh action-register-wcm-nodetypes -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`*
        - Windows: `ConfigEngine.bat action-register-wcm-nodetypes -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`*
        - IBM i: `ConfigEngine.sh action-register-wcm-nodetypes -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`*
   f. Run the following SQL query to verify that the entries in the table now show the new URI:

      ```
      select * from JCR.ICMSTJCRNODEREGISTER
      ```

   g. Restart the WebSphere_Portal server.

5. Required if you use IBM Web Content Manager: Complete the following steps to manually change the JSP components in the Web Resources v70 Library:

Cluster note: In a clustered environment, complete these steps on the primary node only.

a. Log on to WebSphere Portal Express.
b. Go to **Applications** > **Content** > **Web Content Authoring**.
c. Under **Preferences**, select **Edit Shared Settings**.
d. Under **Library Selection**, add **Web Resources v70** to the **Selected Libraries** list.
e. Click **OK**.
f. Under **Item Views**, select **All Items** > **All** > **Components** > **JSP**.
g. Select every **JSP component** from the Web Resources v70 library and then click **Edit**.
h. Update the **Path** field for every JSP component with the new context root path.

   The JSP path includes two parts, which are separated by a semi-colon. The first part is the context path to the IBM Web Content Manager extensions web application and then the second part is the path to the JSP. Update the path to the web application.

   For example, the other path might be: `/wcmextension;/jsp/html/general/UpdateItem.jsp`. If you changed the context root to `mynewcontext`, change the old path to `/mynewcontext/wcmextension;/jsp/html/general/UpdateItem.jsp`.

6. Complete the following steps to edit the context root for every search collection:

   Attention: Edit the context root for each existing search collection.

   a. Log on to WebSphere Portal Express as the administrator.
   b. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
   c. Click **Search Collections**.
   d. Click the search collection that you want to update. For example: **Default Search Collection**.
   e. Click the **Edit Content Source** icon for the first content source in the list.
   f. Edit the URL listed in the **Collect documents link from the URL** with the new context root.
   g. Click **Save**.
   h. Edit the URL in each remaining content source and then save your changes.
   i. Start the WebSphere Portal Express crawler content source for each collection:
      - If the documents are not stored in the search collection but a schedule is defined for the crawler, the crawler automatically runs at the scheduled time. You can also start the crawler manually.
      - If the documents are already collected, select **Regather documents** to update the documents with the new context root information.
   j. Click **Collections from All Services** in the breadcrumb trail and select the next search collection to modify.

7. For combined cumulative fix 7 or earlier: Complete the following steps to change the context root for the **Seedlist_Servlet**:

   a. Log in to the WebSphere Integrated Solutions Console.

b. Go to **Applications** > **Application Types** > **WebSphere enterprise applications**.
c. Click the **Seedlist_Servlet** application link.
d. Click **Context Root For Web Modules**.
e. Change the context root and then click **OK**.
f. Save your changes.

8. Idle standby only: Resynchronize the nodes and restart the cluster.

Table 37. Steps to resynchronize the nodes and restart the cluster.

Cluster type	Steps
Idle standby	Complete the following steps if you have an idle standby environment: 1. Open the deployment manager WebSphere Integrated Solutions Console. 2. Click **System Administration** > **Nodes**, select the primary node from the list, and click **Full Resynchronize**. 3. Click **Servers** > **Clusters**. 4. Select the cluster and click **Stop**. 5. After the cluster stops, restart it by selecting the cluster. Then, click **Start**.

9. For combined cumulative fix 7 or earlier: Complete the following steps on the stand-alone server or on each node within your cluster to create the WebSphere environment variables that IBM Web Content Manager needs:
   a. Locate the `wkplc.properties` and `wkplc_comp.properties` files in the *wp_profile_root*`/ConfigEngine/properties` directory and create backup copies before you change any values.
   b. Use a text editor to open the `wkplc.properties` file and enter the appropriate value for your environment in the **`WpsContextRoot`** property.
   c. Save and close the file.
   d. Use a text editor to open the `wkplc_comp.properties` file and enter the appropriate value for your environment in the following properties:
      - **`WsrpContextRoot`**
      - **`WpsPersonalizedHome`**
      - **`WpsDefaultHome`**

      Attention: Do not enter the same value for **`WpsPersonalizedHome`** and **`WpsDefaultHome`**.
   e. Save and close the file.
   f. Run the following task to create the WebSphere environment variables for Web Content Manager:
      - Linux : `./ConfigEngine.sh create-wcm-servletpath-variables -DServerName=`*`your_application_server_name`* `-DWasPassword=`*`password`*
      - Windows: `ConfigEngine.bat create-wcm-servletpath-variables -DServerName=`*`your_application_server_name`* `-DWasPassword=`*`password`*
      - IBM i: `ConfigEngine.sh create-wcm-servletpath-variables -DServerName=`*`your_application_server_name`* `-DWasPassword=`*`password`*

Note: Check the output for any error messages before you proceed with the next task. If any of the configuration tasks fail, verify the values in the `wkplc.properties` and `wkplc_comp.properties` files.

10. Resynchronize the nodes and restart the cluster.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

Changing the portal URI after an installation

You can change the default portal Uniform Resource Identifier (URI) any time after you install IBM WebSphere Portal Express. Some applications have a fixed context root that cannot be changed.

About this task

IBM WebSphere Portal Express and Web Services for Remote Portlets are installed with a default URI. You can change this URI after installation to better suit the requirements of your organization.

Notes:

- To change the WebSphere Portal Express URI: When you specify the context root, do not specify a value that is the same as a directory that exists in a portlet WAR directory. For example, you set the WebSphere Portal Express context root to `/images`. There is a portlet with the directory structure `/myPortlet.ear/myPortlet.war/images`. This issue might cause a conflict if the portlet encodes URI references to resources in its own `/images` directory. In this situation, the portlet would be unable to display images. WebSphere Portal Express looks for the image resources according to its own context root path instead of the directory path that is specified by the portlet WAR file.
- For changing the URI of a WSRP Producer portal: Changing the WSRP Producer context root does not require that you redeploy all portlets. Run the **`modify-servlet-path`** configuration task only.

  Important: With Version 8, the URI of the context root for the WSRP Producer is `/wps/wsrp`. Before Version 8, this context root was `/wsrp`. If you migrated from an earlier version, you still might have WSRP Consumers that attempt to access the WSRP Producer with the previous context root (`/wsrp`). You can correct this issue in one of the following ways:
  - Modify the context root for the WSRP Producer to `/wsrp`. This change enables the Consumers to access the Producer without requiring further changes to the Consumers.
  - Update the configuration of the WSRP Consumers to use the new context root (`/wps/wsrp`).
- If you use IBM Web Content Manager Syndication, the Syndicators and Subscribers servers that refer to this Portal instance must be updated with the modified URI. Log on to the WebSphere Portal Express syndicating to this instance. Click the **Administration menu** icon. Then, click **Portal Content** > **Syndicators**. Click the edit icon of the Syndicator you want to edit. Update the URL with the new context root information. Then, log on to the WebSphere Portal Express subscribing to this instance. Click the **Administration menu** icon.

Then, click **Portal Content** > **Subscribers**. Click the edit icon of the subscriber you want to edit. Update the URL with the new context root information.

Cluster note: If you modify the URI in a clustered environment, complete the steps that are described here on the primary node only, except where specified differently. Also, verify that **AutoSynch** is set to a frequency of 1 minute.

Procedure

1. Complete the following steps to manually modify the WebSphere Portal Express context root:
   a. Stop the WebSphere_Portal server.
   b. Locate the `wkplc.properties` and `wkplc_comp.properties` files in the *wp_profile_root*`/ConfigEngine/properties` directory and create backup copies before you change any values.
   c. Use a text editor to open the `wkplc.properties` file and enter the appropriate value for your environment in the **WpsContextRoot** property.
   d. Save and close the file.
   e. Use a text editor to open the `wkplc_comp.properties` file and enter the appropriate value for your environment in the following properties:
      - **WsrpContextRoot**
      - **WpsPersonalizedHome**
      - **WpsDefaultHome**

      Attention: Do not enter the same value for **WpsPersonalizedHome** and **WpsDefaultHome**.
   f. Save and close the file.
   g. Start the WebSphere_Portal server in a stand-alone environment or the deployment manager and node agent in a clustered environment.
   h. Open a command prompt and change to the *wp_profile_root*`/ConfigEngine` directory.
   i. Complete the following steps to change the WebSphere Portal Express URI:
      1) To change the context root for the values that you entered in the **WpsContextRoot**, **WsrpContextRoot**, **WpsPersonalizedHome**, and or **WpsDefaultHome** properties, run the following task:
         - Linux :`./ConfigEngine.sh modify-servlet-path -DPortalAdminPwd=`*password* `-DWasPassword=`*password*
         - Windows: `ConfigEngine.bat modify-servlet-path -DPortalAdminPwd=`*password* `-DWasPassword=`*password*
         - IBM i: `ConfigEngine.sh modify-servlet-path -DPortalAdminPwd=`*password* `-DWasPassword=`*password*

         Note: Check the output for any error messages before you proceed with the next task. If any of the configuration tasks fail, verify the values in the `wkplc.properties` and `wkplc_comp.properties` files.
      2) Restart the WebSphere_Portal server.
   j. Run the following task to change the context root for the portlets:
      - Linux : `./ConfigEngine.sh modify-servlet-path-portlets -DPortalAdminPwd=`*password* `-DWasPassword=`*password*
      - Windows: `ConfigEngine.bat modify-servlet-path-portlets -DPortalAdminPwd=`*password* `-DWasPassword=`*password*

- IBM i: `ConfigEngine.sh modify-servlet-path-portlets -DPortalAdminPwd=password -DWasPassword=password`

Note: Check the output for any error messages before you proceed with the next task. If any of the configuration tasks fail, verify the values in the `wkplc.properties` and `wkplc_comp.properties` files.

2. For combined cumulative fix 7 or earlier: Complete the following steps on the Deployment Manager server:
   a. Log on to the Deployment Manager WebSphere Integrated Solutions Console.
   b. Go to **Security** > **Global security**.
   c. Click **Trust association** in the Web and SIP security section.
   d. Click **Interceptors** in the Additional Properties section.
   e. Click **com.ibm.portal.auth.tai.HTTPBasicAuthTAI**.
   f. Edit the **urlBlackList** and **urlWhiteList** parameters with the new context path, for example:
      - **urlBlackList**: */wpsmodified*/myportalmodified*
      - **urlWhiteList**: */wpsmodified*/mycontenthandler*
   g. Click **Apply**.
   h. Save all changes.
   i. Log out of the Deployment Manager WebSphere Integrated Solutions Console.
3. If necessary, start the WebSphere_Portal server in a stand-alone environment or the deployment manager and node agent in a clustered environment.
4. Complete the following steps if you are using an external web server, such as an HTTP Server:
   a. Choose one of the following options that are based on your WebSphere Portal Express environment:

Table 38. `configurewebservername` command options

WebSphere Portal Express environment	Steps
Stand-alone configuration	Complete the following steps in a stand-alone configuration: 1. Copy the following script from the `plugin_root/bin` directory of the web server to the *wp_profile_root*`/bin` directory on your WebSphere Portal Express server: • Linux : `./configure`*`webservername`*`.sh` • Windows: `configure`*`webservername`*`.bat` • IBM i: `configure`*`webservername`*`.sh` where *webservername* is the web server definition name you defined previously when you configured the HTTP Server for WebSphere Portal Express, for example: `configurewebserver1.bat`. 2. Run the following command, from the *wp_profile_root*`/bin` directory: • Linux : `./configure`*`webservername`*`.sh` • Windows: `configure`*`webservername`*`.bat` • IBM i: `configure`*`webservername`*`.sh`
Idle standby configuration	Complete the following steps in a idle standby configuration: 1. Copy the following script from the `plugin_root/bin` directory of the web server to the `dmgr_profile/bin` directory on your Deployment Manager server: • Linux : `./configure`*`webservername`*`.sh` • Windows: `configure`*`webservername`*`.bat` • IBM i: `configure`*`webservername`*`.sh` where *webservername* is the web server definition name you defined previously when you configured the HTTP Server for WebSphere Portal Express, for example: `configurewebserver1.bat`. 2. Run the following command on the Deployment Manager server: • Linux : `./configure`*`webservername`*`.sh` • Windows: `configure`*`webservername`*`.bat` • IBM i: `configure`*`webservername`*`.sh`

b. Regenerate the web server plug-in in WebSphere Application Server. If you are using a remote web server, copy the generated `plugin-cfg.xml` file to the remote server.

   Important: Do not complete these steps if you are changing only the WSRP Producer URI.

c. Restart the web server.

d. Restart the WebSphere_Portal server.

5. For combined cumulative fix 7 or earlier: Complete the following steps to update the registered Application URI entries in the JCR.ICMSTJCRNODEREGISTER table:

   Cluster note: In a clustered environment, complete these steps on the primary node only.

   a. Stop the WebSphere_Portal server.

   b. Back up the database.

   c. Prior to CF04, start the WebSphere_Portal server. CF04 Starting with CF04, do not restart the WebSphere_Portal server.

   d. Complete the following steps to unregister the node types:
      - Open the `ibmcontentwcm.registernodetypes` file, which is in the `/WebSphere/PortalServer/wcm/prereq.wcm/config/nodetypes/` directory.
      - Change `<registerAction action="register"/>` to `<registerAction action="deregister"/>`.
      - Save your changes.
      - Run the following task:
        - Linux : `./ConfigEngine.sh action-register-wcm-nodetypes -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`*
        - Windows: `ConfigEngine.bat action-register-wcm-nodetypes -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`*
        - IBM i: `ConfigEngine.sh action-register-wcm-nodetypes -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`*

   e. Complete the following steps to register the node types:
      - Open the `ibmcontentwcm.registernodetypes` file, which is in the `/WebSphere/PortalServer/wcm/prereq.wcm/config/nodetypes/` directory.
      - Change `<registerAction action="deregister"/>` to `<registerAction action="register"/>`.
      - Update all lines that contain the `<ApplicationURI name="wps/mypoc/?view=auth&uri=wcm:oid:"/>` content.

        Change the name of the attribute value to reflect the new **`WpsContextRoot`** value that is found in the `wkplc.properties` file. For example, if the original value for **`WpsContextRoot`** was `wps` and the new value is `wp8`, change the lines to `<ApplicationURI name="wp8/mypoc/?view=auth&uri=wcm:oid:"/>`.
      - Run the following task:
        - Linux : `./ConfigEngine.sh action-register-wcm-nodetypes -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`*
        - Windows: `ConfigEngine.bat action-register-wcm-nodetypes -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`*
        - IBM i: `ConfigEngine.sh action-register-wcm-nodetypes -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`*

   f. Run the following SQL query to verify that the entries in the table now show the new URI:

      ```
      select * from JCR.ICMSTJCRNODEREGISTER
      ```

   g. Restart the WebSphere_Portal server.

6. Required if you use IBM Web Content Manager: Complete the following steps to manually change the JSP components in the Web Resources v70 Library:

Cluster note: In a clustered environment, complete these steps on the primary node only.

a. Log on to WebSphere Portal Express.
b. Go to **Applications** > **Content** > **Web Content Authoring**.
c. Under **Preferences**, select **Edit Shared Settings**.
d. Under **Library Selection**, add **Web Resources v70** to the **Selected Libraries** list.
e. Click **OK**.
f. Under **Item Views**, select **All Items** > **All** > **Components** > **JSP**.
g. Select every **JSP component** from the Web Resources v70 library and then click **Edit**.
h. Update the **Path** field for every JSP component with the new context root path.

   The JSP path includes two parts, which are separated by a semi-colon. The first part is the context path to the IBM Web Content Manager extensions web application and then the second part is the path to the JSP. Update the path to the web application.

   For example, the other path might be: `/wcmextension;/jsp/html/general/UpdateItem.jsp`. If you changed the context root to `mynewcontext`, change the old path to `/mynewcontext/wcmextension;/jsp/html/general/UpdateItem.jsp`.

7. Optional: Update your custom themes to reference the correct Dojo context root.

   The default Dojo context root in WebSphere Portal Express is `/wps/portal_dojo`. After you run the **`modify-servlet-path`** and **`modify-servlet-path-portlets`** tasks, the Dojo context root is changed to include the new value in the **`WpsContextRoot`** parameter as the prefix. For instance, if the new **`WpsContextRoot`** value is `myco`, then the new Dojo context root becomes `/myco/portal_dojo`. If your theme includes hardcoded references to `"/wps/portal_dojo"`, update those references to the new context root. If you migrated a custom theme, you might find that it has references to `/portal_dojo` without the `/wps` prefix. Look for these references in both the WAR file and in the WebDAV storage for your theme.

   Cluster note: In a clustered environment, complete these steps on the primary node only.

8. Complete the following steps to edit the context root for every search collection:

   Attention: Edit the context root for each existing search collection.

   a. Log on to WebSphere Portal Express as the administrator.
   b. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
   c. Click **Search Collections**.
   d. Click the search collection that you want to update. For example: **Default Search Collection**.
   e. Click the **Edit Content Source** icon for the first content source in the list.
   f. Edit the URL listed in the **Collect documents link from the URL** with the new context root.
   g. Click **Save**.

h. Edit the URL in each remaining content source and then save your changes.
i. Start the WebSphere Portal Express crawler content source for each collection:
   - If the documents are not stored in the search collection but a schedule is defined for the crawler, the crawler automatically runs at the scheduled time. You can also start the crawler manually.
   - If the documents are already collected, select **Regather documents** to update the documents with the new context root information.
j. Click **Collections from All Services** in the breadcrumb trail and select the next search collection to modify.

9. For combined cumulative fix 7 or earlier: Complete the following steps if you changed the context root and you have existing search scopes:
   a. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**. Then, click **Search Scopes**.
   b. Delete the following search scopes:
      - **All Sources**
      - **Managed Web Content**
   c. Restart the portal server to re-create the search scopes with the correct context.
10. For combined cumulative fix 7 or earlier: Complete the following steps to change the context root for the **Seedlist_Servlet**:
    a. Log in to the WebSphere Integrated Solutions Console.
    b. Go to **Applications** > **Application Types** > **WebSphere enterprise applications**.
    c. Click the **Seedlist_Servlet** application link.
    d. Click **Context Root For Web Modules**.
    e. Change the context root and then click **OK**.
    f. Save your changes.
11. Idle standby only: Resynchronize the nodes and restart the cluster.

Table 39. Steps to resynchronize the nodes and restart the cluster.

Cluster type	Steps
Idle standby	Complete the following steps if you have an idle standby environment: 1. Open the deployment manager WebSphere Integrated Solutions Console. 2. Click **System Administration** > **Nodes**, select the primary node from the list, and click **Full Resynchronize**. 3. Click **Servers** > **Clusters**. 4. Select the cluster and click **Stop**. 5. After the cluster stops, restart it by selecting the cluster. Then, click **Start**.

12. For combined cumulative fix 7 or earlier: Complete the following steps on the stand-alone server or on each node within your cluster to create the WebSphere environment variables that IBM Web Content Manager needs:

a. Locate the `wkplc.properties` and `wkplc_comp.properties` files in the *wp_profile_root*`/ConfigEngine/properties` directory and create backup copies before you change any values.
b. Use a text editor to open the `wkplc.properties` file and enter the appropriate value for your environment in the **`WpsContextRoot`** property.
c. Save and close the file.
d. Use a text editor to open the `wkplc_comp.properties` file and enter the appropriate value for your environment in the following properties:
   - **`WsrpContextRoot`**
   - **`WpsPersonalizedHome`**
   - **`WpsDefaultHome`**

   Attention: Do not enter the same value for **`WpsPersonalizedHome`** and **`WpsDefaultHome`**.
e. Save and close the file.
f. Run the following task to create the WebSphere environment variables for Web Content Manager:
   - Linux : `./ConfigEngine.sh create-wcm-servletpath-variables -DServerName=`*`your_application_server_name`* `-DWasPassword=`*`password`*
   - Windows: `ConfigEngine.bat create-wcm-servletpath-variables -DServerName=`*`your_application_server_name`* `-DWasPassword=`*`password`*
   - IBM i: `ConfigEngine.sh create-wcm-servletpath-variables -DServerName=`*`your_application_server_name`* `-DWasPassword=`*`password`*

   Note: Check the output for any error messages before you proceed with the next task. If any of the configuration tasks fail, verify the values in the `wkplc.properties` and `wkplc_comp.properties` files.

13. Resynchronize the nodes and restart the cluster.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

Configuring managed pages

When you create a new installation of IBM WebSphere Portal Express 8.5, managed pages are enabled by default. However, you can also manually disable and enable the feature as needed.

About this task

Migration note: If you migrate from a previous version, managed pages are disabled by default, but you can enable the feature after migration.

"Enabling managed pages"
By default, support for managed pages is enabled for the default virtual portal. However, you can also manually enable this support if managed pages are disabled.

"Disabling managed pages" on page 379
Disable support for managed pages by running the `disable-managed-pages` configuration task.

"Transferring content associations to the Portal Site library" on page 380
When you enable manage pages, any web content pages that you have are converted to managed pages and added to the Portal Site library. However, the content that is associated with the web content pages remains in the original libraries. You can transfer this associated content to the Portal Site library with the `internalize-content-mappings` task.

"Enabling managed pages"
By default, support for managed pages is enabled for the default virtual portal. However, you can also manually enable this support if managed pages are disabled.

"Disabling managed pages" on page 379
Disable support for managed pages by running the `disable-managed-pages` configuration task.

"Transferring content associations to the Portal Site library" on page 380
When you enable manage pages, any web content pages that you have are converted to managed pages and added to the Portal Site library. However, the content that is associated with the web content pages remains in the original libraries. You can transfer this associated content to the Portal Site library with the `internalize-content-mappings` task.

Related concepts:

"Vanity URLs" on page 2094
You can associate vanity URLs with portal pages and labels. Vanity URLs are short URLs that people can easily remember. They are shorter than full WebSphere Portal Express URLs. They are sometimes also called marketing URLs. You can publish vanity URLs for marketing campaigns through different channels, such as email or print. This way, you can use vanity URLs to direct customers to a specific portal page or content item. Interested site visitors who want to view your campaign can then remember or copy the short vanity URL and type it into the browser address field.

Enabling managed pages

By default, support for managed pages is enabled for the default virtual portal. However, you can also manually enable this support if managed pages are disabled.

About this task

Important: Do not attempt to enable managed pages on a server where managed pages are already enabled. If you previously disabled managed pages and want to re-enable the feature, you must ensure that the Portal Site library is empty first. If you fail to remove the page artifacts from the previous configuration, the resulting portal might not work properly.

When supported, managed pages is enabled for a virtual portal, all pages in the virtual portal are copied into the Portal Site library in IBM Web Content Manager. However, the following pages are not treated as managed pages and are not copied:

- Administration pages, as identified by the label `ibm.portal.Administration` and its child pages
- Private pages

Each virtual portal has its own Portal Site library.

Note: To take advantage of the features available to managed pages in the user interface, your pages must use the Portal 8.5 theme.

Cluster consideration: In a cluster, you need to apply this procedure only to the primary node.

Attention: If you lost your JCR workspace and the backup was create before you created the virtual portal, restoring the workspace creates an inconsistency between the database domains. The **create-virtual-portal-site-nodes** task fails because the JCR workspace is missing. Run the **action-migrate-vps** task to correct the inconsistency with the database domains and to restore the JCR workspace.

Procedure

1. Start the portal server.
2. To enable support for managed pages, run the **enable-managed-pages** task from the *wp_profile_root*`/ConfigEngine` directory.

   Windows
   ```
   ConfigEngine.bat enable-managed-pages -DPortalAdminPwd=password
   -DWasPassword=password
   ```

   Linux
   ```
   ./ConfigEngine.sh enable-managed-pages -DPortalAdminPwd=password
   -DWasPassword=password
   ```

   IBM i
   ```
   ConfigEngine.sh enable-managed-pages -DPortalAdminPwd=password
   -DWasPassword=password
   ```

   After you run the **enable-managed-pages** task for the first time, the property **managed.pages** is created in the portal WP Configuration Service. The value of the property is set to `true`.
3. Restart the portal server.
4. To populate web content libraries with information about virtual portals in the system, run the **create-virtual-portal-site-nodes** task from the *wp_profile_root*`/ConfigEngine` directory. For each virtual portal, this task creates a library and a site area that is called `lost-found` for resources that cannot be properly located. If the library or site area exist, the task exits. By default, the task runs on all virtual portals in the system.

   Windows
   ```
   ConfigEngine.bat create-virtual-portal-site-nodes
   -DPortalAdminPwd=password -DWasPassword=password
   ```

   Linux
   ```
   ./ConfigEngine.sh create-virtual-portal-site-nodes
   -DPortalAdminPwd=password -DWasPassword=password
   ```

   IBM i
   ```
   ConfigEngine.sh create-virtual-portal-site-nodes
   -DPortalAdminPwd=password -DWasPassword=password
   ```
5. To populate web content libraries with information about the portal pages in the system, run the **create-page-nodes** task from the *wp_profile_root*`/ConfigEngine` directory.

This task can also be used when portal pages and managed pages artifacts in Web Content Manager are not synchronized. In this case, the task attempts to resynchronize the portal artifacts and web content artifacts, giving precedence to the portal artifacts.

Performance note: Depending on the amount of information in the system, the **create-page-nodes** task can take a long time to run. Because of the database load of the task, do not run the task frequently. The initial run of the task requires the most time, while subsequent runs typically require less time.

Note: If you have many pages, then it might be necessary to increase the soap client timeout. Edit the *wp_profile_root*/properties/soap.client.props file to change the **com.ibm.SOAP.requestTimeout** to 60000.
Attention: If your virtual portals have different administrative accounts, you cannot run the **create-page-nodes** task directly. You must run the task for every virtual portal, including the base virtual portal. Use the **VirtualPortalHost** or **VirtualPortalContext** parameter with the **create-page-nodes** task. Run the **list-all-virtual-portals** task to get a list of all your virtual portals. When you run the create-page-nodes task on the base virtual portal, set the **VirtualPortalContext** value to __NO__VP__ID__.

Windows
```
ConfigEngine.bat create-page-nodes -DPortalAdminPwd=password
-DWasPassword=password
```

Linux
```
./ConfigEngine.sh create-page-nodes -DPortalAdminPwd=password
-DWasPassword=password
```

IBM i
```
ConfigEngine.sh create-page-nodes -DPortalAdminPwd=password
-DWasPassword=password
```

By default, this task is run on all pages in all virtual portals. To limit this task to a specific virtual portal, identify the virtual portal by adding one of the following parameters to the command line. Each parameter requires the prefix -D on the command line.

VirtualPortalHost
Specify the host name of the virtual portal. For example, vp.example.com.

Important: If the host name of the virtual portal is the same as the host name of the default virtual portal, you must also specify the **VirtualPortalContext** property. You can specify the **VirtualPortalHost** property by itself only if the host name is unique.

VirtualPortalContext
Specify the virtual portal context that identifies the virtual portal. For example, vp1.

You can customize the task with the following optional parameters on the command line. Each parameter requires the prefix -D on the command line.

RunParallel
Indicate whether you want the task to run with multiple threads. A value of false indicates a single thread and is the default setting.

A value of true indicates multiple threads, as specified by the work manager **wpsJcrSyncWorkManager** in the WebSphere Integrated Solutions Console. Each thread requires a database connection. For optimal performance, ensure that your database connection pool supports at least as many connections as there are threads in the pool.

Excluded

Specify a list of unique names of page nodes to exclude from the creation process. Excluding a page also excludes its child pages. By default, the portal administration pages (`ibm.portal.Administration`) are excluded.

6. Optional: If you used web content pages before you enabled managed pages, you can transfer the content for those pages to the Portal Site library. If you plan to use the default page templates and store your web content in the Portal Site library, transfer the content for the template pages to the Portal Site library. For more information, go to *Transferring content associations to the Portal Site library*.
7. Optional: Ensure that users have appropriate access to the Web Content Manager REST virtual resource so they can use **Edit mode**. For example, they have `user` access.

Related concepts:

"Page templates" on page 1786
Content authors use the page templates to quickly create pages that are consistent with your site design. They do not have to waste time to configure settings that are probably consistent across your site, such as theme selection, page layout, and more.

Related tasks:

"Transferring content associations to the Portal Site library" on page 380
When you enable manage pages, any web content pages that you have are converted to managed pages and added to the Portal Site library. However, the content that is associated with the web content pages remains in the original libraries. You can transfer this associated content to the Portal Site library with the `internalize-content-mappings` task.

Disabling managed pages

Disable support for managed pages by running the `disable-managed-pages` configuration task.

About this task

Disabling managed pages has the following effects:

- By default, each virtual portal has its own specific workspace where content is stored. When you disable managed pages, only a single workspace for the default virtual portal is available. The workspaces of other virtual portals are still there, but you can no longer access them. Any system associations between pages in those virtual portals and their respective Portal Site libraries no longer work.

  Important: To preserve content in the other virtual portals, you must import or syndicate the libraries into the default virtual portal before disabling managed pages.
- You can still access the Portal Site library for the default virtual portal, but the library is no longer automatically synchronized with the page structure.
- Pages are no longer managed in IBM Web Content Manager, with the following implications:
  - No page drafts can be created.
  - No new versions of pages can be created.
  - Pages are no longer syndicated.

  - Access control changes that you perform in the portal interface are no longer applied to the portal page site area.
  - If you delete a page from the portal interface, the corresponding portal page site area is not deleted.
- If you create a page with either the **Basic** or **Articles** page template, the page has no web content association. This missing association can cause errors if you attempt to add content from the **Create Content** tab of the site toolbar. To use the sample web content items when managed pages are disabled, create a web content association on the page before attempting to add content.
- Having managed pages enabled is a mandatory requirement to have vanity URLs enabled. Therefore, if you have vanity URLs enabled and disable managed pages, vanity URLs are also disabled.

Procedure

1. Run the **disable-managed-pages** task from the *wp_profile_root*/ConfigEngine directory.

   Linux `./ConfigEngine.sh disable-managed-pages -DPortalAdminPwd=`*password* `-DWasPassword=`*password*

   IBM i `ConfigEngine.sh disable-managed-pages -DPortalAdminPwd=`*password* `-DWasPassword=`*password*

   Windows
   `ConfigEngine.bat disable-managed-pages -DPortalAdminPwd=`*password* `-DWasPassword=`*password*

   After running the **disable-managed-pages** task for the first time, the property **managed.pages** is created in the **WP WPConfigService** configuration service. The value of the property is set to `false`.
2. Restart the portal server.

Transferring content associations to the Portal Site library

When you enable manage pages, any web content pages that you have are converted to managed pages and added to the Portal Site library. However, the content that is associated with the web content pages remains in the original libraries. You can transfer this associated content to the Portal Site library with the `internalize-content-mappings` task.

About this task

Note: Administration pages are not intended to be managed pages and so are not included when you enable managed pages.

When you transfer the content association for a page to the Portal Site library, several things happen:

- The content that is referenced by the default content association for the page is copied to the portal page site area for the page. Only the default content association is affected; other content associations for the page are ignored.

  Note: Nested pages are not copied. Nested site areas are not copied in the following cases:
  - The nested site area is referenced by the default association of another page.
  - The nested site area has the same name as an existing site area for the same page.

- Template mappings and content elements that exist in the associated site area are copied over into the portal page. If the template mapping or element exists for the page, the copy is not performed.
- The default content setting for the portal page is modified to reference the copied content.
- The configuration of any web content viewers on the page is updated to reference the content that is stored in the portal page site area. However, viewer configurations that use content paths are not affected.

Procedure

To transfer content associations, run the `internalize-content-mappings` task from the *wp_profile_root*`/ConfigEngine` directory.

Windows
```
ConfigEngine.bat internalize-content-mappings
-DPortalPage=target_page -DIncludeDescendants=true_or_false
-DSynchronous=true_or_false -DPortalAdminPwd=password
-DWasPassword=password
```

Linux
```
./ConfigEngine.sh internalize-content-mappings
-DPortalPage=target_page -DIncludeDescendants=true_or_false
-DSynchronous=true_or_false -DPortalAdminPwd=password
-DWasPassword=password
```

IBM i
```
ConfigEngine.sh internalize-content-mappings
-DPortalPage=target_page -DIncludeDescendants=true_or_false
-DSynchronous=true_or_false -DPortalAdminPwd=password
-DWasPassword=password
```

The following properties must be specified either on the command line or in the `wkplc.properties` file.

PortalPage
The object ID or the unique page name of the page for which you want to transfer content. If the target page is contained in a virtual portal, you must identify the virtual portal by specifying either the **`VirtualPortalContext`** parameter or **`VirtualPortalHost`** parameter.

IncludeDescendants
Specify `true` to transfer content for the target page and any child pages. To transfer content only for the target page, specify `false`. If not specified, the default value is `true`.

Synchronous
Specify `true` to perform the transfer synchronously. To perform the transfer asynchronously, specify `false`. If not specified, the default value is `true`.

Verbose
Specify `true` to output additional information to the log. To generate basic log information, specify `false`. If not specified, the default value is `false`.

CollisionHandling
When you copy content to a page, specify the action to take if the content item exists. By default, that content item is not copied. If you set **`CollisionHandling`** to `replace`, the content item on the page is replaced with the content item to be copied to the page.

VirtualPortalContext
Specify the virtual portal context that identifies the virtual portal. For example, `vp1`.

VirtualPortalHost
Specify the host name of the virtual portal. For example, `vp.example.com`.

Important: If the host name of the virtual portal is the same as the host name of the default virtual portal, you must also specify the **`VirtualPortalContext`** property. You can specify the **`VirtualPortalHost`** property by itself only if the host name is unique.

PortalAdminPwd
The administrator password for WebSphere Portal Express.

WasPassword
The administrator password for WebSphere Application Server.

Example commands:

- Windows: `ConfigEngine.bat internalize-content-mappings -DPortalPage=example.page -DIncludeDescendants=true -DSynchronous=true -DPortalAdminPwd=password -DWasPassword=password`
- Linux : `./ConfigEngine.sh internalize-content-mappings -DPortalPage=example.page -DIncludeDescendants=true -DSynchronous=true -DPortalAdminPwd=password -DWasPassword=password`
- IBM i: `ConfigEngine.sh internalize-content-mappings -DPortalPage=example.page -DIncludeDescendants=true -DSynchronous=true -DPortalAdminPwd=password -DWasPassword=password`

Create a WebSphere Portal profile

During the installation process, the IBM Installation Manager creates the WebSphere Portal Express profile. If you are on WebSphere Portal Express Version 8.5 without a combined cumulative fix applied, then you can use this option in the Configuration Wizard to create an additional profile.

Configuration Wizard

On the Configuration Wizard home page, click **More Options** to find **Create a WebSphere Portal Profile**.

Important: You cannot complete this configuration option, if you have CF01 or a later combined cumulative fix applied.

Related concepts:

"Troubleshooting: Create a WebSphere Portal profile" on page 3594
View troubleshooting information for creating a WebSphere Portal Express profile.

Creating a new profile

After you answer questions and provide information about your system, the wizard generates a custom configuration procedure.

About this task

Depending on your environment, the wizard generates a configuration process. The following steps reflect all possible steps in the configuration process. The steps do not represent a literal configuration. The steps are provided as a reference.

If you click **View Step Command**, you can see the task and properties that are associated with each step in the wizard.

Procedure

1. Create the target profile for WebSphere Portal Express in the WebSphere Application Server

   Important: WebSphere Application Server Version 8.5.5.5 requires that fix PI37248 is installed when creating the managed portal profile. This step fails if PI37248 is not installed.

   Condition
   none

   ConfigEngine task
   none
2. Install the ConfigEngine into the target WebSphere Portal Express profile

   Condition
   none

   ConfigEngine task
   none
3. Register the WebSphere Portal Express components with the ConfigEngine

   Condition
   none

   ConfigEngine task
   `install`
4. Consolidate the properties files for WebSphere Portal Express components used in this configuration into a single properties file

   Condition
   none

   ConfigEngine task
   `consolidate-properties`
5. Prepare the profile for basic configuration

   Condition
   none

   ConfigEngine task
   `gather-runtime-property-files`

   `gather-portalserver-exes`

   `pre-basic-config`

   `copy-shared-objects`

   `enable-oob-security`
6. Validate the database connection and environment

   Condition
   none

   ConfigEngine task
   none
7. Deploy applications into the portal profile

Condition
none

ConfigEngine task
none

8. Configure the JCR, theme, and core runtime components of your portal server

   Condition
   none

   ConfigEngine task
   none

9. Deploy the administration portlets and pages to the portal

   Condition
   none

   ConfigEngine task
   none

10. Deploy the out-of-box pages and portlets to the portal

    Condition
    none

    ConfigEngine task
    none

11. Remove the application server (server1) from the profile

    Condition
    none

    ConfigEngine task
    none

12. Stop the portal server

    Condition
    none

    ConfigEngine task
    none

13. Collect the deployment manager augmentation files and profile templates that are required to build a cell

    Condition
    none

    ConfigEngine task
    none

14. Restart the WebSphere Portal Express server

    Condition
    none

    ConfigEngine task
    none

Remove a WebSphere Portal profile

Use the Configuration Wizard to remove a portal profile.

Configuration Wizard

On the Configuration Wizard home page, click **More Options** to find **Remove a WebSphere Portal Profile**.

Related concepts:

"Troubleshooting: Remove a WebSphere Portal profile" on page 3598
View troubleshooting information for creating a WebSphere Portal Express profile.

Removing a profile

After you answer questions and provide information about your migration, the wizard generates a custom configuration procedure.

About this task

Depending on your environment, the wizard generates a configuration process. The following steps reflect all possible steps in the configuration process. The steps do not represent a literal configuration. The steps are provided as a reference.

If you click **View Step Command**, you can see the task and properties that are associated with each step in the wizard.

Procedure

1. Manual Step: Prepare your system

   Condition
   none

   ConfigEngine task
   none
2. Remove portal node from cluster

   Condition
   You have a clustered node.

   ConfigEngine task
   none
3. Remove portal profile

   Condition
   none

   ConfigEngine task
   none
4. Stop the deployment manager

   Condition
   You selected to remove the deployment manager profile.

   ConfigEngine task
   none
5. Remove the deployment manager profile

   Condition
   You selected to remove the deployment manager profile.

   ConfigEngine task
   none

Managing your WebSphere Portal Express environment

After you install IBM WebSphere Portal Express, you can use the IBM Installation Manager function to manage your environment. The Installation Manager function consists of updating and modifying the environment. You can also uninstall or roll back the modifications you made to your environment.

About this task

Choose the appropriate topic to manage your WebSphere Portal Express environment:

CAUTION:
The information in these files is basic instructions. Before you make any changes to your system, review all readme files that are included with your fixes for complete instructions. Also, review the *Backup and Restore* **topic for instructions on backup and recovery of data files and databases.**

"Using the Installation Manager to install fixes"
Periodically fix packs are released to integrate product code fixes. Between fix pack releases, interim fixes may be recommended or required to ensure product reliability and stability. The Recommended fixes page provides links to fix pack and interim fix downloads, information about what is recommended and what is required, and links to recommended related product fixes. To find the most current service update information, see the Recommended fixes link.

"Using the Installation Manager to modify your environment" on page 387
You can use the Installation Manager function to add or remove the IBM WebSphere Portal Express profile. You can also use the Configuration Wizard to create and add, or remove a profile from your environment. See Create a WebSphere Portal profile and Remove a WebSphere Portal profile for information about using the Configuration Wizard to modify your environment.

"Using the Installation Manager to uninstall interim fixes" on page 388
After upgrading to an interim fix, you can use the IBM Installation Manager function to uninstall it.

"Using the Installation Manager to rollback fixes" on page 388
After upgrading to a fix pack or a cumulative fix, you can use the IBM Installation Manager function to rollback to the original installation.

Related concepts:

Chapter 8, "Backup and restore," on page 771
Backup and recovery of data files and databases is an essential operation for any business system, particularly for data and applications that run in production environments. Create and follow a plan for backing up and recovering data on all tiers of your IBM WebSphere Portal Express deployment. IBM Installation Manager must also be included in backup and recovery planning. If you back up the WebSphere Portal Express file structure and then install a fix pack, the WebSphere Portal Express and IBM Installation Manager become out of sync after you restore the WebSphere Portal Express file system. This condition is not recoverable.

Using the Installation Manager to install fixes

Periodically fix packs are released to integrate product code fixes. Between fix pack releases, interim fixes may be recommended or required to ensure product reliability and stability. The Recommended fixes page provides links to fix pack and interim fix downloads, information about what is recommended and what is required, and links to recommended related product fixes. To find the most current service update information, see the Recommended fixes link.

About this task

Use the IBM Installation Manager update function to update the product files, and then run the ConfigEngine command to update the profile. Complete the following steps to install fixes:

CAUTION:
The information in these files are basic instructions. Before making any changes to your system, review all readme files that are included with your fixes for complete instructions.

Note: A fix is considered an interim fix, a cumulative fix, or a fix pack.

Attention: During the installation, several temporary IBM WebSphere Application Server fixes are installed. When you install your first fix, a warning displays to uninstall the temporary fixes. Uninstall the temporary fixes and then install your fixes.

Procedure

1. Launch the Installation Manager.
2. Add the repository containing the fix.
3. Test the repository connection and provide authentication credentials, if necessary, to access the directory where the repositories are stored.
4. Select **Update** to install the fix.
5. Select or enter the fix level.
6. After accepting the license agreement and reviewing the summary information, run the process to install the fix.
7. When you have installed the fix, you must run the **APPLY-FIX** task from the *wp_profile_root*/ConfigEngine directory to update your profile.

   Note: If you have more than one profile, you must run the ConfigEngine command for each profile.

   Linux
   ```
   ./ConfigEngine.sh APPLY-FIX -DWasPassword=password
   -DPortalAdminPwd=password
   ```

   IBM i
   ```
   ConfigEngine.sh APPLY-FIX -DWasPassword=password
   -DPortalAdminPwd=password
   ```

   Windows
   ```
   ConfigEngine.bat APPLY-FIX -DWasPassword=password
   -DPortalAdminPwd=password
   ```

Related information:

Recommended fixes and updates for WebSphere Portal and Web Content Management

Using the Installation Manager to modify your environment

You can use the Installation Manager function to add or remove the IBM WebSphere Portal Express profile. You can also use the Configuration Wizard to create and add, or remove a profile from your environment. See Create a WebSphere Portal profile and Remove a WebSphere Portal profile for information about using the Configuration Wizard to modify your environment.

About this task

You can use the Installation Manager to remove only profiles that are created with the Installation Manager. If you use the Configuration Wizard to create and add a profile, you must you use the Configuration Wizard to remove that profile.

Complete the following steps to modify your environment using the Installation Manager:

Procedure

1. Launch the Installation Manager.
2. Select **Modify** to add or remove the portal profile.
3. Select the profile that you want to add or remove.
4. Review the summary information and then run the process to add or remove the profile.

Related concepts:

"Create a WebSphere Portal profile" on page 382
During the installation process, the IBM Installation Manager creates the WebSphere Portal Express profile. If you are on WebSphere Portal Express Version 8.5 without a combined cumulative fix applied, then you can use this option in the Configuration Wizard to create an additional profile.

"Remove a WebSphere Portal profile" on page 384
Use the Configuration Wizard to remove a portal profile.

Using the Installation Manager to uninstall interim fixes

After upgrading to an interim fix, you can use the IBM Installation Manager function to uninstall it.

About this task

Complete the following steps if you want to uninstall an interim fix from your environment:

CAUTION:
The information in these files are basic instructions. Before making any changes to your system, review all readme files that are included with your fixes for complete instructions.

Procedure

1. Launch the Installation Manager.
2. Select **Uninstall** to remove the interim fix.

   Note: Ensure that you select the fix level to uninstall and not the product.
3. Select or enter the interim fix level to which you want to return.
4. Clear the interim fix level from the WebSphere Portal Express product build.
5. Review the summary information, and then run the process to uninstall the interim fix.

Using the Installation Manager to rollback fixes

After upgrading to a fix pack or a cumulative fix, you can use the IBM Installation Manager function to rollback to the original installation.

About this task

Use the IBM Installation Manager rollback function to rollback the product files, and then run the ConfigEngine command to update the profile. Complete the following steps to install fixes:

CAUTION:
The information in these files are basic instructions. Before making any changes to your system, review all readme files that are included with your fixes for complete instructions.

Procedure

1. Launch the Installation Manager.
2. Select **Rollback** to remove the fix pack.
3. Select or enter the fix pack level to which you want to return.
4. Review the summary information and then run the process to remove the fix pack or cumulative fix.
5. When you have removed the fix, you must run the **PRE-ROLLBACK-FIX** task from the *wp_profile_root*/ConfigEngine directory to prepare your profile for rollback.

   Note: If you have more than one profile, you must run the ConfigEngine command for each profile.

   Linux
   ```
   ./ConfigEngine.sh PRE-ROLLBACK-FIX -DWasPassword=password
   -DPortalAdminPwd=password
   ```

   IBM i
   ```
   ConfigEngine.sh PRE-ROLLBACK-FIX -DWasPassword=password
   -DPortalAdminPwd=password
   ```

   Windows
   ```
   ConfigEngine.bat PRE-ROLLBACK-FIX -DWasPassword=password
   -DPortalAdminPwd=password
   ```
6. Run the **ROLLBACK-FIX** task from the *wp_profile_root*/ConfigEngine directory to rollback your profile.

   Note: If you have more than one profile, you must run the ConfigEngine command for each profile.

   Linux
   ```
   ./ConfigEngine.sh ROLLBACK-FIX -DWasPassword=password
   -DPortalAdminPwd=password
   ```

   IBM i
   ```
   ConfigEngine.sh ROLLBACK-FIX -DWasPassword=password
   -DPortalAdminPwd=password
   ```

   Windows
   ```
   ConfigEngine.bat ROLLBACK-FIX -DWasPassword=password
   -DPortalAdminPwd=password
   ```

Configuring the IBM License Metric Tool

IBM License Metric Tool monitors license compliance. It recognizes and monitors what product offerings and their versions, releases, and fix packs are installed and used on the system. It measures the processor value units (PVU) available to and used by these assets. The tool ensures compliance with IBM subcapacity licensing requirements and to demonstrate good IT governance. Information about installed software is collected from monitored computers by an agent that can be deployed

on a range of operating systems. It is stored on a central server in a DB2 database and can be accessed through pre-configured reports that are available from a web user interface.

Before you begin

Install and configure WebSphere Portal Express before you configure the IBM License Metric Tool.

Procedure

1. Download the IBM License Metric Tool agent and install it on each server that has IBM WebSphere Portal Express installed. Read Passport Advantage® for instructions on downloading the IBM License Metric Tool agent, prerequisites, and licensing process.
2. Complete the following steps to provide credentials to IBM License Metric Tool:
   a. Open a shell window.
   b. Change to the IBM License Metric Tool installation directory: *agent_home*/wasagent.
   c. Run the following task to start the WebSphere agent:
      - Linux : ./WASAgentClient.sh
      - IBM i: WASAgentClient.sh
      - Windows: WASAgentClient.bat
   d. Enter servers to list the WebSphere server IDs and locations.
   e. Enter credentials *id userid password* to provide credentials for the servers. Where *id* is the server ID and *userid* and *password* correspond to the WebSphere Application Server administrator user ID and password.

      Note: Whenever the IBM Tivoli® License Compliance Manager agent is stopped, rerun the WASAgentClient task and then reissue the credentials.
   f. Complete any additional configuration tasks after installation. Read the following topic for instructions: Configuring IBM License Metric Tool.
3. Follow the IBM License Metric Tool instructions that are provided with your customer agreement to complete the IBM License Metric Tool licensing configuration for WebSphere Portal Express and IBM Web Content Manager.
4. Ensure that the appropriate inventory signature file exists in the *PortalServer_root*/properties/version directory path. The file that is present depends on your license agreement and has a general format of IBM_*offering_nameversion_number*.swtag.

   Tip: Only one *filename*.swtag signature file can exist per product offering. Refer to the following table to determine the correct signature file.

   Note: If the wrong file is found, the wrong offering might be installed. You must uninstall that offering and install the correct offering. If necessary, contact IBM Support for assistance with this process.

Table 40. Product signature files

Product offering / Component name	IBM Tivoli License Manager signature file	Signature file's directory path
IBM WebSphere Portal Express Extend Version 8.5	IBM_WebSphere_Portal_Extend.8.5.0.swtag	The default path of the *PortalServer_root*/properties/version directory

Table 40. Product signature files (continued)

Product offering / Component name	IBM Tivoli License Manager signature file	Signature file's directory path
IBM WebSphere Portal Express Enable Version 8.5	IBM_WebSphere_Portal_Enable.8.5.0.swidtag	The UserData path of the *PortalServer_root*/properties/version directory
IBM WebSphere Portal Express Server Version 8.5	IBM_WebSphere_Portal_Server.8.5.0.swidtag	The UserData path of the *PortalServer_root*/properties/version directory
IBM Web Content Manager Version 8.5	IBM_Web_Content_Manager.8.5.0.swidtag	The UserData path of the *PortalServer_root*/properties/version directory
IBM Web Content Manager Standard Edition Version 8.5	IBM_Web_Content_Manager.8.5.0.swidtag	The UserData path of the *PortalServer_root*/properties/version directory
IBM WebSphere Portal Express Version 8.5	IBM_WebSphere_Portal_Express.8.5.0.swidtag	The UserData path of the *PortalServer_root*/properties/version directory

5. Verify that the `itlm.product` file exists in the *PortalServer_root*/version directory.

Upgrading your existing product offering

After your initial WebSphere Portal Express installation, you can purchase a license for an upgraded product offering.

About this task

Cluster note: In a clustered environment, run these steps on all the nodes.

Response file note: You can record and run a response file to upgrade your existing product offering. Read Use response files to install for information. Use the following steps when you record your response file. Ensure that you exit from the IBM Installation Manager program to finish your recording.

IBM i response file note: Run the sample response file to upgrade your existing product offering. Read Use response files to install for information. For IBM i, edit the appropriate sample response file that is in the `setup_root/responsefiles/iseries` directory. You cannot record the response file from your IBM i operating system. You can record a response file on another operating system. Use the following steps when you record your response file. Ensure that you exit from the IBM Installation Manager program to finish your recording. You must edit the response file that you recorded to add your IBM i specific parameters.

Procedure

1. Stop all the application servers.
2. Open the IBM Installation Manager and complete the following steps to add the content repository:
   a. Go to **File** > **Preferences** > **Repositories**.
   b. Select **Add Repositories**.
   c. Select **Browse** and go to the *offering-install-eimage*/Setup directory.
   d. Select the `repository.config` file.

e. Click **OK**.
f. Ensure that all necessary repositories are checked.
g. Ensure that the content repository is after the Server repository in the list.
h. Click **Test Connections** to ensure that the IBM Installation Manager can successfully access the directory where the repository is stored.
i. Select **Apply**.
j. Select **OK**.

3. On the main IBM Installation Manager panel, select **Install** to begin the installation process.
4. On the Install Package: Select packages to install panel, check the box for the additional package that you purchased. Then, click **Next**.
5. Accept the license agreement and then click **Next**.
6. On the **Location** panel, select **Use the existing package group**. Select the **IBM WebSphere Portal V85** package group name and then click **Next**.
7. On the Features panel, check the box for the additional feature that you purchased. Then, click **Next**.
8. Confirm the Summary panel information and then click **Install**.

What to do next

New profiles are enabled with the upgraded offering. Existing profiles are only updated if they were created with the IBM Installation Manager.

Removing the upgrade package: You can remove any new features and return to your previous installation offering. Start the IBM Installation Manager. On the main IBM Installation Manager panel, select **Uninstall**. Remove the upgrade package that you installed.

Web Content Manager

Set up a content server by installing IBM Web Content Manager in various deployments to provide robust and flexible environments for web content development and delivery. After you install the content server, more configuration steps must be completed according to the role that the server plays in your web content environment.

About this task

Configuration for Web Content Manager is managed through services that are defined as resource environment providers in IBM WebSphere Application Server. For each service, you can edit existing properties and add new properties through the WebSphere Integrated Solutions Console.

"Configuring a web content authoring environment" on page 393
Set up the authoring environment by installing the authoring portlet and enabling other features that are required to support the authoring environment.

"How to configure a web content staging environment" on page 411
Configure the staging environment to emulate the web content delivery environment and allow for testing before deployment.

"Configuring a web content delivery environment" on page 411
Set up your delivery environment by installing web content viewers and enabling any other required features.

"Reserved authoring portlet" on page 430
When you use the web content viewer or web content pages, some scenarios involve web content authoring tasks that are accomplished with authoring tools components. Authoring tasks are run through a special instance of the authoring portlet that is reserved specifically for these tasks and is installed on page that is hidden from the page navigation available to typical users.

"Further configuration options" on page 432
These configuration options are available to address installation requirements for other deployment scenarios.

"Configuring managed pages" on page 375
When you create a new installation of IBM WebSphere Portal Express 8.5, managed pages are enabled by default. However, you can also manually disable and enable the feature as needed.

"Managing tagging and rating for web content" on page 443
When you use tagging and rating with web content, the web content viewer provides extra scope options for the filtering of tagging and rating results. Because changes in the web content system can affect the accuracy of the tagging and rating information that is used by the portal, it is important to keep the scope information up to date by synchronizing the scopes regularly.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Configuring a web content authoring environment

Set up the authoring environment by installing the authoring portlet and enabling other features that are required to support the authoring environment.

"How to install the authoring portlet" on page 394
Pages that include the Web Content Manager authoring portlet and the local rendering portlet are created when you install WebSphere Portal. You run the authoring portlet configuration task only if you have previously uninstalled your authoring or local rendering portlets. The authoring portlet configuration task automatically creates Web Content Manager pages and installs the authoring portlet and local rendering portlets.

"Further authoring portlet configuration options" on page 395
Further authoring portlet configuration options can be specified by using the portlet administration view.

"Web content authoring options" on page 396
You can tailor the authoring behavior of your web content environment by changing configuration settings such as workflow, profiling, and version control.

"EditLive! editor toolbar configuration options" on page 401
Run these configuration tasks to change the configuration of the EditLive! editor toolbar.

"How to configure authoring portlet search" on page 402
You can change the configuration of the authoring portlet to change how search works.

"Increase timeouts to prevent save errors" on page 403
If users are experiencing timeout errors when they try to save items, you can increase the `total transaction lifetime timeout` setting of your WebSphere Portal Express server.

"Configuring remote server access for links" on page 404
Before you can add links to files and documents that are stored in remote content management systems into web content elements, you must configure your server with information about the remote system and the settings that are used to handle communication with the system.

"Setting up support for federated documents" on page 405
Before you can access metadata from federated documents, you need to configure access to the remote servers that contain the documents and specify information about the feeds or service documents that are used to retrieve the documents. You can also tune the cache settings that are used with the federated documents feature.

How to install the authoring portlet

Pages that include the Web Content Manager authoring portlet and the local rendering portlet are created when you install WebSphere Portal. You run the authoring portlet configuration task only if you have previously uninstalled your authoring or local rendering portlets. The authoring portlet configuration task automatically creates Web Content Manager pages and installs the authoring portlet and local rendering portlets.

1. Open a command prompt.
2. Run the `configure-wcm-authoring` task from the *wp_profile_root*`/ConfigEngine` directory.

   Linux
   ```
   ./ConfigEngine.sh configure-wcm-authoring
   -DPortalAdminPwd=password -DWasUserid=username
   -DWasPassword=password
   ```

   IBM i
   ```
   ConfigEngine.sh configure-wcm-authoring -DPortalAdminPwd=password
   -DWasUserid=username -DWasPassword=password
   ```

   Windows
   ```
   ConfigEngine.bat configure-wcm-authoring
   -DPortalAdminPwd=password -DWasUserid=username
   -DWasPassword=password
   ```
3. Log out of the portal and log back in.
4. Select **Web Content** from the product banner to access the authoring portlet.

Locale consistency

The language that is displayed in the authoring portlet is determined by the region or locale of the user. There are, however, some elements of the authoring portlet that use WebSphere Portal Express functions, such as date selection fields. They are displayed with the locale of the WebSphere Portal Express server. For this reason, the language and locales of the site being created, the client and server should be consistent.

If your site contains content in different languages, then a separate Web Content Manager authoring applications should be set up for each language on different WebSphere Portal Express Servers. These can then be combined into one site using a staging server.

Note: If a user changes their locale, any currently opened Web Content Manager dialogs will be closed. A user also must start a new session before it is displayed using the new locale. It is best practice to have the client's correct locale set before using Web Content Manager.

Further authoring portlet configuration options

Further authoring portlet configuration options can be specified by using the portlet administration view.

To add further authoring portlet configuration parameters:

- Log in to the portal as an administrator.
- Click the **Administration menu** icon. Then, click **Portlet Management** > **Portlets**.
- Search for the **Web Content Authoring portlet**.
- Open the configuration view.

You can add any of the following optional configuration parameters:

Table 41. Further authoring portlet configuration parameters

Parameter	Details
htmlfield.rows	Defines the number of rows to display in an HTML field that is used in an element design or presentation template. If not specified, the default setting of 15 rows is used.
htmlfield.columns	Defines the width of an HTML field that is used in an element design or presentation template. The width is defined as the number of characters to display per row. If not specified, the default setting of 85 characters is used.
htmlfield.embedded.rows	Defines the number of rows to display in an HTML field that is used in an element design or presentation template, but not an HTML component. If not specified, the number of rows that are defined by using *htmlfield.rows* is used.
htmlfield.embedded.columns	Defines the width of an HTML field that is used in an element design or presentation template, but not an HTML component. The width is defined as the number of characters to display per row. If not specified, the number of characters that are defined by using *htmlfield.columns* is used.
htmlfield.htmlcomponent.rows	Defines the number of rows to display in the HTML field that is used in an HTML component. If not specified, the number of rows that are defined by using *htmlfield.rows* is used.
htmlfield.htmlcomponent.columns	Defines the width of the HTML field that is used in an HTML component. The width is defined as the number of characters to display per row. If not specified, the number of characters that are defined by using *htmlfield.columns* is used.
htmlfield.presentation.rows	Defines the number of rows to display in the HTML field that is used in a presentation template. If not specified, the number of rows that are defined by using *htmlfield.rows* is used.

Table 41. Further authoring portlet configuration parameters (continued)

Parameter	Details
htmlfield.presentation.columns	Defines the width of the HTML field that is used in a presentation template. The width is defined as the number of characters to display per row. If not specified, the number of characters that are defined by using *htmlfield.columns* is used.
edit_live_custom_licence	Use this parameter to enter a custom license key to use in place of the OEM license for Ephox EditLive with this format:`Account ID,Domain,Expiration,License Key,Licensee,Product,Release`

Note: All users need to log off and login before any configuration changes appear in the authoring portlet.

Web content authoring options

You can tailor the authoring behavior of your web content environment by changing configuration settings such as workflow, profiling, and version control.

You define and manage authoring options in the **WCM WCMConfigService** service by using the WebSphere Integrated Solutions Console.

Enabling workflows

As default, the IBM Web Content Manager application will workflow content items only. You can update the `WCM WCMConfigService` service to enable workflows for different items.

To enable workflows, create a new property for the item type to which you want to apply workflow, and specify a value of `com.aptrix.pluto.workflow.WorkflowControl` for the property. You can enable workflow for the following item types:

- Content items (`control.Content`)
- Presentation templates (`control.Style`)
- Authoring templates (`control.Template`)
- Taxonomy items (`control.Taxonomy`)
- Categories (`control.Category`)
- Site area items (`control.SiteArea`)
- Library components (`control.Cmpnt`)

For example to enable workflow for authoring templates, you would add the following property:

- Property name: `control.Template`
- Value: `com.aptrix.pluto.workflow.WorkflowControl`

To disable workflow for an item type, you set the property to "false". For example, to disable workflow for authoring templates, you would do the following:

- Property name: `control.Template`
- Value: `false`

Note: If workflows are enabled for the following items, a workflow view is not available in the item views navigator.

- Site areas.
- Taxonomies and categories.
- Workflows, workflow stages, or workflow actions.

Individual items can still be moved through workflow stages by accessing them through the normal item views and approving them.

Note: Only content items can be moved through a workflow by using the web content API. If you enable workflows for other item types, you cannot approve or reject these items by using the API.

Enabling profiling

As default, the Web Content Manager application profiles content items only. You can update the `WCM WCMConfigService` service to enable profiling for different items.

To enable profiling, create a new property for the item type to which you want to apply profiling, and specify a value of `com.aptrix.pluto.taxonomy.ProfileControl` for the property. You can enable workflow for the following item types:

- Content items (`control.Content`)
- Presentation templates (`control.Style`)
- Authoring templates (`control.Template`)
- Taxonomy items (`control.Taxonomy`)
- Categories (`control.Category`)
- Site area items (`control.SiteArea`)
- Library components (`control.Cmpnt`)

For example to enable profiling for components, you would add the following property:

- Property name: `control.Cmpnt`
- Value: `com.aptrix.pluto.taxonomy.ProfileControl`

To disable profiling for an item type, you set the property to "false". For example, to disable profiling on components, you would do the following:

- Property name: `control.Cmpnt`
- Value: `false`

Version control options

By default version control is enabled with the following properties:

- `versioningStrategy.AuthoringTemplate`
- `versioningStrategy.Component`
- `versioningStrategy.Content`
- `versioningStrategy.PresentationTemplate`
- `versioningStrategy.Taxonomy`
- `versioningStrategy.Workflow`
- `versioningStrategy.Default`

You can use the following values to specify version control settings:

always
A version is saved every time a non-workflowed item is saved, or every time a workflowed item is published.

manual
Versions are saved when a user with at least editor access chooses to save a version. This setting causes the following changes in the interface:

- The **Save Version** button is available in the read mode of non-workflowed items and in workflowed items in the published state.
- The **Save and Version** button is available in the edit mode of non-workflowed items and in workflowed items in the published state.

never Disable version control for an item type.

If a version control strategy is not defined for an item type, then the version control strategy that is specified in by the `versioningStrategy.Default` property is used.

Inheritance options

By default, inheritance is automatically propagated down to each item. You can disable automatic inheritance by specifying the following property:

- Property name: `default.inherit.permissions.enabled`
- Value: `false`

When this setting is specified, it is applied only to new items. The inheritance on existing items remain unchanged.

Hierarchical item locking options

When a content item is being edited, it is locked. Other users are prevented from editing the content item until it is unlocked. Locking of site areas, taxonomies, and categories are configurable and is not enabled by default. To enable locking for hierarchical item types, specify the following properties: change the following parameters to "true":

Property name	Value
`wcm.authoringui.lock.taxonomies`	`true`
`wcm.authoringui.lock.categories`	`true`
`wcm.authoringui.lock.siteareas`	`true`
`wcm.authoringui.lock.projects`	`true`

When locking is enabled for site areas, you cannot create any children within the locked site area. For example, if a site area is locked, you cannot create any new site areas or content items within that site area until it is unlocked. This applies only to direct children of the locked parent. Items that are descendants of the children of a locked parent are not affected.

Defining valid mime types for the image element

You define the mime types of files that are allowed to be uploaded into the image element by using the `imageresourcecmpnt.allowedmimetypes` property and a list of mime types for the value. For example:

- Property name: `imageresourcecmpnt.allowedmimetypes`

- Value: `image/gif,image/jpeg`

This will prevent users uploading non-image files into the image element.

Active content filtering

Active content filtering provides the ability to strip specified HTML fragments from HTML entered in elements. This includes rich text and HTML elements. Active content filtering is configured using the `active.content.filtering.enable` property. By default, active content filtering is enabled. If enabled, this will prevent a user from introducing malicious code into a website such as cross site scripting.

For example, if a user entered this code into an HTML element:

```
Welcome
<a href="javascript:window.alert("boo!")">my link</a>
<script language="javascript">window.alert("boo 2!")</script>
Click the link for a surprise.
```

It would be changed to this when saved:

```
Welcome
<a href="<"- active content removed -->">my link</a>
<"- active content removed -->
Click the link for a surprise.
```

Setting the default child placement position

You can set the parameter **wcm.authoringui.childPlacementDefault** to specify the default placement of new content items.

Property value	Description
`start`	This setting will, by default, place a new content item as the first content item within a site area.
`end`	This setting will, by default, place a new content item as the last content item within a site area.

- If this parameter is not set, the default child position is "end".
- The default placement position that is specified in an authoring template overrides this setting for content items that are created with that authoring template.

Setting the size of the breadcrumb library dropdown

You can set the parameter **wcm.authoringui.breadcrumbLibrariesMaximum** to specify the number of libraries that are shown in the authoring interface breadcrumb. For example, `wcm.authoringui.breadcrumbLibrariesMaximum=16`

- If this parameter is not set, only the first 10 libraries are displayed.
- The value of this parameter must be an integer between 5 and 50.
- It is recommended that its value should be between 10 and 20.
- If more than this number of libraries exist, the remaining libraries are accessible by using the **Select from all libraries** option.

Expired items

By default, expired items are displayed alongside published and draft items.

To determine if expired items are listed in views, you can specify the `wcm.authoringui.showexpireditems` property in the `WCM` `WCMConfigService` service by using the WebSphere Integrated Solutions Console:

- If set to `true`, expired items are displayed alongside published and draft items.
- If set to `false`, only published and draft items are displayed.
- If not specified, this setting defaults to `true`.

Default inplace editing mode

These parameters are used to define the default inplace editing mode for text fields and rich text fields:

- `inplaceEdit.defaultModeForRichText`
- `inplaceEdit.defaultModeForText`

Use these values to specify the editing mode for text and rich text fields:

- Specify `inplace` to enable inplace editing of an element. Not all fields support inplace mode. If an element does not support inplace mode, dialog mode is used instead.
- Specify `embed` to enable embedded editing of an element. Not all fields support embed mode. If an element does not support embed mode, dialog mode is used instead.
- Specify `dialog` to enable editing in a dialog. This is useful for larger elements such as rich text elements that may not be suitable for inplace editing. All fields support dialog mode.

If this setting is not specified, inplace editing mode is used by text fields by default, and dialog editing mode is used by rich text fields by default.

CF03 CF03 From version 8.5.0 CF3, if this setting is not specified, embed mode is used by text fields and rich text fields by default.

The default inplace editing mode can be overridden in EditableElement tags using the *mode* parameter.

Note: The default rich text editor is always used when the modes 'inplace' or 'embed' are used. When the 'dialog' mode is used, the rich text editor selected in the authoring portlet settings, or in the content template for content items, is used.

Default css styles for inplace editable fields

The default css class used for inplace editable fields is **wcm-default-inplace-editable**.

This class can be overridden by adding the following setting:
`inplaceEdit.defaultClasses=class1 class2`

As many classes as required are added to this setting, separated by spaces.

You should base your custom classes on the default style sheet located at *AppServer_root*`\installedApps\nodename\wcm.ear\wcm-inplaceEdit.war\css\default-style.css`.

Note: Any classes specified on the EditableElement or EditableProperty tag will take precedence over this value.

If you need to use the default css class as well, add it to the list of classes. For example: `inplaceEdit.defaultClasses=wcm-default-inplace-editable class1 class2`

Restrict users ability to apply authoring templates

By default, only managers have access to the **Apply Template** button. To allow all users to apply a new authoring template to content items they have edit access to, change this setting to false:

- `wcm.authoringui.onlyShowApplyTemplateButtonForManagers=false`

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

EditLive! editor toolbar configuration options

Run these configuration tasks to change the configuration of the EditLive! editor toolbar.

Note: In a clustered environment, these tasks are only run on the primary server.

Reverting to the EditLive! editor version 7 toolbar

1. Open a command prompt.
2. Run the following command from the *wp_profile_root*/ConfigEngine directory:

   Windows

   ```
   ConfigEngine.bat configure-wcm-ephox-editor-v7-configuration
   -DWasPassword=password -DPortalAdminId=username
   -DPortalAdminPwd=password
   ```

   Linux

   ```
   ./ConfigEngine.sh configure-wcm-ephox-editor-v7-configuration
   -DWasPassword=password -DPortalAdminId=username
   -DPortalAdminPwd=password
   ```

   IBM i

   ```
   ConfigEngine.sh configure-wcm-ephox-editor-v7-configuration
   -DWasPassword=password -DPortalAdminId=username
   -DPortalAdminPwd=password
   ```

   Note: An administrator user name and password is not required if you already specified the portal administrator user name and password with the **PortalAdminId** and **PortalAdminPwd** settings in the wkplc.properties file.
3. Restart the server.

Note: To revert to the default editor toolbar, run the task that is named **remove-wcm-ephox-editor-v7-configuration** on the primary node only.

Using a custom EditLive! editor toolbar

1. The EditLive! editor uses a custom configuration file that is named config.xml.jsp to set custom parameters for the toolbar. Copy your custom configuration file to *wp_profile_root*\PortalServer\wcm\shared\app\config\ephox.

   Note: Sample configurations can be found in *PortalServer_root*\wcm\prereq.wcm\wcm\config\templates\shared\app\config\ephox

2. Open a command prompt.
3. Run the following command from the *wp_profile_root*/ConfigEngine directory:

   Windows

   ```
   ConfigEngine.bat configure-wcm-ephox-editor-custom-configuration
   -DWasPassword=password -DPortalAdminId=username
   -DPortalAdminPwd=password
   ```

   Linux

   ```
   ./ConfigEngine.sh configure-wcm-ephox-editor-custom-configuration
   -DWasPassword=password -DPortalAdminId=username
   -DPortalAdminPwd=password
   ```

   IBM i

   ```
   ConfigEngine.sh configure-wcm-ephox-editor-custom-configuration
   -DWasPassword=password -DPortalAdminId=username
   -DPortalAdminPwd=password
   ```

   Note: An administrator user name and password is not required if you already specified the portal administrator user name and password with the **PortalAdminId** and **PortalAdminPwd** settings in the wkplc.properties file.
4. Restart the server.

Note: To revert to the default editor toolbar, run the task that is named **remove-wcm-ephox-editor-custom-configuration** on the primary node only.

How to configure authoring portlet search

You can change the configuration of the authoring portlet to change how search works.

You define and manage authoring portlet search options in the WCM WCMConfigService service by using the WebSphere Integrated Solutions Console.

You can specify the following properties:

wcm.authoringui.advancedsearch.searchonselection

- Possible values are true or false.
- If set to true, when you click **Advanced Search**, an advanced search is automatically run based on any text that is entered in the basic search. If nothing is entered in the basic search, advanced search is not automatically run.
- If set to false, advanced search is not automatically run if there is existing text in the basic search field.
- If the property is not specified, this setting defaults to the false behavior.

wcm.authoringui.simplesearch.addstar

- Possible values are true or false.
- If set to true, a wildcard character is added to the end of text that is entered in the basic search. For example, searching for Span automatically searches for Span* and displays search results that have a title, description or keywords that begin with the word Span such as Spanish.
- If set to false, only exact matches to the text entered in the basic search field are searched for.
- If the property is not specified, this setting defaults to the false behavior.

wcm.authoringui.advancedsearch.addstar

- Possible values are `true` or `false`.
- If set to `true`, a wildcard character is added to the end of text that is entered in the advanced search. For example, searching for `Span` automatically searches for `Span*` and displays search results that have a title, description or keywords that begin with the word `Span` such as `Spanish`.
- If set to `false`, only exact matches to the text entered in the advanced search field are searched for.
- If the property is not specified, this setting defaults to the `false` behavior.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Increase timeouts to prevent save errors

If users are experiencing timeout errors when they try to save items, you can increase the `total transaction lifetime timeout` setting of your WebSphere Portal Express server.

The `total transaction lifetime timeout` setting is changed by using the WebSphere Integrated Solutions Console.

Go to **Servers** > **Server Types** > **WebSphere application servers** > *portal_server* > **Container Services** > **Transaction Service**.

The `total transaction lifetime timeout` setting must be changed to the same amount on all the servers in your web content system.

Alternatives to increasing server timeouts

Increasing the `total transaction lifetime timeout` setting might not always be the best solution to server timeouts. Increasing this setting too much might cause a drop in performance. Timeout errors that are generated when saving items occur when the current transaction finishes before the item is saved. If the item you are saving contains large amounts of data, it might be better to redesign the item rather than change the `total transaction lifetime timeout` setting:

Table 42. Alternatives to increasing server timeouts

Option	Details
Authoring Templates	If many elements are added to an authoring template, you might experience a timeout error when an item is saved. Instead of using a single authoring template, create multiple authoring templates that contain only those elements that are required for a specific task.
Presentation templates and components	You might receive timeout errors when you try to save presentation templates or components that contain large amounts of HTML or rich text in their designs. You can instead create multiple HTML or rich text components and then reference these components in the presentation templates or component designs.

Table 42. Alternatives to increasing server timeouts (continued)

Option	Details
Site areas and content items	You can receive timeout errors when you try to save site areas and content items that contain elements that use large amounts of HTML. You can instead create multiple HTML or rich text components and then reference these components in element designs. If many elements are added to a site area or content item, you can also experience a timeout error when you try to save the item. In this case, you can reduce the number of elements that are stored in the site area or content item.
Downloadable files	Another alternative to creating web content with large amounts of HTML or rich text is to provide information on your website in the form of downloadable files. These files can be stored as file resource elements.

Configuring remote server access for links

Before you can add links to files and documents that are stored in remote content management systems into web content elements, you must configure your server with information about the remote system and the settings that are used to handle communication with the system.

About this task

To prevent linking to unsafe servers, you need to specify a list of allowed domains that your portal can access by using the portal's AJAX proxy component. You can use the global AJAX proxy configuration to customize the outgoing HTTP traffic, such as applying specific HTTP timeout values or configuring an outbound HTTP proxy server. To do this, map the URL patterns for the Enterprise Content Manager server to the `federated_documents_policy` dynamic policy using the `WP ConfigService` configuration service.

Procedure

1. Log in to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Click **WP ConfigService**.
4. Under **Additional Properties**, click **Custom Properties**.
5. Click **New**, and enter the property name `wp.proxy.config.urlreplacement.federated_documents_policy.`*`suffix`*, and set the string value to the URL pattern of the Enterprise Content Manager server.

   For example, to enable the server to access information from the Enterprise Content Manager server `ecm.example.com` on port 10038 over HTTP, you would add the following property:

```
wp.proxy.config.urlreplacement.federated_documents_policy.1=http://ecm.example.com:10038/*
```

   Note: The value of the property key *`suffix`* can be any value as long as it is unique within the set of keys mapping to the `federated_documents_policy`.
6. Create additional properties as needed for any other Enterprise Content Manager servers that you need to access through the server.
7. Save your changes, and restart the portal server.

What to do next

If a user tries to access a server (for example, `www.example.com`) that has not been added to the list of allowed domains, the following message is displayed:

```
Access to remote server www.example.com has not been granted.
Please contact your system administrator.
```

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Inserting a link to remote content

IBM DB2 Content Manager publication library

Related information:

AJAX proxy configuration

Setting up support for federated documents

Before you can access metadata from federated documents, you need to configure access to the remote servers that contain the documents and specify information about the feeds or service documents that are used to retrieve the documents. You can also tune the cache settings that are used with the federated documents feature.

Before you begin

Authentication requirement: Before you can use the federated documents feature, you must complete one of the following steps:

- Enable single sign-on (SSO) in IBM WebSphere Application Server between the portal server and the content management system.
- Use a content management system that supports HTTP basic authentication, and enable a credential vault slot that stores the credentials to authenticate with the remote server.

  "Configuring access to remote systems for federated documents" on page 406
  To retrieve metadata information for documents on remote content management systems, configure the federated documents feature with information about the remote system and the settings that are used to handle communication with the system.

  "Configuring the federated documents feature" on page 407
  Configure the federated documents feature to specify information about the source servers for the documents that are available to users.

  "Cache tuning for federated documents" on page 410
  The federated documents feature uses the document list cache, the document data cache, and the feed type cache to manage information about the list of documents, the document data, and the types of feeds a server provides.

Configuring access to remote systems for federated documents:

To retrieve metadata information for documents on remote content management systems, configure the federated documents feature with information about the remote system and the settings that are used to handle communication with the system.

About this task

Because the federated documents feature uses the AJAX proxy component to access the remote content management system, you can use the global AJAX proxy configuration to customize the outgoing HTTP traffic, such as applying specific HTTP timeout values or configuring an outbound HTTP proxy server. You must list the individual content management servers to be accessed through the federated documents feature as allowed request targets in the AJAX proxy configuration, and enable LTPA cookie forwarding for those requests. To do this, map the URL patterns for the content management server to the `federated_documents_policy` dynamic policy using the `WP ConfigService` configuration service.

Procedure

1. Log in to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Click **WP ConfigService**.
4. Under **Additional Properties**, click **Custom Properties**.
5. Click **New**, and enter the property name `wp.proxy.config.urlreplacement.federated_documents_policy.`*`suffix`*, and set the string value to the URL pattern of the content management server.

   For example, to enable the federated documents feature to access information from the content management server `ecm.example.com` on port 10038 over HTTP, you would add the following property:

```
wp.proxy.config.urlreplacement.federated_documents_policy.1=http://ecm.example.com:10038/*
```

   Note: The value of the property key *`suffix`* can be any value as long as it is unique within the set of keys mapping to the `federated_documents_policy` dynamic policy.
6. Create additional properties as needed for any other content management servers that you need to access through the federated documents feature.
7. Optional: The federated documents feature can also consume arbitrary ATOM feeds. To enable this, you can map the URL prefix of the ATOM feed to the `default_policy` dynamic policy.
   a. Click **New**, and enter the property name `wp.proxy.config.urlreplacement.default_policy.`*`suffix`*, and set the string value to the URL pattern of the server providing the ATOM feed.

      For example, to enable the federated documents feature to access ATOM feeds from the server `www.example.com`, you would add the following property:

      ```
      wp.proxy.config.urlreplacement.default_policy.1=http://www.example.com/*
      ```

      The value of the property key *`suffix`* can be any value as long as it is unique within the set of keys mapping to the `default_policy` dynamic policy.

Important: To prevent security token forwarding to untrusted servers, be sure that you do not use the `federated_documents_policy` dynamic policy for those servers.

b. Create additional properties as needed for any other ATOM feed servers that you need to access through the federated documents feature.

8. Save your changes, and restart the portal server.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Configuring the federated documents feature:

Configure the federated documents feature to specify information about the source servers for the documents that are available to users.

About this task

When the portal retrieves documents from a remote server, authentication might be required to access the documents on the remote server. You can use several types of authentication:

- Single sign-on (SSO) between the portal and the remote server
- User name and password information in the user interface. Only HTTP basic authentication is supported for CMIS servers.
- Credential vault slots that handle HTTP authentication

In addition to enabling or disabling credential vault slots for authentication, you can identify the servers that provide documents. For each server, you can define characteristics such as the type of document that the server returns and the title that is used to identify the server.

Procedure

1. Log in to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Click **WP FederatedDocumentsService**.
4. Under **Additional Properties**, click **Custom Properties**.
5. Specify whether credential vaults slots are used for authentication with remote servers.

   Because you can access federated documents through either the personalization editor or the rich text editor that is provided with Web Content Manager, you can configure credential vault slots for each method independently.

   a. If you are accessing federated documents through the personalization editor, click **wp.federated.documents.pzn.vaultselection.enabled**. To enable credential vaults slots, set the value to `true`, or, to disable credential vault slots, set the value to `false`. By default, the value is `true`.

   b. If you are accessing federated documents through the rich text editor in Web Content Manager, click **wp.federated.documents.wcm.vaultselection.enabled**. To enable credential

vaults slots, set the value to `true`, or, to disable credential vault slots, set the value to `false`. By default, the value is `true`.

If you enable credential vault slots, users can select a credential vault slot in the user interface. You can also use the property **wp.federated.documents.*suffix*.vault.slot** to specify a default credential slot to be used with a given remote server.

6. Specify whether users can enter their own servers when they access remote content or can use only predefined servers that you configure.
   a. Click **wp.federated.documents.custom.server.enabled**.
   b. To allow users to enter their own servers, set the value to `true`. To prevent users from entering custom servers, set the value to `false`. When set to `false`, the user interface does not display the entry field for custom servers. By default, the value is `true`.
7. Specify whether documents from servers that support Document Services remote interfaces can be retrieved by the portal. Examples of products that support Document Services remote interfaces include IBM Content Manager, and IBM FileNet Content Manager.
   a. Click **wp.federated.documents.document.services.enabled**.
   b. To enable access to Document Services feeds, set the value to `true`. To disable access to Document Services feeds, set the value to `false`. If set to `false`, users can still access servers that support CMIS or Atom feeds, but connections to Document Services servers are not supported. `CF06` Prior to CF06, this value is set to `true` by default. In CF06 or higher, this value is set to `false` by default.
8. For each remote server that contains documents you want to access from the portal, configure the server URL, feed type, and extra optional properties.

   The value of the *`suffix`* portion of the property key is used to group related properties for each server. Use the same *`suffix`* value for properties that are related to the same server. The *`suffix`* can be any value when it is unique across the property keys.

   For each property, click **New** and enter the name and value:

   wp.federated.documents.*suffix*.url
   Value: The URL for an Atom feed or CMIS service document for the remote server. This property is required.

   wp.federated.documents.*suffix*.type
   Value:
   - The value `CMIS` indicates that the remote server provides a CMIS service document.
   - The value `DocumentServices` indicates that the remote server supports Document Services remote interfaces.
   - The value `ATOM` indicates that the remote server provides a generic Atom feed.

   If no value is specified, a default value of `CMIS` is used.

   wp.federated.documents.*suffix*.title.default
   Value: The title that is used to identify this source server in the user interface, when there is no resource bundle that is defined to provide title text. If no default title and no resource bundle are defined, the value of the wp.federated.documents.*suffix*.url property is used in the user interface.

wp.federated.documents.*suffix*.nls.resources
Value: The name of the resource bundle that contains the translated title and description that is used to identify this source server in the user interface. If this property is not defined, the default title is used. If no default title and no resource bundle are defined, the value of the wp.federated.documents.*suffix*.url property is used in the user interface.

wp.federated.documents.*suffix*.vault.slot
Value: The name of the credential vault slot that stores the credentials that are used for authentication with the remote server. Credential vault slots are set up and managed by the portal administrator. This property defines the default credential vault slot that is predefined in the user interface, although the user can also select a different slot if one is available. If this property is not defined, the user interface does not display a default credential vault slot, but you can still select a slot from the available list. This property is optional.

Note: The credential vault slot must contain the credentials that are required for authentication with the remote server.

wp.federated.documents.*suffix*.override.authentication.enabled
Value: `true` or `false`. When set to `true`, the user can change the authentication method for the server in the user interface. When set to `false`, the user interface does not display the field to change the authentication method. The default value is `true`.

9. Optional: Configure the amount of data that is returned for the summary metadata attribute of the document.
   a. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
   b. Click **WCM WCMConfigService**.
   c. Under **Additional Properties**, click **Custom Properties**.
   d. Click **wcm.pzn.ecm.max.field.length**, and enter the number of characters to be returned. If no value is specified, the default value is 128 characters.
10. Optional: Configure whether property changes are automatically loaded.
    By default, the Federated Documents service automatically reloads properties at a specified interval, without requiring you to restart the portal. You can change the automatic reloading behavior or modify the reloading interval.
    a. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
    b. Click **WP FederatedDocumentsService**.
    c. Under **Additional Properties**, click **Custom Properties**.
    d. Click **wp.federated.documents.document.service.reload.disabled**, and specify a value of `true` to disable automatic reloading of properties. The default value is `false`.
    e. Click **wp.federated.documents.document.service.reload.interval**, and specify the interval in seconds for reloading properties. The default value is 3 seconds.
11. Save your changes. The Federated Documents service automatically reloads any updated properties. If you disabled automatic reloading, restart the portal server.
12. If you enable credential vault slots, grant access to credential vault slots for all authenticated users.

a. Log in to the portal as an administrator.
b. Click the **Administration menu** icon. Then, click **Access** > **Resource Permissions**.
c. From the list of resource types, go to **Virtual Resources**.
d. For the **ADMIN_SLOTS** resource, click the **Assign Access** icon.
e. Edit the **User** role, and add the **All Authenticated Portal Users** group to the role.

Cache tuning for federated documents:

The federated documents feature uses the document list cache, the document data cache, and the feed type cache to manage information about the list of documents, the document data, and the types of feeds a server provides.

- The *document list cache* contains the list of document identifiers that are contained in the rule selection result of a specific user and a specific selection rule. The cache is activated by default with a default cache entry lifetime of 10 minutes.
- The *document data cache* contains the metadata of a specific document. The cache is activated by default with a default cache entry lifetime of 10 minutes.
- The *feed type cache* contains the type of feed for a feed URL. The feed type can be Document Services, CMIS, or ATOM. The cache is activated by default with a default cache entry lifetime of 24 hours.

To tune these caches, you can configure the Cache Manager Service (`WP CacheManagerService`) in the WebSphere Integrated Solutions Console by using the following properties:

- Document list cache: `cacheinstance.com.ibm.pzn.wcm.ecm.DocumentListCache`
- Document data cache: `cacheinstance.com.ibm.pzn.wcm.ecm.DocumentMetaDataCache`
- Feed type cache: `cacheinstance.com.ibm.pzn.wcm.ecm.FeedTypeServerCache`

Updates occurring on the remote content management system might not immediately be reflected on the portal side if there is a corresponding entry in the cache. The individual cache life time values determine the maximum time lag for corresponding updates.

Note:

- The time lag for new documents to be visible and deleted documents to be removed depends on the lifetime value for the configured document list cache.
- The time lag for updates in the metadata in a document (for example, changes to the document title) depends on the configured lifetime value for the document list cache.

The user-specific document list cache is explicitly invalidated each time the user logs in, so that the most current list of available document identifiers is available upon login.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

How to configure a web content staging environment

Configure the staging environment to emulate the web content delivery environment and allow for testing before deployment.

You define and manage staging environment options in the **WCM WCMConfigService** service by using the WebSphere Integrated Solutions Console.

- If your staging server is to be used purely as a holding server where changes to your site are accumulated before you syndicate these changes to a delivery environment, then you might need to review only the syndication settings of the staging server. In most cases, you would ensure that automatic syndication is disabled.
- If you are using your staging environment for user acceptance testing before you syndicate to a delivery environment, then you need to ensure that all other settings configured on your staging server match the same settings on the delivery server.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Configuring a web content delivery environment

Set up your delivery environment by installing web content viewers and enabling any other required features.

"Setting up site analysis for the Web Content Viewer"
To track usage data for the Web Content Viewer, you can configure the portal for site analysis logging for the Web Content Viewer.

"XML configuration interface parameters for the Web Content Viewer" on page 415
As with other portlets in your portal, you can use the XML configuration interface (**xmlaccess** command) to deploy and configure the Web Content Viewer. To simplify the configuration of the portlet with the XML configuration interface, the portlet parameters you can specify accept path values in addition to the standard IDs.

"Caching options" on page 418
Both IBM Web Content Manager generated web pages and content from external data sources can be cached by the Web Content Manager application. If used correctly, Web Content Manager caching can dramatically increase the performance of a site.

"Pre-rendering options" on page 425
You can enable pre-rendering so that content can be viewed either through a IBM Web Content Manager application or as a stand-alone site that is accessed through a web server.

Setting up site analysis for the Web Content Viewer

To track usage data for the Web Content Viewer, you can configure the portal for site analysis logging for the Web Content Viewer.

"Enabling the Web Content Viewer logger" on page 412
To take advantage of the site analysis logging available for the Web Content Viewer, you need to configure the **WP SiteAnalyzerLogService** service and activate the `SiteAnalyzerJSRPortletLogger` service.

"Site analysis example for the Web Content Viewer"
The site analysis log uses the NCSA Combined log format, which is a combination of NCSA Common log format and three extra fields: the referrer field, the user_agent field, and the cookie field. This example describes typical site analysis logging information for the Web Content Viewer.

Related tasks:

"Analyzing portal usage data" on page 1687
You can collect data about the usage of your portal and analyze them.

"Instrumenting web content for Active Site Analytics" on page 3399
You can collect information from web content for Active Site Analytics.

Enabling the Web Content Viewer logger:

To take advantage of the site analysis logging available for the Web Content Viewer, you need to configure the **WP SiteAnalyzerLogService** service and activate the `SiteAnalyzerJSRPortletLogger` service.

Before you begin

Before you activate the `SiteAnalyzerJSRPortletLogger` logger, you must ensure that site analysis is enabled for the portal in general, as described in Logging and analyzing server side site data.

Procedure

1. Log in to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Click **WP SiteAnalyzerLogService**.
4. Activate the `SiteAnalyzerJSRPortletLogger` logger through the **WP SiteAnalyzerLogService** by defining the parameter `SiteAnalyzerJSRPortletLogger.isLogging` and by setting the parameter value to `true`.
5. Save your changes, and restart the portal.

Site analysis example for the Web Content Viewer:

The site analysis log uses the NCSA Combined log format, which is a combination of NCSA Common log format and three extra fields: the referrer field, the user_agent field, and the cookie field. This example describes typical site analysis logging information for the Web Content Viewer.

The IBM WebSphere Portal Express site analysis log is:
wp_profile_root`/logs/WebSphere_Portal/sa_`*date_time*`.log`

where *date_time* is the date and time the file was created. The current (active) log file is named *sa.log*.

Note: The WP SiteAnalyzerService might be configured to use different file names.

The following example displays a sample entry in the site analysis log as it is written by the Web Content Viewer if the SiteAnalyzerJSRPortletLogger is enabled.

```
9.37.3.88 - jdoe [22/Nov/2008:22:11:27 +0100] "GET /Portlet/
5_8000CB1A00U6B02NVSPH1G20G1/Web_Content_Viewer_(JSR_286)/Web%20Content
%2fTestSite01%2fTestSiteArea01
```

```
%2fTestContent01?PortletPID=5_8000CB1A00U6B02NVSPH1G20G1&PortletMode=view
&PortletState=normal&RequestType=render&PUBLIC_CONTEXT=%2fWeb%20Content
%2fTestSite01 %2fTestSiteArea01%2fTestContent01 HTTP/1.1" 200 -1
"http://myserver.company.com/Page/ 6_8000CB1A00UR402F0JC25U1O25/MyPage"
"Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.18)
Gecko/20081029 Firefox/2.0.0.18" "JSESSIONID=0000JwIm04xm7btVLwzCj9Qo-uj:-
1"
```

The table describes each field of the log format:

Table 43. Explanation of each field in the log format

Field in the Example	Log Field Name and Explanation
9.37.3.88	**host** The IP address of the HTTP client that sent the request. **Important:** If there is a reverse proxy server between the client and the portal, the IP address that is logged is that of the reverse proxy server rather than the HTTP client. To log the IP address of the HTTP client, you must remove the reverse proxy server from the environment.
-	**rcf931** The identifier that is used to identify the client making the request. If the client identifier is not known, the field is set to the hyphen character (-).
jdoe	**username** The user ID for the client. If the user ID is not known, the field is set to the hyphen character (-).
[22/Nov/2008:22:11:27 +0100]	**date:time timezone** The date and time of the HTTP request.

Table 43. Explanation of each field in the log format (continued)

Field in the Example	Log Field Name and Explanation
`"GET /Portlet/[...] HTTP/1.1"`	**request** The HTTP method, the URI of the requested resource, and the version of HTTP used by the client. The URI is composed of the following elements: • The identifier `Portlet`. • The ID of the Web Content Viewer instance that is requested. • The administrative name of the Web Content Viewer (Note: This name is always the same unless the portlet has been cloned.). • The context path of the rendered Web Content Manager item encoded in UTF-8. • A query string containing the following parameters: **PortletPID** The ID of the Web Content Viewer instance that is requested. **PortletMode** The mode in which the portlet is rendered. Note that the Web Content Viewer writes log entries only in its view mode. **PortletState** The portlet window state. **RequestType** The request type (note that the Web Content Viewer writes log entries only for render requests). This is followed by a list of all request parameters that are available to the Web Content Viewer instance as UTF-8 encoded key-value-pairs.
`200`	**statuscode** The HTTP status code for the request.
`-1`	**bytes** The number of bytes of data that is transferred from the client as part of the request. A value of -1 indicates that the number of bytes is unknown.

Table 43. Explanation of each field in the log format (continued)

Field in the Example	Log Field Name and Explanation
"http://myserver.company.com/Page/6_8000CB1A00UR402F0JC25U1O25/MyPage"	**referrer** The referrer in case of portlet site analysis log entries identifies the portal page on which the Web Content Viewer instance is rendered.
"Mozilla/5.0 [...]"	**user_agent** The type of web browser that is used by the client.
"JSESSIONID=0000JwIm04xm7btVLwzCj9Qo-uj:-1"	**cookies** The name and value of a cookie that was sent to the client browser as part of the request. If multiple cookies were sent, the list is delimited by the semicolon character.

Related reference:

Understanding the site analysis log

XML configuration interface parameters for the Web Content Viewer

As with other portlets in your portal, you can use the XML configuration interface (**xmlaccess** command) to deploy and configure the Web Content Viewer. To simplify the configuration of the portlet with the XML configuration interface, the portlet parameters you can specify accept path values in addition to the standard IDs.

By default, the Web Content Viewer is configured with unique IDs. This has the advantage that the configuration does not break if an item is renamed of moved. However, when you are configuring a portlet with the XML configuration interface, it can be difficult to determine the unique ID of an item. When you are configuring the Web Content Viewer, you can reference web content items by their path, as well as by their IDs, by using the following parameters:

AUTHORINGTEMPLATE_OVERRIDE
Specifies the authoring templates of the profile section. The parameter can contain multiple values, which are separated by commas. The list can contain both ID and path values.

CATEGORY_OVERRIDE
Specifies the categories of the profile section. To list multiple categories, separate the categories by commas. You can use both ID values and path values.

SITEAREA_OVERRIDE
Specifies the site areas of the profile section. To list multiple site areas, separate the site areas by commas. You can use both ID values and path values.

WCM_BROADCASTS_TO
Specifies the link broadcasting setting for the Web Content Viewer. Values include:

- WCM_LINKING_DYNAMIC: Information about the web content item that is displayed in the Web Content Viewer is used to dynamically determine to which page the context is broadcast.
- WCM_LINKING_SELF: The context of the current Web Content Viewer is broadcast to other Web Content Viewers on the same portal page.
- WCM_LINKING_OTHER: The context of the current Web Content Viewer is broadcast to other Web Content Viewers on another portal page, as specified by the WCM_PORTAL_PAGE_ID parameter.
- WCM_LINKING_NONE: The context of the current Web Content Viewer is not broadcast to other Web Content Viewers.

WCM_COMPONENT_IDR
Specifies a library component and is only used if content type **Component** is selected.

WCM_CONTENT_COMPONENT
Specifies the name of the element to be displayed, when the WCM_CONTENT_TYPE parameter has the value CONTENT_COMPONENT.

WCM_CONTENT_CONTEXT_IDR
Specifies the content render context. It can be a content item or site area, as specified by the WCM_CONTENT_CONTEXT_TYPE parameter.

WCM_CONTENT_CONTEXT_TYPE
Specifies the type of the configured content context. Values include:
- CONTENT: Indicates that the content context is a content item.
- PARENT: Indicates that the content context is a site area.

WCM_CONTENT_TYPE
Specifies the item to be displayed. Values include:
- CONTENT: Indicates that the item to be displayed is a content item.
- COMPONENT: Indicates that the item to be displayed is a component.
- CONTENT_COMPONENT: Indicates that the item to be displayed is an element.
- SUMMARY: Indicates that the item is to be rendered with the summary presentation template. A summary presentation template can be specified in the item's content template.
- ALTERNATE: Indicates that the item is to be rendered with the alternative presentation template, as specified by the parameter WCM_DESIGN_IDR.

WCM_DESIGN_IDR
Specifies an alternative presentation template.

WCM_LISTENS_TO
Specifies how the Web Content Viewer is configured to receive links that are broadcast from other Web Content Viewers. Values include:
- WCM_LINKING_OTHER: Information is received from any Web Content Viewer broadcasting links.
- WCM_LINKING_SELF: Information is received only from this Web Content Viewer.
- WCM_LINKING_NONE: No information from other Web Content Viewers is received.

WCM_PAGE_TITLE
Used with the WCM_PAGE_TITLE_TYPE parameter, this parameter specifies the page title for the Web Content Viewer. Values include:

- The user-defined title for the page, if the WCM_PAGE_TITLE_TYPE parameter has a value of WCM_PAGE_TITLE_TYPE_GENERAL.
- The name of the resource bundle containing the title for the page, if the WCM_PAGE_TITLE_TYPE parameter has a value of WCM_PAGE_TITLE_TYPE_RESBUN.

WCM_PAGE_TITLE_TYPE

Specifies how the page title is displayed for the Web Content Viewer. Values include:

- WCM_PAGE_TITLE_TYPE_DEFAULT: The default title that is defined in the portal's administration interface is used.
- WCM_PAGE_TITLE_TYPE_GENERAL: A user-defined title is used, as specified by WCM_PAGE_TITLE parameter.
- WCM_PAGE_TITLE_TYPE_RESBUN: The title is defined in a resource bundle, as specified by WCM_PAGE_TITLE parameter.
- WCM_PAGE_TITLE_TYPE_DYN: The title is defined by the value of the **Display title** field for the content item that is displayed in the Web Content Viewer.
- WCM_PAGE_TITLE_TYPE_DYN_CONTENT_CMPNT: The title is defined by the value of an element of the content item that is displayed in the Web Content Viewer. You must also specify the element name using the **WCM_PAGE_TITLE** parameter.

WCM_PORTAL_PAGE_ID

Specifies the unique name or object ID of the page, which is the target for link broadcasts when the WCM_BROADCASTS_TO parameter is set to WCM_LINKING_OTHER.

WCM_PORTLET_TITLE

Used with the WCM_PORTLET_TITLE_TYPE parameter, this parameter specifies the portlet title for the Web Content Viewer. Values include:

- The user-defined title for the portlet, if the WCM_PORTLET_TITLE_TYPE parameter has a value of WCM_PORTLET_TITLE_TYPE_GENERAL.
- The name of the resource bundle containing the title for the portlet, if the WCM_PORTLET_TITLE_TYPE parameter has a value of WCM_PORTLET_TITLE_TYPE_RESBUN.

WCM_PORTLET_TITLE_TYPE

Specifies how the portlet title is displayed for the Web Content Viewer. Values include:

- WCM_PORTLET_TITLE_TYPE_DEFAULT: The default title that is defined in the portal's administration interface is used.
- WCM_PORTLET_TITLE_TYPE_GENERAL: A user-defined title is used, as specified by WCM_PORTLET_TITLE parameter.
- WCM_PORTLET_TITLE_TYPE_RESBUN: The title is defined in a resource bundle, as specified by WCM_PORTLET_TITLE parameter.
- WCM_PORTLET_TITLE_TYPE_DYN: The title is defined by the value of the **Display title** field for the content item that is displayed in the Web Content Viewer.
- WCM_PORTLET_TITLE_TYPE_DYN_CONTENT_CMPNT: The title is defined by the value of an element of the content item that is displayed in the Web Content Viewer. You must also specify the element name using the WCM_PORTLET_TITLE parameter.

When specifying a content path, you must begin with the forward slash character (/) followed by the library name, as indicated in the following examples of valid content paths:

```
/mylib/myfolder/mysitearea/mycontent
```

or

```
/mylib/mypresentationtemplate
```

Note: If you configure an item by its path rather than by its ID, the portlet configuration can become invalid if the item is renamed or moved. If an item has been configured by its path, the Web Content Viewer displays a small path icon after the item when you are in the **Edit Shared Settings** or **Configure** mode.

Important: When configuring an item by its path, you cannot build the path from the **Display title** fields of the items in the path. You must use the **Name** fields of the items when specifying the path.

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Caching options

Both IBM Web Content Manager generated web pages and content from external data sources can be cached by the Web Content Manager application. If used correctly, Web Content Manager caching can dramatically increase the performance of a site.

"Web content cache types"
Learn about the types of caching used by IBM Web Content Manager, basic web content caching, and advanced web content caching.

"Caching versus pre-rendering" on page 420
Content that is displayed in rendering portlets and through IBM Web Content Manager can be cached. An alternative to caching is the use of the pre-rendering feature. View the differences between each strategy.

"Strategies for expiring content" on page 421
Like caching strategies, a server's default expiration strategies can be set in the `WCM WCMConfigService` service by using the WebSphere Integrated Solutions Console. Custom expiring parameters can also be set in connect tags and URL requests to override a server's default expiration strategies.

"Web content cache configuration" on page 422
You can tailor the caching behavior of your web content environment by changing configuration settings such as the default cache type and expire settings.

"Data cache configuration" on page 425
Data caching is used to cache data that is retrieved by the IBM Web Content Manager application from external sources that use connect tags or by requests that are made through URLs.

Web content cache types:

Learn about the types of caching used by IBM Web Content Manager, basic web content caching, and advanced web content caching.

Basic web content caching

This option is the simplest caching option. The first time a web page is rendered by the Web Content Manager application, it is stored in a cache. Users then access this page from the cache until it expires. Only then is the web page rendered afresh. The main benefit of this scenario is improved performance. Basic caching is used only on static content that does not require "real-time" access.

Advanced web content caching

There are two major differences between basic caching and advanced caching:

- Advanced caching can cache pages based on different user profiles.
- Cache parameters in connect tags and URL requests can be used to override your server's default advanced web content caching settings that allow you to set custom cache settings for individual web pages or components.

Table 44. Advanced caching types

Advanced caching type	**Details**
Site caching	This type is the same as the basic web content cache except that cache parameters in connect tags and URL requests can be used to override your server's default advanced web content caching settings.
Session caching	When session caching is enabled, a copy of each web page a user visits is stored in the session cache. The User accesses the cached version of a web page until they start a new session, or until the cached web page is expired from the cache.
User caching	When user caching is enabled, a copy of each web page a user visits is stored in the user cache. The user accesses the cached version of a web page until the cached web page is expired from the cache.
Secured caching	Secured caching is used on sites where the item security features are used to grant different users access to different web pages and components based on the groups they belong to.
Personalized caching	Personalized caching is used to cache web pages of users who have the same "personalization profile". This means that users who have selected the same personalization categories and keywords, and who belong to the same group, share a single cache.

Default web content caching versus custom caching

Cache parameters in connect tags and URL requests can be used to override your server's default advanced web content caching settings allowing you to set custom cache settings for individual web pages or components.

In most cases, basic, site and session caching would only be used as your server's default web content cache. User, secured and personalized caching would mostly be used when using custom caching in connect tags and URL requests.

Note: If basic caching is used as your default web content cache, custom caching cannot be used.

Cache comparisons

Table 45. Basic caching versus advanced caching

Function	Basic caching	Advanced caching
Memory usage per item:	Medium	High
Performance improvement:	Very High	High
Custom caching available:	No	Yes
Connect tag processing:	No	Yes
Web Content Viewer Portlet:	No	Yes

Caching Personalization components:

Web content caching can sometimes be used with Personalization components but will depend on the conditions set in the personalization rule, or the resources used to determine the rule results. Cache testing will be required to determine if the content returned by your personalization component can be cached using web content caching.

Caching versus pre-rendering:

Content that is displayed in rendering portlets and through IBM Web Content Manager can be cached. An alternative to caching is the use of the pre-rendering feature. View the differences between each strategy.

A pre-rendered site can be viewed in two ways:

Using a web server
Viewing a pre-rendered site through a web server is similar to using basic caching because the displayed content is static and custom caching cannot be used.

Using Web Content Manager
Viewing a pre-rendered site through Web Content Manager is similar to using advanced caching because content can be dynamic and custom caching can be used.

Basic caching versus a pre-rendered site delivered with a web server

At first glance, the pre-rendering feature and basic caching do the same thing. There are some differences that determine which feature is the best for you.

The main difference between the two features is that the pre-rendering feature takes a snapshot of the entire site each time it is run. Basic caching caches only on a page-by-page basis. If performance is your main issue, then pre-rendering might be the answer. If not, then basic caching might be a better option.

Table 46. Basic caching versus a pre-rendered site delivered with a web server

Function	Basic caching	Pre-rendered site that is delivered with a web server
Performance:	Very fast	Extremely fast
Connect tag processing:	Yes	No
Custom caching:	Yes	No

Table 46. Basic caching versus a pre-rendered site delivered with a web server (continued)

Function	Basic caching	Pre-rendered site that is delivered with a web server
Memory requirements:	Low to Medium	Memory requirements depend on the web server that is being used.
Disk requirements:	Low to Medium	Potentially very high as the entire site must be able to fit on disk.
Unexpected broken links:	Yes As some pages can be cached at different times, there is a small chance that not all the links on a cached page are currently valid.	No The site is pre-rendered in a single batch, greatly reducing the chances of inconsistencies in the site.

Advanced caching versus a pre-rendered site delivered by using Web Content Manager

These options are similar. You might need to test both strategies before you which is best for your site.

Table 47. Advanced caching versus a pre-rendered site delivered by using Web Content Manager

Function	Advanced caching	Pre-rendered site that is delivered through Web Content Manager
Performance:	Fast when cached, but slower if the requested page has expired from the cache. (As tag processing has a cost, this depends on how many connect tags a page contains.)	Fast, but as tag processing has a cost, this depends on how many connect tags a page contains.
Connect tag processing:	Yes	No
Custom caching:	Yes	No
Memory requirements:	Medium to high.	Medium to high.
Disk requirements:	Medium to high.	Medium to high.
Unexpected broken links:	Yes As some pages may be cached at different times, there is a small chance that not all the links on a cached page will be currently valid.	No The site is pre-rendered in a single batch, greatly reducing the chances of inconsistencies in the site.

Strategies for expiring content:

Like caching strategies, a server's default expiration strategies can be set in the `WCM WCMConfigService` service by using the WebSphere Integrated Solutions Console. Custom expiring parameters can also be set in connect tags and URL requests to override a server's default expiration strategies.

Note: If basic caching is used as your default web content cache, custom expiration cannot be used.

In most cases the expiry schedule is based around how often the source content is updated. So, if the source content is updated hourly, then each cache would be expired hourly. If the source content is updated daily, then each cache would be expired daily.

Beyond these examples, a different expiry schedule would be used. If your web pages were only updated weekly, or monthly, you would still schedule your caches to expire daily. Otherwise, when your source content was updated, it might take up to a week for it to appear on your site.

Caching expiries versus workflow expiries

The expiration parameter in a workflow is not related to the `Expires` parameter in IBM Web Content Manager caching. A page that is set to expire at midnight as part of a workflow will only do so if it has not already been saved in a cache. The page remains in the cache until expired by the Web Content Manager application regardless of the `Expires` setting in a workflow.

Related concepts:

"Cache expire parameters" on page 1842
You use the "expires" parameter in IBM Web Content Manager tags and URLs to specify how long to maintain data in the cache before it is expired. When data expires from a cache, the next request for the data will be retrieved from the original server. The **`expires`** parameter is not mandatory.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Web content cache configuration:

You can tailor the caching behavior of your web content environment by changing configuration settings such as the default cache type and expire settings.

You define and manage web content cache options in the `WCM WCMConfigService` service by using the WebSphere Integrated Solutions Console.

Go to **Resources** > **Resource Environment** > **Resource Environment Providers** > **WCM WCMConfigService** > **Custom Properties**.

Setting the default web content cache type

The default web content caching environment for your web content server is specified by the following properties:

- `connect.businesslogic.defaultcache`
- `connect.moduleconfig.ajpe.contentcache.defaultcontentcache`

Table 48. Caching parameters

Parameter	**`defaultcache`** value	**`defaultcontentcache`** value
No caching:	`false`	`None`

Table 48. Caching parameters (continued)

Parameter	defaultcache value	defaultcontentcache value
Basic cache:	true	Not specified
Site caching:	false	Site
Session caching:	false	Session
User caching:	false	User
Secured caching:	false	Secured
Personalized caching:	false	Personalized

Further default web content cache parameters

Web content cache configuration settings are specified by the following properties in the WCM WCMConfigService service.

Table 49. Cache properties per cache type

Cache Type	Properties
Basic cache:	connect.businesslogic.defaultcacheexpires connect.businesslogic.defaultcache
Advanced cache: All	connect.moduleconfig.ajpe.contentcache.defaultcontentcache connect.moduleconfig.ajpe.contentcache.contentcacheexpires
Advanced cache: Session cache only	connect.sessioncacheconfig.memcachesize

Table 50. Cache properties details

Cache Property	Details
contentcacheexpires	This property sets the default expiry for all advanced caches. It can be either a relative period or an absolute date and time.
defaultcache	If true, basic caching is enabled. If false or missing, advanced caching is enabled.
defaultcacheexpires	This property sets the default expiry for the basic cache. It can be either a relative period or an absolute date and time.
defaultcontentcache	If the advanced cache is enabled, the default advanced cache is set here.
resourceserver.browserCacheMaxAge	This property is used to define the maximum time that an item is stored in a web browser cache.
resourceserver.maxCacheObjectSize	This property is used to define the maximum size of objects that can be cached in kilobytes. By default this property is set to 300.

Additional Cache Keys

These additional cache keys are used to customize and optimize both basic and advanced caching.

Table 51. Additional Keys

Key	Purpose	Prerequisites
locale	Enable caching based on browser locale.	8.5
portalcontext	Enable caching based on portal context.	8.5 CF02 (PI20951)
portalmapping	Enable caching based on portal mapping.	8.5 CF02 (PI20951)
portletcontext	Enable caching based on portlet context.	8.5 CF02 (PI20951)
deviceclass	Enable caching based on device type.	8.5 CF04 (PI27550)

To configure these additional cache keys, add the required caching keys to the value of the property, using commas to separate each value:

Basic Caching:
connect.businesslogic.cache.additionalcachekeys.requestattributes

Advanced Caching:
connect.moduleconfig.ajpe.contentcache.additionalcachekeys.requestattributes

Cache expire time formats

When you use the cache expire settings that are listed in Table 3, you can specify either a relative time, or absolute time:

- REL {integer-value}{units}
- ABS {date-format-string}

{units} =

- d|D for days
- m|M for months
- s|S for seconds
- h|H for hours

{date-format-string} =

- Mon, 06 Nov 2000 09:00:00 GMT
- Monday, 06-Nov-00 09:00:00 GMT
- Mon Nov 6 09:00:00 2000
- 6 Nov 2000 9:00 AM

Note: The last two formats assume GMT.

Examples:

- `contentcacheexpires="REL 300S"`
- `contentcacheexpires="ABS Mon, 06 Nov 2000 09:00:00 GMT"`

Data cache configuration:

Data caching is used to cache data that is retrieved by the IBM Web Content Manager application from external sources that use connect tags or by requests that are made through URLs.

You define and manage data cache options in the `WCM WCMConfigService` service by using the WebSphere Integrated Solutions Console.

Specify the following properties for data cache options:

`connect.connector.httpconnector.defaultcache`
Used when no cache is specified in a request for data. Possible values are `true` or `false`. If `true`, the data is stored in the site cache.

`connect.connector.httpconnector.defaultcacheexpires`
The expiry date/time for items added to a cache (site or session) if the expiry date/time is not specified in the request.

`connect.connector.sqlconnector.defaultcache`
Determines whether to cache data by default or not. Possible values are `true` or `false`.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Pre-rendering options

You can enable pre-rendering so that content can be viewed either through a IBM Web Content Manager application or as a stand-alone site that is accessed through a web server.

You define and manage pre-rendering options in the **WCM WCMConfigService** service by using the WebSphere Integrated Solutions Console.

Pre-rendering automatically

Although you can manually pre-render a website through the URL interface, you can also configure pre-rendering to run automatically when the server starts.

1. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
2. Click **WCM WCMConfigService**.
3. Under **Additional Properties**, click **Custom Properties**.
4. Edit the **`connect.businesslogic.module`** property, and append `cacher` to the value. For example:

   ```
   web,mail,default,ajpe,federatedproxy,ajpecatselect,memberfixer,workflowenablement,
   itemdispatcher,plutouploadfile,plutodownloadfile,synd,subs,syndication,
   refreshallitems,unlocklibrary,custom,data,clearversions,clearhistory,
   reseteventlog,cacher
   ```

5. Ensure that **`connect.businesslogic.module.cacher.autoload`** is set to true.
6. Save your changes and restart the server.

Enable pre-rendering for sites viewed by using Web Content Manager

This option is used when you are accessing the pre-rendered site through Web Content Manager. This increases performance as static content is accessed from the pre-rendered site, but dynamic content is still rendered through Web Content Manager.

To enable users to access the pre-rendered site through a Web Content Manager application, specify the **connect.businesslogic.module.default.class** property in the **WCM WCMConfigService** service by using the WebSphere Integrated Solutions Console.

- Property name: **connect.businesslogic.module.default.class**
- Value: `com.aptrix.cacher.CacherModule`

Note: You cannot use the local rendering portlet (Web Content Viewer) when pre-rendering is set as the default module.

Enable pre-rendering for stand-alone sites

This option is used when you are using Web Content Manager to generate a pre-rendered site, but are not using Web Content Manager to view the pre-rendered site. You need to use a web server to view the pre-rendered site.

Specify the **connect.businesslogic.module.cacher.class** property in the **WCM WCMConfigService** service by using the WebSphere Integrated Solutions Console.

- Property name: **connect.businesslogic.module.cacher.class**
- Value: `com.aptrix.cacher.CacherModule`

Specify the following properties to configure caching. Default values are listed, although you can tailor these values as needed. Unless you explicitly set a value for a property, the default value is used.

connect.moduleconfig.cacher.destdir
Value: `${USER_INSTALL_ROOT}/PortalServer/wcm/ilwwcm/cacher`

Base directory under which each site cache is created. There is one subdirectory created for each site.

Important: If the prerenderer is run with the **connect.moduleconfig.cacher.overwritecache** property set to `true`, any files in the **connect.moduleconfig.cacher.destdir** path that were not written in the last run of the pre-renderer is deleted. For this reason, ensure that the **connect.moduleconfig.cacher.destdir** path is only used for storing rendered content and that it does not contain any other data that cannot be re-created.

connect.moduleconfig.cacher.tempdir
Value: `${USER_INSTALL_ROOT}/PortalServer/wcm/ilwwcm/cacher/temp`

The temporary directory that is required to build the site cache before moving the data over to the base directory specified by the `connect.moduleconfig.cacher.destdir` property.

connect.moduleconfig.cacher.delay
Example value: 1

This property is used to set the time, in seconds, between requesting a page while caching.

connect.moduleconfig.cacher.busydelay
Example value: 5

This property is used to set the time, in seconds, of the busy delay setting. This is used if running within the busy start to busy end period. Otherwise the delay setting is used.

connect.moduleconfig.cacher.busystart/connect.moduleconfig.cacher.busyend
Example value: 9:00 am/5:00 pm

These settings determine the times between which the busy delay setting is used. Enter an absolute time as shown.

connect.moduleconfig.cacher.overwritecache

true The pre-renderer overwrites files in the destdir cache directory (then delete unneeded files). This results in a progressive change in site content as seen by the user. This is the default value.

false The first time a site is pre-rendered, the cached site files are added to the destination directory. As changes are made to the site through the authoring portlet, the new version of the site is gradually cached in the temporary directory and the old site remains in the destination directory. After the cacher has finished caching the site completely, the contents of the temporary directory are moved to the destination directory that will then contain both old and new versions of the cached site.

Note: A value of false should not be used if a web server is used to display the pre-rendered data because some web servers lock the data directories.

connect.moduleconfig.cacher.rendereruser
Example value: Anonymous.

This determines the user to be used to render the Web Content Manager content. Either type Anonymous or Administrator or a specific user or group name.

The site is pre-rendered based on this user's security rights. If the user specified here does not have access to a particular component, it is not pre-rendered.

connect.moduleconfig.cacher.task.cacherurl
Example value: http://${WCM_HOST}:${WCM_PORT}/${WCM_CONTEXT_ROOT}/connect/

The full URL to be used as the replacement for the connect servlet in pre-rendered pages. The URL ends with the string specified in **connect.moduleconfig.cacher.task.servletpath** property if it is not blank. The context of cacherurl is used when generating a URL with pre-rendering. This property is not used when a page belongs to a site that has not already been pre-rendered at a site level by the scheduled task or through a SRV=cacheSite request.

connect.moduleconfig.cacher.task.servletpath
Example value: /connect

The path of the substituted connect servlet that is defined in the **connect.moduleconfig.cacher.task.cacherurl** property. This property can remain blank if the cacherurl context should be used unchanged.

connect.moduleconfig.cacher.defaultcontentname
Example value: `index.html`

This sets the name of the default or home file that is used when accessing the pre-rendered site. This normally would be `index.html`.

connect.moduleconfig.cacher.task.siteareas
Example value: `LibraryA/SiteAreaA,LibraryB/SiteAreaB,SiteAreaC`

The site areas within a Web Content Manager environment to cache are entered here, separated by commas. This property provides the option of specifying the library in addition to the site area. If the library is specified, the pre-renderer looks for the site area in that library. If no library is specified, the default library is used, as specified in the `defaultLibrary` property.

Note: If any of your site area names contain commas, you must create separate parameters for each site area using this format: `connect.moduleconfig.cacher.task.siteareas.N`

N represents a different integer for each parameter. For example, if you want to pre-render a site area named "SiteArea,Red" and a site named "Site,Yellow", you would need to create the following parameters:

```
connect.moduleconfig.cacher.task.siteareas.1=MyLib/SiteArea,Red
connect.moduleconfig.cacher.task.siteareas.2=Site,Yellow
```

connect.moduleconfig.cacher.task.interval.recurrence
connect.moduleconfig.cacher.task.interval.startdelay
The CacherModule can be set to run after a recurring number of minutes.

recurrence:
Example value: `10`.

The recurring period in minutes for a recurring task.

startdelay:
Example value: `1`

The delay in minutes prior to starting the first recurring task.

Note: If you do not configure pre-rendering to start automatically when the server starts, pre-rendering at intervals does not work until you manually trigger the cacher module.

connect.moduleconfig.cacher.task.scheduled.times
Example value: `3:00 am`

Alternately, the CacherModule can be set to run at certain times. Enter a series of absolute times, separated by commas.

Important: When specifying time values, be sure you conform to the format `H:MM am|pm`, including the use of the colon (`:`) character and the space. Incorrectly specified values prevent pre-rendering from functioning properly.

Note: If you do not configure pre-rendering to start automatically when the server starts, pre-rendering at scheduled times does not work until you manually trigger the cacher module.

Pre-rendering resources

connect.moduleconfig.cacher.useTieredResourceFolders

Value: `false`

All resources, such as images and file resources, are stored under the following folder:

`CACHER_DIR\LIBRARY\SITEAREA\resources`

By default, each individual resource is saved under its own folder. For example, a resource with the ID of "7961d78049717f29bc57fee5670e9d7b" will be stored under this folder:

`CACHER_DIR\LIBRARY\SITEAREA\resources\`**`7961d78049717f29bc57fee5670e9d7b`**

You can change this behavior so that resources are stored under a tiered set of sub-folders based on the first two characters of the resource ID by changing the value of **connect.moduleconfig.cacher.useTieredResourceFolders** to `true`. For example, a resource with the ID of "7961d78049717f29bc57fee5670e9d7b" will be stored under this folder:

`CACHER_DIR\LIBRARY\SITEAREA\resources\7\9\`

All other resources that whose IDs begin with "79" will also be stored under this folder. This is done to reduce the number of sub-folders under the "resources" folders.

Minimum configuration settings

You can pre-render sites either manually, or to a schedule, or at intervals. These are the minimum settings required for each type of pre-rendering.

Manual:

- `connect.moduleconfig.cacher.rendereruser`
- `connect.businesslogic.module.cacher.class`

Scheduled:

- `connect.moduleconfig.cacher.rendereruser`
- `connect.businesslogic.module.cacher.class`
- `connect.businesslogic.module.cacher.autoload`
- `connect.moduleconfig.cacher.task.siteareas`
- `connect.moduleconfig.cacher.task.scheduled.times`
- `connect.businesslogic.module`

At intervals:

- `connect.moduleconfig.cacher.rendereruser`
- `connect.businesslogic.module.cacher.class`
- `connect.businesslogic.module.cacher.autoload`
- `connect.moduleconfig.cacher.task.siteareas`
- `connect.moduleconfig.cacher.task.interval.recurrence`
- `connect.businesslogic.module`

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some

of these services.

Reserved authoring portlet

When you use the web content viewer or web content pages, some scenarios involve web content authoring tasks that are accomplished with authoring tools components. Authoring tasks are run through a special instance of the authoring portlet that is reserved specifically for these tasks and is installed on page that is hidden from the page navigation available to typical users.

The following tasks use the reserved authoring portlet:

- Selecting a web content folder when you create or edit the properties of a web content page.
- Configuring the web content viewer, such as selecting the content item to display.
- Using inline editing or authoring tools components that are rendered in the web content viewer.

Authoring tasks are run in a separate window that opens from the current page, but you can configure the behavior of authoring tools components to redirect users to the hidden page that contain the reserved authoring portlet.

Ensuring the availability of the reserved authoring portlet

If either the authoring portlet instance or the hidden portal page is not available or if the user lacks the permission to access either of them, the authoring tasks that require the reserved authoring portlet fail, causing web content pages and the web content viewer to be unusable. For this reason, you must be careful when you administer the reserved authoring portlet and the hidden portal page.

The following conditions are essential for the proper function of the reserved authoring portlet:

- Users must have the User role on the hidden portal page.
- Users must have the User role on the reserved authoring portlet.
- The reserved authoring portlet must be the only portlet that is on the hidden portal page.
- The unique name of the hidden portal page must be `com.ibm.wps.hiddenpage.wcm.Authoring_Portlet`.
- The unique name of the portlet window of the authoring portlet instance on the hidden portal page must be `com.ibm.wps.hiddenpage.wcm.control.Authoring_Portlet`.

Availability problems that are related to the reserved authoring portlet or the hidden portal page are identified by the following symptoms:

- The `SystemOut.log` file for the portal server contains error messages that are referenced by the authoring portlet or hidden page. For example:

```
EJPDB0124E: The specified string
[com.ibm.wps.hiddenpage.wcm.Authoring_Portlet] can neither be
deserialized as an object ID nor resolved as a unique name.
EJPDB0124E: The specified string
[com.ibm.wps.hiddenpage.wcm.control.Authoring_Portlet] can neither be
deserialized as an object ID nor resolved as a unique name.
```

- When a separate window is opened from the current page to run the authoring task, the new window displays the following message:

  `Error 400: EJPPH0006E: The resolution of a URI failed. Refer to the stack trace for more detailed information.`
- When a separate window is opened from the current page to run the authoring task, the new window is empty.
- When the user is redirected to another portal page to run the authoring task, the user is redirected to the default portal page instead of the page that contains the reserved portlet.
- When the user is redirected to another portal page to run the authoring task, the user is redirected to an empty page.

If any of these problems occur, verify that the conditions for proper operation of the reserved authoring portlet and hidden portlet page are fully implemented.

Note: If the reserved authoring portlet or the hidden portlet page are removed inadvertently, you can deploy them again using the `action-install-wcm-hidden-authoring` configuration task.

"Configuring the reserved authoring portlet"
The reserved authoring portlet is essential to the proper operation of web content pages and the web content viewer, so it is important that the configuration of the reserved authoring portlet is similar to the configuration of other instances of the authoring portlet.

Configuring the reserved authoring portlet

The reserved authoring portlet is essential to the proper operation of web content pages and the web content viewer, so it is important that the configuration of the reserved authoring portlet is similar to the configuration of other instances of the authoring portlet.

Procedure

1. Log in to the portal as an administrator.
2. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
3. Search for the page with the unique name of `com.ibm.wps.hiddenpage.wcm.Authoring_Portlet`.
4. Click the **Edit Page Layout** icon for the page.
5. Select **Edit shared settings** from the portlet menu, and specify any settings for the reserved authoring portlet. The available settings and the process for updating them is the same for the reserved authoring portlet as it is for any other instance of the authoring portlet.

   Note: Changes made to the reserved authoring portlet with the **Edit shared settings** mode affect only the reserved authoring portlet and no other instances of the authoring portlet. To ensure a consistent authoring experience, you can make the same changes to other authoring portlet instances by using the **Edit shared settings** mode for each instance. Alternatively, you can make the same changes to every instance of the authoring portlet by using the **Configure** mode from a single instance. Changes that you make in the **Configure** mode also affect the reserved authoring portlet.
6. Save your changes.

Further configuration options

These configuration options are available to address installation requirements for other deployment scenarios.

"Web content substitution variables"
IBM Web Content Manager uses several substitution variables that are defined in the configuration for IBM WebSphere Application Server.

"Setting scoped configuration settings for virtual portals" on page 433
Web Content Manager configuration settings can be scoped for individual virtual portals.

"Disabling Workflow Actions" on page 434
Disable workflow action on servers that do not require workflows to be processed, such as a subscriber. This strategy can improve performance and reduce the number of versions that are created for each item.

"Enabling connect tags" on page 435
Enable connect tags to reference web content components and apply customized caching to the components.

"Enabling email" on page 435
To use the email workflow action, you must configure Web Content Manager to use your SMTP server.

"Defining alternative administrators for multi-realm configurations" on page 436
Web Content Manager requires a user to run various system tasks such as initialization, and background tasks such as syndication. By default, this system user is the configured JCR domain administrator. In scenarios where the configured user realm does not contain the domain administrator then an alternative user must be provided.

Web content substitution variables

IBM Web Content Manager uses several substitution variables that are defined in the configuration for IBM WebSphere Application Server.

If you need to modify these variables, use the WebSphere Integrated Solutions Console for the application server. If you are working with a managed cell or cluster, use the WebSphere Integrated Solutions Console for the deployment manager.

Table 52. Web content substitution variables

Variable	Description
`WCM_CONTEXT_ROOT`	The context root for the enterprise application for Web Content Manager. Example: `wps/wcm`
`WCM_HOST`	The fully qualified host name of the server. Example: `www.example.com`
`WCM_ILWWCM_HOME`	This variable is the directory where the Web Content Manager application is installed Example: *`PortalServer_root`*`/wcm`
`WCM_PORT`	The port number that is used to access the portal. Example: `10038`

Table 52. Web content substitution variables (continued)

Variable	Description
WCM_SCHEMA	The database schema name of the JCR domain that is configured for use with IBM WebSphere Portal Express. Example: jcr
WCM_SEARCHSEED_CONTEXT_ROOT	The context root for the Search Seed portlet. Example: wps/wcmsearchseed
WCM_WEB_APP_HOME	The directory path where the ilwwcm.war file is located.
WCM_WPS_CONTEXT_ROOT	The context root or base URI for the portal. All URLs beginning with this path is reserved for the portal. Example: wps http://hostname.example.com:10038/wps/portal
WCM_WPS_DEFAULT_HOME	The default portal page. This page is the page for users who are not logged in. Example: portal http://hostname.example.com:10038/wps/portal
WCM_WPS_PERSONALIZED_HOME	The portal page for users who are already logged in to the portal. This page cannot be accessed by anonymous users. Example: myportal http://hostname.example.com:10038/wps/myportal

Setting scoped configuration settings for virtual portals

Web Content Manager configuration settings can be scoped for individual virtual portals.

Procedure

1. Log in to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers** > **WCM WCMConfigService** > **Custom properties**.

   Cluster note: If you are using this web content server as part of a cluster, ensure that you use the WebSphere Integrated Solutions Console for the deployment manager when you edit configuration properties.
3. Add a configuration setting for each virtual portal that requires unique configuration settings. You can set this setting for either the host name or context path:

   Host name:

   vp.*uniquename*.hostname=*ExistingVPHost*

   Context path:

   vp.*uniquename*.context=*ExistingVPContext*

4. To add a scoped configuration setting for a virtual portal, use this format:
   PropertyKey.vp.*uniquename=override_value*

   This setting overrides the *property key* in the virtual portal that is specified in the *unique name*.
5. Restart the server or cluster.

Example

If your virtual portal was named "yellowportal", you might create a scoped configuration for it using this format.

```
vp.yellow.context=yellowportal
```

The default value for the deployment subscriber is "false":
`deployment.subscriberOnly=false`

You can override this value for the "yellowportal" virtual portal and change it to "true" by using this configuration parameter:
`deployment.subscriberOnly.vp.yellow=true`

If the base configuration for a setting is different from all the virtual portals, it is more efficient to use a base override setting. To do this, add this setting:
`enable.base.portal.overrides=true`

To add a setting just for the base portal, add "`.base`" to the end of the parameter name.

For example, if you use the default setting `deployment.subscriberOnly=true`, you can add another setting that is named `deployment.subscriberOnly.base=false` that is applied to the base portal only. All the virtual portals use the default value of `deployment.subscriberOnly=true`.

Disabling Workflow Actions

Disable workflow action on servers that do not require workflows to be processed, such as a subscriber. This strategy can improve performance and reduce the number of versions that are created for each item.

Procedure

1. Log in to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers** > **WCM WCMConfigService** > **Custom properties**.

   Cluster note: If you are using this web content server as part of a cluster, ensure that you use the WebSphere Integrated Solutions Console for the deployment manager when you edit configuration properties.
3. Add a configuration setting named `disableWorkflowAction`.
   a. Use a comma to separate the workflow action names that you want to disable. For example:
      `disableWorkflowAction=ScheduledMoveAction1,ScheduledMoveAction2`

      Note: Workflow action names are case-sensitive.
   b. To disable all workflow actions on the server, use this setting:
      `disableWorkflowAction=*`
4. Restart the server or cluster.

Enabling connect tags

Enable connect tags to reference web content components and apply customized caching to the components.

Procedure

1. Log in to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers** > **WCM WCMConfigService** > **Custom properties**.

   Cluster note: If you are using this web content server as part of a cluster, ensure that you use the WebSphere Integrated Solutions Console for the deployment manager when you edit configuration properties.
3. Specify `connect.businesslogic` properties to process connect tags from any host or from specific hosts.

   Process connect tags from any host

   Add the following property:

   - Property name: `connect.businesslogic.processunknownhosts`
   - Value: `true`

   Process connect tags from specific hosts

   Add the following property:

   - Property name: `connect.businesslogic.processunknownhosts`
   - Value: `false`

   For each host for which you want to enable processing, add a property:

   - Property name: `connect.businesslogic.hosts.`*`hostname`*
   - Value: `true`
4. Restart the server or cluster.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Enabling email

To use the email workflow action, you must configure Web Content Manager to use your SMTP server.

Procedure

1. Log in to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers** > **WCM WCMConfigService** > **Custom properties**.

   Cluster note: If you are using this web content server as part of a cluster, ensure that you use the WebSphere Integrated Solutions Console for the deployment manager when you edit configuration properties.
3. Specify `connect.connector.mailconnect` properties to use your SMTP server. Add the following properties:

   Default SMTP server

   - Property name: `connect.connector.mailconnector.defaultsmtpserver`

- Value: *mail.yourmailserver.com*

Default SMTP port

- Property name: `connect.connector.mailconnector.defaultsmtpport`
- Value: *yourport*

Default email address for "from" field

- Property name: `connect.connector.mailconnector.defaultfromaddress`
- Value: *admin@yourmailserver.com*

Default email address for "reply-to" field

- Property name: `connect.connector.mailconnector.defaultreplytoaddress`
- Value: *admin@yourmailserver.com*

4. If you use a secured SMTP server, you need to specify a user name and password to access the SMTP server: Add the following properties:

   Default user name

   - Property name: `connect.connector.mailconnector.defaultusername`
   - Value: *username*

   Default password

   - Property name: `connect.connector.mailconnector.defaultpassword`
   - Value: *password*

5. Save your changes.
6. Restart the portal for the new settings to take effect.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Defining alternative administrators for multi-realm configurations

Web Content Manager requires a user to run various system tasks such as initialization, and background tasks such as syndication. By default, this system user is the configured JCR domain administrator. In scenarios where the configured user realm does not contain the domain administrator then an alternative user must be provided.

Procedure

1. Log in to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers** > **WCM WCMConfigService** > **Custom properties**.

   Cluster note: If you are using this web content server as part of a cluster, ensure that you use the WebSphere Integrated Solutions Console for the deployment manager when you edit configuration properties.
3. The settings are different for the base portal and any virtual portals.

   The base portal

   - Property name: `wcm.admin.user.dn`
   - Value: The distinguished name of the alternative admin.

For example: `wcm.admin.user.dn=uid=myadmin,o=myRealm`

You can use a base override setting for this property. See "Setting scoped configuration settings for virtual portals" on page 433 for further information about setting a base override setting for your base portal.

Virtual portals
Use scoped configuration settings to apply different settings for specific virtual portals. See "Setting scoped configuration settings for virtual portals" on page 433 for further information.

4. Save your changes.
5. Restart the portal for the new settings to take effect.

Configuring managed pages

When you create a new installation of IBM WebSphere Portal Express 8.5, managed pages are enabled by default. However, you can also manually disable and enable the feature as needed.

About this task

Migration note: If you migrate from a previous version, managed pages are disabled by default, but you can enable the feature after migration.

"Enabling managed pages" on page 376
By default, support for managed pages is enabled for the default virtual portal. However, you can also manually enable this support if managed pages are disabled.

"Disabling managed pages" on page 379
Disable support for managed pages by running the `disable-managed-pages` configuration task.

"Transferring content associations to the Portal Site library" on page 380
When you enable manage pages, any web content pages that you have are converted to managed pages and added to the Portal Site library. However, the content that is associated with the web content pages remains in the original libraries. You can transfer this associated content to the Portal Site library with the `internalize-content-mappings` task.

"Enabling managed pages" on page 376
By default, support for managed pages is enabled for the default virtual portal. However, you can also manually enable this support if managed pages are disabled.

"Disabling managed pages" on page 379
Disable support for managed pages by running the `disable-managed-pages` configuration task.

"Transferring content associations to the Portal Site library" on page 380
When you enable manage pages, any web content pages that you have are converted to managed pages and added to the Portal Site library. However, the content that is associated with the web content pages remains in the original libraries. You can transfer this associated content to the Portal Site library with the `internalize-content-mappings` task.

Related concepts:

"Vanity URLs" on page 2094
You can associate vanity URLs with portal pages and labels. Vanity URLs are short URLs that people can easily remember. They are shorter than full WebSphere Portal Express URLs. They are sometimes also called marketing URLs. You can publish vanity URLs for marketing campaigns through different channels, such as email or print. This way, you can use vanity URLs to direct customers to a specific portal page or content item. Interested site visitors who want to view your campaign can then remember or copy the short vanity URL and type it into the browser address field.

Enabling managed pages

By default, support for managed pages is enabled for the default virtual portal. However, you can also manually enable this support if managed pages are disabled.

About this task

Important: Do not attempt to enable managed pages on a server where managed pages are already enabled. If you previously disabled managed pages and want to re-enable the feature, you must ensure that the Portal Site library is empty first. If you fail to remove the page artifacts from the previous configuration, the resulting portal might not work properly.

When supported, managed pages is enabled for a virtual portal, all pages in the virtual portal are copied into the Portal Site library in IBM Web Content Manager. However, the following pages are not treated as managed pages and are not copied:

- Administration pages, as identified by the label `ibm.portal.Administration` and its child pages
- Private pages

Each virtual portal has its own Portal Site library.

Note: To take advantage of the features available to managed pages in the user interface, your pages must use the Portal 8.5 theme.

Cluster consideration: In a cluster, you need to apply this procedure only to the primary node.

Attention: If you lost your JCR workspace and the backup was create before you created the virtual portal, restoring the workspace creates an inconsistency between the database domains. The **create-virtual-portal-site-nodes** task fails because the JCR workspace is missing. Run the **action-migrate-vps** task to correct the inconsistency with the database domains and to restore the JCR workspace.

Procedure

1. Start the portal server.
2. To enable support for managed pages, run the **enable-managed-pages** task from the *wp_profile_root*`/ConfigEngine` directory.

   Windows

   ```
   ConfigEngine.bat enable-managed-pages -DPortalAdminPwd=password
   -DWasPassword=password
   ```

   Linux

   ```
   ./ConfigEngine.sh enable-managed-pages -DPortalAdminPwd=password
   -DWasPassword=password
   ```

IBM i `ConfigEngine.sh enable-managed-pages -DPortalAdminPwd=`*`password`*
`-DWasPassword=`*`password`*

After you run the **enable-managed-pages** task for the first time, the property **managed.pages** is created in the portal WP Configuration Service. The value of the property is set to `true`.

3. Restart the portal server.
4. To populate web content libraries with information about virtual portals in the system, run the **create-virtual-portal-site-nodes** task from the *wp_profile_root*`/ConfigEngine` directory. For each virtual portal, this task creates a library and a site area that is called `lost-found` for resources that cannot be properly located. If the library or site area exist, the task exits. By default, the task runs on all virtual portals in the system.

   Windows
   `ConfigEngine.bat create-virtual-portal-site-nodes`
   `-DPortalAdminPwd=`*`password`* `-DWasPassword=`*`password`*

   Linux `./ConfigEngine.sh create-virtual-portal-site-nodes`
   `-DPortalAdminPwd=`*`password`* `-DWasPassword=`*`password`*

   IBM i `ConfigEngine.sh create-virtual-portal-site-nodes`
   `-DPortalAdminPwd=`*`password`* `-DWasPassword=`*`password`*
5. To populate web content libraries with information about the portal pages in the system, run the **create-page-nodes** task from the *wp_profile_root*`/ConfigEngine` directory.

   This task can also be used when portal pages and managed pages artifacts in Web Content Manager are not synchronized. In this case, the task attempts to resynchronize the portal artifacts and web content artifacts, giving precedence to the portal artifacts.

   Performance note: Depending on the amount of information in the system, the **create-page-nodes** task can take a long time to run. Because of the database load of the task, do not run the task frequently. The initial run of the task requires the most time, while subsequent runs typically require less time.

   Note: If you have many pages, then it might be necessary to increase the soap client timeout. Edit the *wp_profile_root*`/properties/soap.client.props` file to change the **com.ibm.SOAP.requestTimeout** to `60000`.
   Attention: If your virtual portals have different administrative accounts, you cannot run the **create-page-nodes** task directly. You must run the task for every virtual portal, including the base virtual portal. Use the **VirtualPortalHost** or **VirtualPortalContext** parameter with the **create-page-nodes** task. Run the **list-all-virtual-portals** task to get a list of all your virtual portals. When you run the create-page-nodes task on the base virtual portal, set the **VirtualPortalContext** value to `__NO__VP__ID__`.

   Windows
   `ConfigEngine.bat create-page-nodes -DPortalAdminPwd=`*`password`*
   `-DWasPassword=`*`password`*

   Linux `./ConfigEngine.sh create-page-nodes -DPortalAdminPwd=`*`password`*
   `-DWasPassword=`*`password`*

   IBM i `ConfigEngine.sh create-page-nodes -DPortalAdminPwd=`*`password`*
   `-DWasPassword=`*`password`*

By default, this task is run on all pages in all virtual portals. To limit this task to a specific virtual portal, identify the virtual portal by adding one of the following parameters to the command line. Each parameter requires the prefix `-D` on the command line.

VirtualPortalHost
Specify the host name of the virtual portal. For example, `vp.example.com`.

Important: If the host name of the virtual portal is the same as the host name of the default virtual portal, you must also specify the **`VirtualPortalContext`** property. You can specify the **`VirtualPortalHost`** property by itself only if the host name is unique.

VirtualPortalContext
Specify the virtual portal context that identifies the virtual portal. For example, `vp1`.

You can customize the task with the following optional parameters on the command line. Each parameter requires the prefix `-D` on the command line.

`RunParallel`
Indicate whether you want the task to run with multiple threads. A value of `false` indicates a single thread and is the default setting.

A value of `true` indicates multiple threads, as specified by the work manager **wpsJcrSyncWorkManager** in the WebSphere Integrated Solutions Console. Each thread requires a database connection. For optimal performance, ensure that your database connection pool supports at least as many connections as there are threads in the pool.

`Excluded`
Specify a list of unique names of page nodes to exclude from the creation process. Excluding a page also excludes its child pages. By default, the portal administration pages (`ibm.portal.Administration`) are excluded.

6. Optional: If you used web content pages before you enabled managed pages, you can transfer the content for those pages to the Portal Site library. If you plan to use the default page templates and store your web content in the Portal Site library, transfer the content for the template pages to the Portal Site library. For more information, go to *Transferring content associations to the Portal Site library*.
7. Optional: Ensure that users have appropriate access to the Web Content Manager REST virtual resource so they can use **Edit mode**. For example, they have `user` access.

Related concepts:

"Page templates" on page 1786
Content authors use the page templates to quickly create pages that are consistent with your site design. They do not have to waste time to configure settings that are probably consistent across your site, such as theme selection, page layout, and more.

Related tasks:

"Transferring content associations to the Portal Site library" on page 380
When you enable manage pages, any web content pages that you have are converted to managed pages and added to the Portal Site library. However, the content that is associated with the web content pages remains in the original libraries. You can transfer this associated content to the Portal Site library with the `internalize-content-mappings` task.

Disabling managed pages

Disable support for managed pages by running the `disable-managed-pages` configuration task.

About this task

Disabling managed pages has the following effects:

- By default, each virtual portal has its own specific workspace where content is stored. When you disable managed pages, only a single workspace for the default virtual portal is available. The workspaces of other virtual portals are still there, but you can no longer access them. Any system associations between pages in those virtual portals and their respective Portal Site libraries no longer work.

  Important: To preserve content in the other virtual portals, you must import or syndicate the libraries into the default virtual portal before disabling managed pages.
- You can still access the Portal Site library for the default virtual portal, but the library is no longer automatically synchronized with the page structure.
- Pages are no longer managed in IBM Web Content Manager, with the following implications:
  - No page drafts can be created.
  - No new versions of pages can be created.
  - Pages are no longer syndicated.
  - Access control changes that you perform in the portal interface are no longer applied to the portal page site area.
  - If you delete a page from the portal interface, the corresponding portal page site area is not deleted.
- If you create a page with either the **Basic** or **Articles** page template, the page has no web content association. This missing association can cause errors if you attempt to add content from the **Create Content** tab of the site toolbar. To use the sample web content items when managed pages are disabled, create a web content association on the page before attempting to add content.
- Having managed pages enabled is a mandatory requirement to have vanity URLs enabled. Therefore, if you have vanity URLs enabled and disable managed pages, vanity URLs are also disabled.

Procedure

1. Run the **disable-managed-pages** task from the *wp_profile_root*/ConfigEngine directory.

   Linux `./ConfigEngine.sh disable-managed-pages -DPortalAdminPwd=`*password* `-DWasPassword=`*password*

   IBM i `ConfigEngine.sh disable-managed-pages -DPortalAdminPwd=`*password* `-DWasPassword=`*password*

   Windows
   `ConfigEngine.bat disable-managed-pages -DPortalAdminPwd=`*password* `-DWasPassword=`*password*

   After running the **disable-managed-pages** task for the first time, the property **managed.pages** is created in the **WP WPConfigService** configuration service. The value of the property is set to `false`.
2. Restart the portal server.

Transferring content associations to the Portal Site library

When you enable manage pages, any web content pages that you have are converted to managed pages and added to the Portal Site library. However, the content that is associated with the web content pages remains in the original libraries. You can transfer this associated content to the Portal Site library with the `internalize-content-mappings` task.

About this task

Note: Administration pages are not intended to be managed pages and so are not included when you enable managed pages.

When you transfer the content association for a page to the Portal Site library, several things happen:

- The content that is referenced by the default content association for the page is copied to the portal page site area for the page. Only the default content association is affected; other content associations for the page are ignored.

  Note: Nested pages are not copied. Nested site areas are not copied in the following cases:
  - The nested site area is referenced by the default association of another page.
  - The nested site area has the same name as an existing site area for the same page.
- Template mappings and content elements that exist in the associated site area are copied over into the portal page. If the template mapping or element exists for the page, the copy is not performed.
- The default content setting for the portal page is modified to reference the copied content.
- The configuration of any web content viewers on the page is updated to reference the content that is stored in the portal page site area. However, viewer configurations that use content paths are not affected.

Procedure

To transfer content associations, run the `internalize-content-mappings` task from the *wp_profile_root*`/ConfigEngine` directory.

Windows

```
ConfigEngine.bat internalize-content-mappings
-DPortalPage=target_page -DIncludeDescendants=true_or_false
-DSynchronous=true_or_false -DPortalAdminPwd=password
-DWasPassword=password
```

Linux

```
./ConfigEngine.sh internalize-content-mappings
-DPortalPage=target_page -DIncludeDescendants=true_or_false
-DSynchronous=true_or_false -DPortalAdminPwd=password
-DWasPassword=password
```

IBM i

```
ConfigEngine.sh internalize-content-mappings
-DPortalPage=target_page -DIncludeDescendants=true_or_false
-DSynchronous=true_or_false -DPortalAdminPwd=password
-DWasPassword=password
```

The following properties must be specified either on the command line or in the `wkplc.properties` file.

PortalPage
The object ID or the unique page name of the page for which you want to transfer content. If the target page is contained in a virtual portal, you must identify the virtual portal by specifying either the **VirtualPortalContext** parameter or **VirtualPortalHost** parameter.

IncludeDescendants
Specify `true` to transfer content for the target page and any child pages. To transfer content only for the target page, specify `false`. If not specified, the default value is `true`.

Synchronous
Specify `true` to perform the transfer synchronously. To perform the transfer asynchronously, specify `false`. If not specified, the default value is `true`.

Verbose
Specify `true` to output additional information to the log. To generate basic log information, specify `false`. If not specified, the default value is `false`.

CollisionHandling
When you copy content to a page, specify the action to take if the content item exists. By default, that content item is not copied. If you set **CollisionHandling** to `replace`, the content item on the page is replaced with the content item to be copied to the page.

VirtualPortalContext
Specify the virtual portal context that identifies the virtual portal. For example, `vp1`.

VirtualPortalHost
Specify the host name of the virtual portal. For example, `vp.example.com`.

Important: If the host name of the virtual portal is the same as the host name of the default virtual portal, you must also specify the **VirtualPortalContext** property. You can specify the **VirtualPortalHost** property by itself only if the host name is unique.

PortalAdminPwd
The administrator password for WebSphere Portal Express.

WasPassword
The administrator password for WebSphere Application Server.

Example commands:

- Windows: `ConfigEngine.bat internalize-content-mappings -DPortalPage=example.page -DIncludeDescendants=true -DSynchronous=true -DPortalAdminPwd=password -DWasPassword=password`
- Linux : `./ConfigEngine.sh internalize-content-mappings -DPortalPage=example.page -DIncludeDescendants=true -DSynchronous=true -DPortalAdminPwd=password -DWasPassword=password`
- IBM i: `ConfigEngine.sh internalize-content-mappings -DPortalPage=example.page -DIncludeDescendants=true -DSynchronous=true -DPortalAdminPwd=password -DWasPassword=password`

Managing tagging and rating for web content

When you use tagging and rating with web content, the web content viewer provides extra scope options for the filtering of tagging and rating results. Because changes in the web content system can affect the accuracy of the tagging and

rating information that is used by the portal, it is important to keep the scope information up to date by synchronizing the scopes regularly.

"Applying tagging and rating scopes to web content"
Scoping is typically used to filter the tag cloud or ratings overview according to hierarchical metadata attached to the resources that are being tagged. When you apply tagging and rating to web content, you can scope these display components according to authoring template, category, or content item parent.

"Synchronizing scopes for web content"
When users are tagging or rating web content, the web content viewer provides the tagging or rating information to the portal, where it is stored. If information in the web content system changes, this change can cause the tagging and rating information that is stored in the portal to be out of sync. This issue can happen, for example, if content items are moved or category information changes. To ensure that the tagging and rating information is current, synchronize the scopes that are used for web content. You can set up automatic synchronization according to different conditions or run a manual synchronization as needed.

Applying tagging and rating scopes to web content

Scoping is typically used to filter the tag cloud or ratings overview according to hierarchical metadata attached to the resources that are being tagged. When you apply tagging and rating to web content, you can scope these display components according to authoring template, category, or content item parent.

About this task

You can configure the advanced settings of the web content viewer to limit results to show only tags or ratings that are associated with one or more of the following scopes:

- The parent of the content item that is being displayed (for example, a site area).
- The authoring template that is used to generate the content item or site area that is being displayed.
- The categories that are used to profile the content item that is being displayed. In this way, you can manage scopes from within your web content system by defining taxonomies for your content items.

Related tasks:

"Synchronizing scopes for web content"
When users are tagging or rating web content, the web content viewer provides the tagging or rating information to the portal, where it is stored. If information in the web content system changes, this change can cause the tagging and rating information that is stored in the portal to be out of sync. This issue can happen, for example, if content items are moved or category information changes. To ensure that the tagging and rating information is current, synchronize the scopes that are used for web content. You can set up automatic synchronization according to different conditions or run a manual synchronization as needed.

Synchronizing scopes for web content

When users are tagging or rating web content, the web content viewer provides the tagging or rating information to the portal, where it is stored. If information in the web content system changes, this change can cause the tagging and rating information that is stored in the portal to be out of sync. This issue can happen, for example, if content items are moved or category information changes. To ensure that the tagging and rating information is current, synchronize the scopes that are

used for web content. You can set up automatic synchronization according to different conditions or run a manual synchronization as needed.

"Synchronizing scopes when items change"
To automatically run scope synchronization whenever an item changes in the web content system, specify the `tagging.syndication.enableItemModificationSynchronization` property in the Web Content Manager configuration service.

"Synchronizing scopes after syndication" on page 446
To automatically run scope synchronization whenever syndication occurs, specify the `tagging.syndication.enableTagSynchronization` property in the Web Content Manager configuration service.

"Scheduling scope synchronization" on page 446
You can schedule scope synchronization to be run at specific times by defining the schedule with the XML configuration interface.

"Synchronizing scopes manually" on page 448
If automatic synchronization is not enabled for the scopes that are used for web content, or if you want to run synchronization outside of a scheduled synchronization period, you can manually start the synchronization process.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Synchronizing scopes when items change:

To automatically run scope synchronization whenever an item changes in the web content system, specify the `tagging.syndication.enableItemModificationSynchronization` property in the Web Content Manager configuration service.

About this task

Note: This type of synchronization works only for individual item changes. For example, this type of synchronization is not automatically run when an entire site area or folder is moved. To synchronize scopes after such a change, you can run synchronization manually.

Procedure

1. Log in to the WebSphere Integrated Solutions Console (`http://`*`hostname.example.com`*`:10027/ibm/console`).
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Click **WCM WCMConfigService**.
4. Under **Additional Properties**, click **Custom Properties**.
5. Add the `tagging.syndication.enableItemModificationSynchronization` property.
   a. Click **New**, and enter the property name `tagging.syndication.enableItemModificationSynchronization`.
   b. Set the string value to `true`.
6. Click **OK**, and save the changes to the master configuration.
7. Restart the portal.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Synchronizing scopes after syndication:

To automatically run scope synchronization whenever syndication occurs, specify the `tagging.syndication.enableTagSynchronization` property in the Web Content Manager configuration service.

Procedure

1. Log in to the WebSphere Integrated Solutions Console (`http://`*`hostname.example.com`*`:10027/ibm/console`).
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Click **WCM WCMConfigService**.
4. Under **Additional Properties**, click **Custom Properties**.
5. Add the `tagging.syndication.enableTagSynchronization` property.
   a. Click **New**, and enter the property name `tagging.syndication.enableTagSynchronization`.
   b. Set the string value to `true`.
6. Click **OK**, and save the changes to the master configuration.
7. Restart the portal.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Scheduling scope synchronization:

You can schedule scope synchronization to be run at specific times by defining the schedule with the XML configuration interface.

Procedure

1. Verify whether any scheduled synchronizations are defined for the portal.
   a. Create an export file that you can use with the **xmlaccess** command. Here is an example of a request you can use to query the current configuration:

```
<?xml version="1.0" encoding="UTF-8"?>
<request type="export" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd" >
  <portal action="locate">
    <task action="export" name="com.ibm.portal.cp.SynchronizationTask"/>
  </portal>
</request>
```

   b. Run the **xmlaccess** command, specifying the export file. The resulting output file contains any scheduled synchronization times that are defined in the portal.
2. Set the synchronization schedule.

a. To set a time for a scheduled synchronization, create an XML request document. For example, to schedule a synchronization to occur at 15:36 hours every day, you would use a request like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<request type="update" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd">
  <portal action="locate">
    <task action="create" name="com.ibm.portal.cp.SynchronizationTask">
      <startTime>15:36</startTime>
    </task>
  </portal>
</request>
```

For each scheduled synchronization, create a separate `task` element and specify the time with a `startTime` element.

b. Run the **xmlaccess** command, specifying the file that contains the scheduling request. Scope information for the web content system is then synchronized automatically according to the schedule you defined.

3. Optional: If you want to set a minimum time before subsequent synchronizations are run, specify the `tagging.syndication.minimumTagSynchronizationTimeInterval` property in the Web Content Manager configuration service.
   a. Log in to the WebSphere Integrated Solutions Console (`http://`*`hostname.example.com`*`:10027/ibm/console`).
   b. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
   c. Click **WP ConfigService**.
   d. Under **Additional Properties**, click **Custom Properties**.
   e. Click **New**, and enter the property name `tagging.syndication.minimumTagSynchronizationTimeInterval`.
   f. Set the string value to the number of seconds between synchronizations.
   g. Click **OK**, and save the changes to the master configuration.
   h. Restart the portal.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Related reference:

"XML configuration reference" on page 1082
Learn more about XML input structure, XML tags for portal resources, attributes for special purposes such as locale data, object IDs, and more. Also find information about action attributes that determine the type of processing that the XML configuration reference applies to the portal resource. There are syntax restrictions that you need to consider as well as information about how to determine the object IDs for portal resources.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Synchronizing scopes manually:

If automatic synchronization is not enabled for the scopes that are used for web content, or if you want to run synchronization outside of a scheduled synchronization period, you can manually start the synchronization process.

Procedure

To manually run synchronization, run the `cp-sync` configuration task or submit an XML request to the portal by using the XML configuration interface.

- To run synchronization with a configuration task, run the following task from the *wp_profile_root*`/ConfigEngine` directory:
  - Windows: `ConfigEngine.bat cp-sync -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`*
  - Linux: `./ConfigEngine.sh cp-sync -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`*
  - IBM i: `ConfigEngine.sh cp-sync -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`*
- Create an XML request file and submit it using the **`xmlaccess`** command. Here is an example of a request you can use to start synchronization:

```
<?xml version="1.0" encoding="UTF-8"?>
<request type="update" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd">
  <portal action="locate">
    <task action="create" name="com.ibm.portal.cp.SynchronizationTask"/>
  </portal>
</request>
```

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Syndication

Use syndication to replicate web content library data from one server to another server. Syndication is based on a syndicator and subscriber relationship. The syndicator has the current data. The subscriber received the current data from the syndicator.

"Syndication relationships" on page 449
Syndication is the method that is used by IBM Web Content Manager to replicate data from a web content library on one server to a web content library on another server.

"Syndication properties" on page 454
You can tailor the syndication behavior of your web content environment by changing configuration settings such as the syndication interval and automatic syndication.

"Syndication tuning" on page 456
While syndication is a vital part of keeping your web content current, the same syndication strategy is not appropriate for every environment. Depending on how you deploy IBM Web Content Manager, you can use different syndication strategies to balance the currency of your content with the performance your environment requires.

"Syndication troubleshooting" on page 459
If you encounter issues when syndicating, there are some common methods available to troubleshoot these issues.

"Creating a syndication relationship from the command line" on page 462
You can set up syndication relationships by using the Administration Portlets or the command line. To set up a syndication relationship from the command line, use the XML configuration interface (XML access) and the ConfigEngine command to configure the subscriber.

Related information:

Creating a syndication relationship by using the Administration Portlets

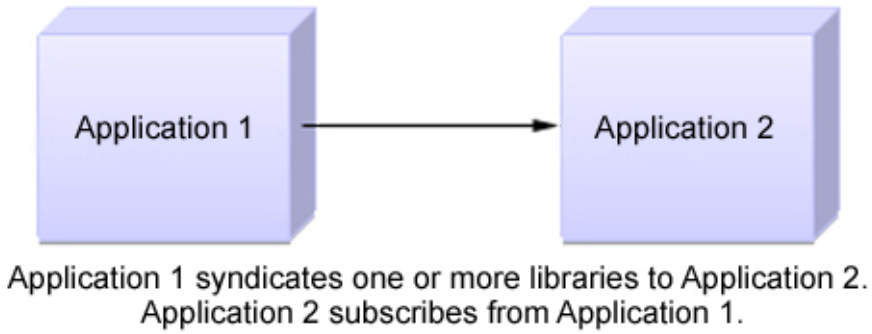
Syndication relationships

Syndication is the method that is used by IBM Web Content Manager to replicate data from a web content library on one server to a web content library on another server.

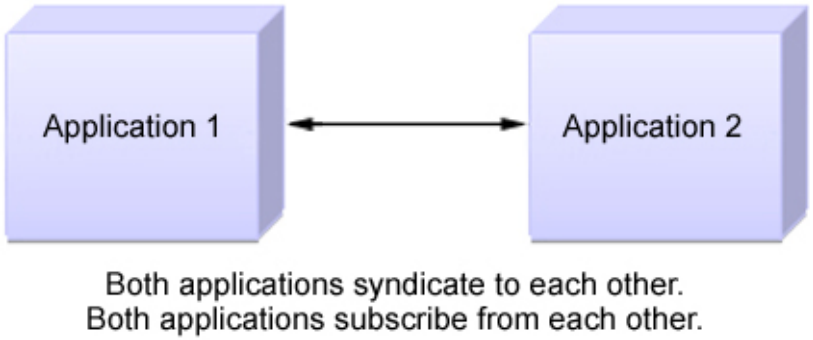
The relationship between a syndicator and a subscriber can be either a one-way or two-way relationship.

One-way syndication

Application 1 syndicates one or more libraries to Application 2, and Application 2 subscribes from Application 1.



Two-way syndication

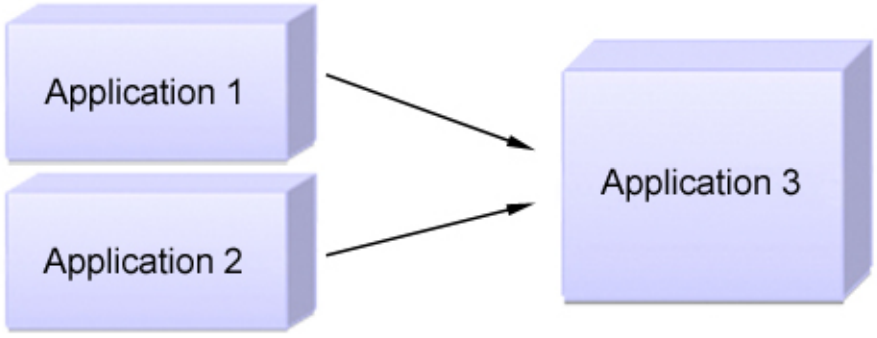


Note:

- When two-way syndication is enabled, you must first establish the syndication relationship from Application 1 to Application 2. After the libraries are replicated to Application 2, you can set up the syndication relationship between Application 2 and Application 1.
- When two-way syndication is used, all-item syndication must be enabled for each library that exists on both servers.
- Although it is possible to set up more than one syndication relationship between the same two applications, there is no reason to do so. The additional syndication relationships are not required because when a syndication relationship is established between two applications, no further relationships are established.

Multiple syndication relationships

Syndicators can syndicate libraries to multiple subscribers, and subscribers can subscribe to libraries from multiple syndicators.



Both Application 1 and 2 syndicate to Application 3.
Application 3 subscribes from both Application 1 and 2.

Syndication methods

There are three syndication methods available when a syndication relationship is configured:

Live items:

Live item syndication is mostly used when you syndicate to a staging or delivery server. The following items are syndicated:

- Published
- Expired

Draft items, projects, and items in a project are not syndicated.

Live and projects:

The advantage of using "Live and projects" syndication is to gradually syndicate projects to a staging or delivery server rather that waiting to syndicate all the items in a project after they all achieve a published state. The following items are syndicated:

- Published
- Expired

- Projects
- Draft items in a project

Draft items outside of projects are not syndicated.

All items:
All item syndication is mostly used when you syndicate between servers within an authoring environment. The following items are syndicated:

- Published
- Expired
- Projects
- Draft items in a project
- Other draft items
- Versions
- Deleted items

Switching from "all item" syndication to "live item" syndication: When you switch from "all item" syndication to "live and projects" syndication or "live item" syndication, any drafts previously syndicated to the subscriber are not removed.

Moving draft items between libraries: If you move a draft item from a library that uses "all item" syndication to a library that uses "live item" syndication, the draft item is also moved on the subscriber because the action occurred on the library that uses "all item" syndication. This behavior allows for some draft items to be included in a subscriber library even though "live item" syndication is being used.

CF05
Syndication modes

Each syndication relationship can be configured with a different syndication mode. This determines how syndication is scheduled.

CF05 **mode="*modetype*"**

- **Configured**: This uses the mode that is configured in the WCM WCMConfigService service.
- **Automatic**: Syndication is scheduled automatically based on the configured syndication interval set on the syndicator.
- **Manual**: Syndication occurs only when requested by using the administration portlet.

CF07
Manual syndication types

You can manually syndicate by using the following methods:

Update
This method syndicates items that are newer than the previous syndication. Items that are newer than the last syndication on the syndicator are sent to the subscriber. Items that are newer on the subscriber are not updated. Items that are created on the subscriber that do not exist on the syndicator are not removed from the subscriber.

Rebuild
This method syndicates all items that are newer on the syndicator. All items that are newer on the syndicator are sent to the subscriber. Items that

are newer on the subscriber are not updated. Items that are created on the subscriber that do not exist on the syndicator are not removed from the subscriber.

Rebuild with mirror

If you select the mirror option, all items on the subscriber are reset to mirror the syndicator. All items that are newer on the syndicator are sent to the subscriber. Items that are newer on the subscriber are overwritten with the older version from the syndicator. Items that are created on the subscriber that do not exist on the syndicator are removed from the subscriber. Version history is not syndicated.

Restriction: The **Rebuild with mirror** option can only be used when syndicating between servers that use CF07 or higher.

Restriction: The **Rebuild with mirror** option must not be used with two-way syndication.

Restriction: The **Rebuild with mirror** option can be used only on a syndicator.

Note: The **Rebuild with mirror** option will not automatically cascade through all subscribers downstream of the syndicator. You might need to repeat the **Rebuild with mirror** action on subscribers further down the syndication chain to synchronize all servers.

Cross version syndication

Cross-version syndication is supported between the following releases.

- WebSphere Portal version 7.0.0.2 with CF26 or higher.
- WebSphere Portal 8.0.0.1 with CF09 or higher.
- WebSphere Portal 8.5 or higher.

Syndicating from a newer software level to an older software level is only supported between different fix-pack levels of the same release. When syndicating between releases, only older to newer is supported.

See “Cross version syndication” on page 806 for further information.

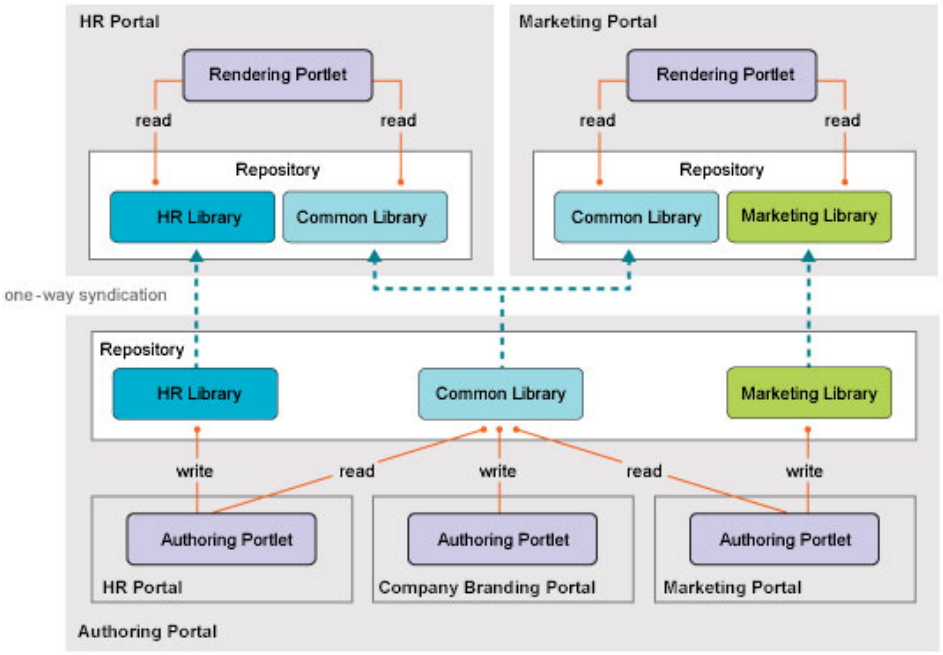
Web content libraries and syndication relationships

All the items that you work with as part of your Web Content Manager authoring environment are stored in web content libraries. When you syndicate data between applications, you do so on a library by library basis. As part of the definition of a syndicator or subscriber, you specify which web content libraries are to be included during syndication.

Because syndication is run on a library by library basis, it is important to consider how to organize your content between libraries to support your Web Content Manager environment. For example, suppose that you are using a single authoring server to develop content for two delivery servers, an intranet site that provides Human Resources information that is intended for internal employees of a company and an external Internet site that provides marketing material that is intended for customers and others outside the company. A basic approach to

support this environment would be to use two web content libraries, one for content specific to each site. You would then set up two syndication relationships with each going from the authoring server to the appropriate delivery server.

For easier management, you might divide your content further into three libraries, where one library contains data common to both the intranet and Internet sites and the other two libraries contain site-specific content. The following example demonstrates this configuration, with the addition of two other authoring portlets so that the content of each library is maintained by a different authoring portlet.



In this case you might set up several syndication relationships between the authoring server and the delivery servers:

- The Common Library syndicates to the intranet site (Human Resources Portal).
- The Common Library syndicates to the Internet site (Marketing Portal).
- The HR Library syndicates to the intranet site (Human Resources Portal).
- The Marketing Library syndicates to the Internet site (Marketing Portal).

Note: Web Content Manager provides flexibility in how you set up your syndication relationships. If you need to syndicate multiple libraries from one server to another, you can choose to use one syndication relationship that includes all the libraries, or you can choose to use separate syndication relationships for each library, or even a combination of both approaches, depending on how many libraries you are syndicating. The best approach for your situation depends not only on how many libraries are involved but also on how the libraries are related to one another. For example, you use a single syndication relationship for libraries that reference each other, as when one library contains design items like templates that are used by content in the other library. However, if the libraries are independent of one another and you think you might want to suspend syndication of one library but not the other, separate syndication relationships for each library can provide that.

Important:

- First-time syndication to an existing library is not supported. If you attempt to syndicate a library to a subscriber that already has a library with the same name, an error results.
- Some information about a Library is only syndicated the first time syndication occurs and not on subsequent updates and rebuilds. If you change the user access to a library, you must manually make the same changes to any subscriber libraries if you want the same settings on all your syndicated libraries.
- If content from one library (Library A) uses an item from another library (Library B), you must include both libraries in the syndicator to ensure that all items are syndicated successfully. If you include only Library A in the syndicator, any items in Library A that reference items in Library B are not syndicated, and syndication errors are generated.
- If you add a library to a syndicator after the initial syndication, you click `Update` to force the new library to be syndicated immediately.

Access control and syndication

Although syndication can be used to keep data current between libraries on different servers, access control settings for the libraries are not included as part of syndication. Depending on how your environment is set up and what policies you have in place for library access, there are extra considerations for access control when syndicating.

User consistency
: For user level access to remain consistent between the syndicator and subscriber, both servers must be configured to use the same user repository. If different user repositories are used, syndication occurs but there are errors in the subscriber log indicating missing users. If access controls are determined by using only virtual users and groups, such as "All authenticated" and "Anonymous Users", then there is no need to use the same user repository on the syndicator and subscriber.

First time syndication on a new library
: Because library access control settings are not syndicated, you must manually set access permissions on the subscriber's library when you syndicate for the first time. If the library does not exist on the subscriber, it is created during syndication. By default, no access control settings are specified on the new library, so you must set them manually before users can access content in the new library. The settings on the subscriber library do not have to match those on the syndicator library. This allows you to specify different levels of access for users and groups on the subscriber.

Syndication properties

You can tailor the syndication behavior of your web content environment by changing configuration settings such as the syndication interval and automatic syndication.

You define and manage syndication options in the **WCM WCMConfigService** service by using the WebSphere Integrated Solutions Console.

Changing the syndication interval

Although the frequency of syndication is set by default during installation, you can change the syndication interval to better suit the needs of your environment.

For example, you might shorten the interval in an active authoring environment where users must collaborate heavily and rely on timely replication. Similarly, you might lengthen the interval to avoid excessive replication of data that does not change often.

Note: The syndication interval applies to all syndication operations and cannot be specified separately for different syndicator-subscriber pairs.

To change the syndication interval, modify the **deployment.itemChangedTaskDelay** property. By default, the syndication interval is set to 30 seconds. Specify the number of seconds to use as the syndication interval, with a minimum of `0` seconds and a maximum of `65536` seconds. A value of `0` prevents syndication from occurring. If you set the value to so short an interval that syndication cannot complete before the interval expires, syndication begins again when the previous syndication completes.

Syndication scheduling mode

Syndication can be set to either happen automatically, or manually:

Automatic
: Syndication is scheduled automatically based on the configured syndication interval.

Manual
: Syndication occurs only when requested by using the administration portlet.

Automatic syndication is enabled by default. To disable automatic syndication, specify the following property:
connect.moduleconfig.syndication.inittasks=false

Configuring a subscriber-only server

A syndicator server uses several processes to gather and queue content for syndication. These processes can sometimes affect server performance when run. However, a subscriber-only server does not require these processes, so you can improve performance on the subscriber-only server by disabling the processes.

To do this, ensure that `deployment.subscriberOnly` property is set to `true`.

Enabling secure syndication by using SSL

To enable and use SSL for syndication, the following properties must be changed in the `WCM WCMConfigService` to use the "https" protocol and the appropriate port.

- **deployment.itemDispatcherUrl=http://${WCM_HOST}:${WCM_PORT}/ ${WCM_CONTEXT_ROOT}/connect/?MOD=ItemDispatcher**
- **deployment.syndicatorUrl=http://${WCM_HOST}:${WCM_PORT}/ ${WCM_CONTEXT_ROOT}/connect/?MOD=Synd**
- **deployment.subscriberUrl=http://${WCM_HOST}:${WCM_PORT}/ ${WCM_CONTEXT_ROOT}/connect/?MOD=Subs**

For example, to enable SSL for syndication where the server is configured to provide SSL on port 10080, change **deployment.syndicatorUrl** from `http://${WCM_HOST}:${WCM_PORT}/${WCM_CONTEXT_ROOT}/connect/?MOD=Synd` to `https://${WCM_HOST}:10080/${WCM_CONTEXT_ROOT}/connect/?MOD=Synd`.

If self-signed certificates are used, extra steps maybe necessary to ensure the certificates that are exchanged are trusted on both servers. See the "SSL Configurations" topic in the WebSphere Application Server documentation.

Related tasks:

“Setting service configuration properties” on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Syndication tuning

While syndication is a vital part of keeping your web content current, the same syndication strategy is not appropriate for every environment. Depending on how you deploy IBM Web Content Manager, you can use different syndication strategies to balance the currency of your content with the performance your environment requires.

There are several different means to manage how and when content is replicated to other servers. It is important to note that, as with any process, syndication entails a performance cost. You must account for this when you set not only the frequency of syndication, but also the number of syndication relationships for a server.

Syndication interval

The syndication interval controls the frequency of syndication while automatic syndication is enabled and can be used by administrators to put an upper limit how often syndication occurs. Because up-to-date information is important to any web content environment, automatic syndication is enabled by default when IBM WebSphere Portal Express is installed. While the default setting for the syndication interval ensures maximum currency, you can choose to adjust the value if the currency demands of your environment do not call for the shortest interval.

Here are some general guidelines for setting the syndication interval.

Table 53. Syndication interval

Interval setting	Recommended environments
10 minutes – 2 hours	Staging servers to delivery servers.
30 seconds – 10 minutes	Any environment that requires frequent replication, such as an authoring server to a staging server, a test server, or distributed authoring server. When you increase the syndication interval for environments where authoring servers are involved, be mindful that timely replication is often essential, particularly in collaborative authoring environments where multiple authors on different servers might be working on the same content.

Syndication types

Live items:

Live item syndication is mostly used when you syndicate to a staging or delivery server. The following items are syndicated:

- Published
- Expired

Draft items, projects, and items in a project are not syndicated.

Live and projects:
The advantage of using "Live and projects" syndication is to gradually syndicate projects to a staging or delivery server rather that waiting to syndicate all the items in a project after they all achieve a published state. The following items are syndicated:

- Published
- Expired
- Projects
- Draft items in a project

Draft items outside of projects are not syndicated.

All items:
All item syndication is mostly used when you syndicate between servers within an authoring environment. The following items are syndicated:

- Published
- Expired
- Projects
- Draft items in a project
- Other draft items
- Versions
- Deleted items

Manual syndication

It is not necessary to wait for the syndication interval to elapse to run syndication. Manual syndication can be used at any time to cause syndication to occur immediately. There are different ways to take advantage of manual syndication:

- You can use manual syndication along with the syndication interval to provide flexibility. For example, if you are using an increased syndication interval to optimize performance between servers, you can still run manual syndication when you need to update content without waiting for the next syndication.
- If you require complete control of when syndication occurs, you can use manual syndication only. For this approach, you must disable automatic syndication so that the syndication interval setting is ignored.

CF07 You can manually syndicate by using the following methods:

Update
This method syndicates items that are newer than the previous syndication. Items that are newer than the last syndication on the syndicator are sent to the subscriber. Items that are newer on the subscriber are not updated. Items that are created on the subscriber that do not exist on the syndicator are not removed from the subscriber.

Rebuild
This method syndicates all items that are newer on the syndicator. All items that are newer on the syndicator are sent to the subscriber. Items that

are newer on the subscriber are not updated. Items that are created on the subscriber that do not exist on the syndicator are not removed from the subscriber.

Rebuild with mirror

If you select the mirror option, all items on the subscriber are reset to mirror the syndicator. All items that are newer on the syndicator are sent to the subscriber. Items that are newer on the subscriber are overwritten with the older version from the syndicator. Items that are created on the subscriber that do not exist on the syndicator are removed from the subscriber. Version history is not syndicated.

Restriction: The **Rebuild with mirror** option can only be used when syndicating between servers that use CF07 or higher.

Restriction: The **Rebuild with mirror** option must not be used with two-way syndication.

Restriction: The **Rebuild with mirror** option can be used only on a syndicator.

Note: The **Rebuild with mirror** option will not automatically cascade through all subscribers downstream of the syndicator. You might need to repeat the **Rebuild with mirror** action on subscribers further down the syndication chain to synchronize all servers.

CF05

Syndication modes

Each syndication relationship can be configured with a different syndication mode. This determines how syndication is scheduled.

CF05 **mode="***modetype***"**

- **Configured**: This uses the mode that is configured in the WCM WCMConfigService service.
- **Automatic**: Syndication is scheduled automatically based on the configured syndication interval.
- **Manual**: Syndication occurs only when requested by using the administration portlet.

Publish and expire dates

Because the syndication interval is not based on a scheduled time of the day, it is not suitable for setting up syndication to run during off-peak times. The publish date and expire date are the preferred methods for doing this.

The publish date for each content item specifies the date and time when the content is published to a website. After a content item is approved and the publish date is reached, the content item is queued for syndication according to the next syndication interval. By using the publish date, you can delay the publishing of content, effectively delaying its syndication. For example, if you are using a short syndication interval to ensure timely replication of most content, you can delay the syndication of less urgent content through the publish date for that content.

The expire date specifies the date and time when the content item is removed from a website. As with the publish date, you can use the expire date to cause the syndication activity that is associated with removing the content to occur at a time when other syndication activity is less intensive.

For more information on the publish and expire dates in the workflow process, refer to “Item status” on page 1929.

Syndicating large libraries

When you syndicate large libraries, you might also need to adjust these timeout settings:

1. Log in to the WebSphere Integrated Solutions Console.
2. Click **Servers** > **Server Types** > **WebSphere application servers** > *portal_server* > **Container Services** > **Transaction Service**.
3. Change the **total transaction lifetime timeout** and the **maximum transaction timeout** settings to 2000 seconds.

Syndication troubleshooting

If you encounter issues when syndicating, there are some common methods available to troubleshoot these issues.

Common issues

Table 54. Common issues

Issue	Solution
Unable to reach host	This is a common reason why syndication does not work. The URL for the syndicator or the subscriber might not be valid. You might need to use the IP address rather than the domain name.
Syndicator becomes unresponsive during syndication	Syndication can require a large amount of resources to run successfully. Consequently, if your server is performing other tasks at the same time as syndication, the process of syndication might slow or stop altogether. You must schedule your syndication to occur at times when the server load is at its lowest.
Syndicator status hangs on "Pending", or "Pending, Active"	If you are attempting to update or rebuild a syndicated library that contains large number of items, the syndicator status might hang on "Pending", or "Pending, Active". This can occur because the syndicator keeps retrying to syndicate when some items fail to syndicate to the subscriber, or when a system timeout occurs on the subscriber when saving data. Improving the performance of your database can help avoid these situations. For example, two of the database attributes that DB2 relies upon to perform optimally are the database catalog statistics and the physical organization of the data in the tables. Catalog statistics must be recomputed periodically during the life of the database, particularly after periods of heavy data modifications (inserts, updates, and deletes) such as a population phase. To fix this, you must run "Runstats" on the JCR database before and after syndication. The DB2 runstats command is used to count and record the statistical details about tables, indexes, and columns. See database performance for information on using the "Runstats" task. Due to the heavy load of computing these statistics, it is recommend that this maintenance occurs during off hours, periods of low demand, or when the portal is offline.

Table 54. Common issues (continued)

Issue	Solution
Time-outs during syndication	Time-outs during syndication are often caused by the failure of large items to be saved. Increasing the `total transaction lifetime timeout` setting of your IBM WebSphere Portal Express server can address this issue. The `total transaction lifetime timeout` setting of your subscriber must be at least the same as the syndicator. The `total transaction lifetime timeout` setting is changed by using the WebSphere Integrated Solutions Console. Go to **Servers > Server Types > WebSphere application servers** > ***portal_server*->Container Services** > **Transaction Service**. See the WebSphere Application Server information center for more information.
Subscriber becomes unresponsive during syndication	If you are attempting to syndicate a library that contains more than 10000 items, the subscriber machine might become unresponsive during the syndication operation. This action can occur due to an insufficient Java heap size setting on the subscriber. To update the maximum Java heap size that is used by the portal application server on the subscriber machine, complete the following steps: 1. In the WebSphere Integrated Solutions Console, click **System administration** > **Deployment manager** > **Java and Process Management** > **Process Definition** > **Java Virtual Machine**. 2. Update the value in the **Maximum Heap Size** field. A value of at least 1024 MB is recommended. 3. Click **OK**, and then save your changes. In addition, ensure that you have at least as much swap space allocated on the subscriber machine as you have physical memory.
500 errors on ext2 and ext3 versions of Linux	If you receive 500 errors on ext2 and ext3 versions of Linux, you exceeded the number of children that a parent folder can hold. You cannot store more than 32768 children under one folder on ext2 and ext3 versions of Linux. Move some content items out of the affected site area to another site area. So that none of your site areas contain more than 32768 children under one folder and then try syndicating again. You can move the content items back to the correct site areas after syndication is complete.

Table 54. Common issues (continued)

Issue	Solution
Subscriber is successful but syndicator is pending with no updates or failed items	Verify that the subscriber URL is correct. For example, the subscriber URL might be one of the following URLs: • `http://myportalhostname.ibm.com:port/wps/wcm/connect?MOD=Synd`, where *myportalhostname* is the name of your portal host • `http://mywebserverhostname.ibm.com:port/wps/wcm/connect?MOD=Synd`, where *mywebserverhostname* is the name of your web server host **Note:** In some configurations, TAM automatically appends itself to the subscriber URL, as in the following example: • `http://TAMmyportalhostname.ibm.com:port/wps/wcm/connect?MOD=Synd` 1. Click the **Administration menu** icon. Then, click **Portal Content** > **Syndicators**. 2. Click the **Edit Syndicator** icon for your idle syndicator. 3. Verify that the subscriber URL begins with either your portal host name or your web server host name. If it does not, update the URL so that it matches the examples of the subscriber URLs.

Other solutions

Table 55. Other solutions

Option	Details
Resetting the web content event log.	To assist in the troubleshooting process, you can reset the web content event log. For more information, see *Resetting the web content event log* in the related links.
Enabling the "deployment.enableReport" setting on the subscriber.	You can update the `WCM WCMConfigService` service and specify the `deployment.enableReport` property with a value of `true`. This update enables high level reporting of syndication to the SystemOut log on the subscriber server. It provides a summary of items that were processed, and which items failed syndication to help you troubleshoot syndication issues.

Working with failed items

From time to time items fail to syndicate. You use the failed items view to review a list of failed items and then run syndication again after you fix the issue.

1. Log on to your syndicator as an administrator.
2. Click the **Administration menu** icon. Then, click **Portal Content** > **Syndicators**.
3. The number of failed items for a syndicator are displayed in the Failed Items column. Click the number of failed items to open the Failed Items view.
   - Each failed item for the selected syndicator is displayed in the **Failed Items** view. Information is displayed about each failed item, including information about how the appropriate action required to fix the issue.
   - The Root and Impact columns are used to find the root cause of a syndication failure, and what secondary items are impacted by the root cause. By finding and fixing the root cause of the syndication failure, you also potentially fix the syndication failures of the items that are impacted by the root cause.

- The Important Items tab can also be used to narrow down which items are the most crucial to fix.

4. After you identified and fixed the issues, you can click **Retry** to initiate syndication for individual items, or use the **Retry All** in the Important Items tab to try to syndicate all failed items. You can also choose to update or rebuild a syndication relationship.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

"Resetting the web content event log" on page 1195
From time to time, you might need to reset the web content event log. The event log can be reset only on a syndicator server. Any changes that are made by resetting the event log are then syndicated to its corresponding subscribers. In most cases, you reset the event log on the server you imported or migrated data onto, or on a syndicator to troubleshoot syndication problems in a syndication relationship.

Creating a syndication relationship from the command line

You can set up syndication relationships by using the Administration Portlets or the command line. To set up a syndication relationship from the command line, use the XML configuration interface (XML access) and the ConfigEngine command to configure the subscriber.

Before you begin

Make sure that you have the appropriate configuration setup before you enable syndication.

Disk space

Before you use syndication, you must ensure that your subscriber has sufficient memory to receive the data that is syndicated from the syndicator. For example, if you are going to syndicate all libraries, then you need at least as much space on your subscriber as the database used by your syndicator.

Swap space

Ensure that you have at least as much swap space allocated on the subscriber server as you have physical memory.

Cross version syndication

Cross-version syndication is supported between the following releases. Syndication from a newer release to an older release is not supported:

- WebSphere Portal version 7.0.0.2 with CF26 or higher.
- WebSphere Portal 8.0.0.1 with CF09 or higher.
- WebSphere Portal 8.5 or higher.

Library already exists on the subscriber

First-time syndication to an existing library is not supported. If you attempt to syndicate a library to a subscriber that already has a library with the same name, an error results.

Large library with more that 10000 items

To syndicate a library that contains more than 10000 items, update the maximum Java heap size that is used by the portal application server on the subscriber server:

1. In the WebSphere Integrated Solutions Console, browse to the Java virtual machine settings.

   Stand-alone server:

   Servers > **Server Types** > **WebSphere application servers** > **WebSphere_Portal** > **Java and Process Management** > **Process definition** > **Java Virtual Machine**

   Clustered server:

   System administration > **Deployment manager** > **Java and Process Management** > **Process Definition** > **Java Virtual Machine**
2. Update the value in the **Maximum Heap Size** field. A value of at least 1024 MB is recommended.
3. Click **OK**, and then save your changes.

About this task

To set up syndication between web content libraries on two IBM Web Content Manager applications, establish a relationship between a syndicator and a subscriber. The syndicating server contains the data to be replicated, and the subscribing server receives the replicated data.

What can and cannot be syndicated:

- Information about a web content library is only syndicated the first time syndication occurs and not on subsequent updates and rebuilds. If a library is renamed or library user access is changed, this information is not syndicated to the Subscriber.
- If you change the name of a library or change user access to a library, these changes are not syndicated. If you want the same settings on all your syndicated libraries, you must manually make the same changes to any subscriber libraries.
- If content from one library (Library A) uses an item from another library (Library B), you must include both libraries in the syndicator. Including both libraries ensures that all items are syndicated successfully.
- If you include only Library A in the syndicator, any items in Library A that reference items in Library B are not syndicated. Syndication errors are also generated.
- If you add a library to a syndicator after the initial syndication, you must click **Rebuild** to force the new library to be syndicated immediately.
- If you are creating a two-way syndication relationship, you must use a consistent syndication strategy. For example, if you syndicate "All items", then both syndication relationships must be syndicating "All items".

Procedure

1. Ensure both the subscriber and syndicator are running and that they can access each other over a network.
2. On the subscriber server, create a shared credential vault slot with the XML configuration interface.
   a. Create the `CreateVaultSlot.xml` file using a text editor.

This sample file uses the following values that you can change to reflect your environment:

syndication-slot
The name of the shared credential vault slot.

wpsadmin
The user ID for the portal administrator.

passw0rd
The password for the portal administrator.

```
<?xml version="1.0" encoding="UTF-8"?>
<request
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
    type="update" create-oids="true">

    <!-- Sample for creating a new credential vault slot. This script creates a     -->
    <!-- credential vault resource and a shared slot in the Default Admin Segment   -->
    <portal action="locate">
         <credential-segment action="locate" adapter-type="default" name="DefaultAdminSegment"
             user-mapped="false">
             <description>Default Admin Segment</description>
             <credential-slot action="update" active="false" name="syndication-slot"
                 resource="syndication-resource" secrettype="userid-password" system="true">
                 <localedata locale="en">
                     <description>used for syndicator and subscriber pair</description>
                 </localedata>
                 <password-secret action="create" external-id="wpsadmin"
                     user="uid=wpsadmin,o=defaultWIMFileBasedRealm">passw0rd</password-secret>
             </credential-slot>
         </credential-segment>
    </portal>
</request>
```

b. Run the **xmlaccess** command with the CreateVaultSlot.xml file.

Linux `./xmlaccess.sh -in CreateVaultSlot.xml -out slot-out.xml -url http://localhost:10039/wps/config -user wpsadmin -password passw0rd`

IBM i `xmlaccess.sh -in CreateVaultSlot.xml -out slot-out.xml -url http://localhost:10039/wps/config -user wpsadmin -password passw0rd`

Windows
`xmlaccess.bat -in CreateVaultSlot.xml -out slot-out.xml -url http://localhost:10039/wps/config -user wpsadmin -password passw0rd`

3. On the subscriber server, create a virtual portal on the subscriber with the create-virtual-portal task.

   This example creates a virtual portal called *sample*, although you can change this value to reflect your environment.

Linux `./ConfigEngine.sh create-virtual-portal -DPortalAdminPwd=passw0rd -DWasPassword=passw0rd -DVirtualPortalTitle=sample -DVirtualPortalContext=sample`

IBM i `ConfigEngine.sh create-virtual-portal -DPortalAdminPwd=passw0rd -DWasPassword=passw0rd -DVirtualPortalTitle=sample -DVirtualPortalContext=sample`

Windows
`ConfigEngine.bat create-virtual-portal -DPortalAdminPwd=passw0rd -DWasPassword=passw0rd -DVirtualPortalTitle=sample -DVirtualPortalContext=sample`

4. On the subscriber server, set up the syndication relationship with the **run-wcm-admin-task-subscribe-now** task.

   This sample command uses the following additional values that you can change to reflect your environment:

syndicator-hostname
The host name of the syndicator server.

syndicator1
This name is used for the syndicator item that is created on the syndicator server. Enter a name that helps identify the syndication relationship you are creating. This name must be unique and cannot be the same as an existing syndicator name.

Note: To reuse syndicator names of previously deleted syndication relationships on a subscriber, you must also delete the same relationship on the syndicator.

subscriber1
This name is used for the subscriber item that is created on the subscriber server. Enter a name that helps identify the syndication relationship you are creating. This name must be unique and cannot be the same as an existing subscriber name.

updateAfterCreation
If set to `true`, a syndication update is run as soon as the syndication pair is successfully created. If not specified, the default setting is `true`. This is a one-off syndication event not related to any automatic configuration settings.

In addition, use the following properties to identify the libraries to which you are subscribing and the type of syndication that you want to perform. For each syndication relationship, you can specify only one type of syndication. Separate multiple libraries with commas.

liveItems="*library_name_1*,*library_name_2*"
Live item syndication is mostly used when you syndicate to a staging or delivery server. The following items are syndicated:

- Published
- Expired

Draft items, projects, and items in a project are not syndicated.

liveProjectsItems="*library_name_1*,*library_name_2*"
Use "Live and projects" syndication to gradually syndicate projects to a staging or delivery server, rather than waiting until all items in a project achieve a published state. The following items are syndicated:

- Published
- Expired
- Projects
- Draft items in a project

Draft items outside of projects are not syndicated.

allItems="*library_name_1*,*library_name_2*"
All item syndication is mostly used when you syndicate between servers within an authoring environment. The following items are syndicated:

- Published
- Expired
- Projects
- Draft items in a project
- Other draft items

- Versions
- Deleted items

CF05 **mode="*modetype*"**

- **Configured**: This uses the mode configured in the WCM WCMConfigService service.
- **Automatic**: Syndication will be scheduled automatically based on the configured syndication interval.
- **Manual**: Syndication will occur only when requested using the administration portlet.

Example commands:

Linux
```
./ConfigEngine.sh run-wcm-admin-task-subscribe-now
-Dsyndicator=http://syndicator-hostname:10039/wps/wcm
-DvaultSlotName=syndication-slot -DsyndicatorName=syndicator1
-DsubscriberName=subscriber1 -DVirtualPortalContext=sample
-DliveItems="Web Content,Portal Site" -DPortalAdminPwd=passw0rd
-DWasPassword=passw0rd
```

IBM i
```
ConfigEngine.sh run-wcm-admin-task-subscribe-now
-Dsyndicator=http://syndicator-hostname:10039/wps/wcm
-DvaultSlotName=syndication-slot -DsyndicatorName=syndicator1
-DsubscriberName=subscriber1 -DVirtualPortalContext=sample
-DliveItems="Web Content,Portal Site" -DPortalAdminPwd=passw0rd
-DWasPassword=passw0rd
```

Windows
```
ConfigEngine.bat run-wcm-admin-task-subscribe-now
-Dsyndicator=http://syndicator-hostname:10039/wps/wcm
-DvaultSlotName=syndication-slot -DsyndicatorName=syndicator1
-DsubscriberName=subscriber1 -DVirtualPortalContext=sample
-DliveItems="Web Content,Portal Site" -DPortalAdminPwd=passw0rd
-DWasPassword=passw0rd
```

Database Management Systems

Configure the connection between IBM WebSphere Portal Express and your database management system. The **Database Transfer** configuration option in the Configuration Wizard assigns users and permissions, creates databases, obtains support for database collation, and transfers your database.

"DB2: Database transfer" on page 467
Your portal is installed with an Apache Derby database. The database that is immediately available for use is good for demonstration and portlet and theme development environments. Otherwise, you must configure portal to use a production-level database. Use the Configuration Wizard to transfer the data and configure a new database server.

"IBM DB2 for i: Database transfer" on page 476
Your portal is installed with an Apache Derby database. The database that is immediately available for use is good for demonstration and portlet and theme development environments. Otherwise, you must configure portal to use a production-level database. Use the Configuration Wizard to transfer the data and configure a new database server.

"DB2 for z/OS: Database transfer" on page 480
Your portal is installed with an Apache Derby database. The database that is immediately available for use is good for demonstration and portlet and theme

development environments. Otherwise, you must configure portal to use a production-level database. Use the Configuration Wizard to transfer the data and configure a new database server.

"SQL Server: Database transfer" on page 487
Your portal is installed with an Apache Derby database. The database that is immediately available for use is good for demonstration and portlet and theme development environments. Otherwise, you must configure portal to use a production-level database. Use the Configuration Wizard to transfer the data and configure a new database server.

"Oracle: Database transfer" on page 495
Your portal is installed with an Apache Derby database. The database that is immediately available for use is good for demonstration and portlet and theme development environments. Otherwise, you must configure portal to use a production-level database. Use the Configuration Wizard to transfer the data and configure a new database server.

"Manual Steps: Database Transfer option in the Configuration Wizard" on page 504
The database transfer option in the Configuration Wizard includes manual steps. Some manual steps include scripts for you or your database administrator to run. For reference only, you can see the manual steps in this section of the product documentation. Use the Configuration Wizard to complete the deployment configuration. The wizard generates specific instructions and scripts based on your unique input.

"Oracle: Creating JCR table spaces (Automatic Storage Management)" on page 554
If you configured your database with Automatic Storage Management, you might need to perform additional manual instructions to create JCR table spaces when you use the Database Transfer option in the Configuration Wizard . If you select the option to let the wizard create your database schemas and assign permissions, you must perform the steps in this topic after you run the setup database script.

"Assigning custom table spaces" on page 555
You cannot use the **Database Transfer** option in the configuration wizard to assign custom table spaces on your database server. You can perform manual steps to assign custom table spaces.

"DB2: Enabling support for high availability recovery and rollforward recovery " on page 560
Optional: To prevent data loss on DB2, modify the JCR schema to support High Availability Disaster Recovery and rollforward recovery.

"Connecting to existing database domains" on page 561
View the steps to share a database domain between two separate WebSphere Portal Express instances (*instance1* and *instance2*), where *instance2* is connected to an *instance1* database domain.

DB2: Database transfer

Your portal is installed with an Apache Derby database. The database that is immediately available for use is good for demonstration and portlet and theme development environments. Otherwise, you must configure portal to use a production-level database. Use the Configuration Wizard to transfer the data and configure a new database server.

Configuration Wizard

The primary Configuration Wizard options are based on your target configuration topology, such as a stand-alone server or a cluster. The database transfer option is included with both **Set Up a Stand-alone Server** and **Set Up a Cluster**. For the stand-alone server topology, run the database transfer option before you run the federated security option. For the cluster option, run the database transfer before you create your deployment manager profile.

Validation

Prevent a possible database transfer failure by validating your entries in the wizard. When you chose to validate settings for the Database Transfer option, field syntax and database connection validations are performed to prevent a possible failure before you run the configuration.

Syntax validation checks that you:

- Enter a valid port number in the range of 1 - 65535.
- Use the correct format for your host name.

The database connection validation checks that a connection can be made to your database server. The wizard can connect to your database server to validate the host name and port number values that you enter.

On the Database Settings panel, you can choose to turn validation on or off by selecting **Yes** or **No** to the **Connect to database server to validate settings** option. Select **No**, if you know that your parameters are correct and your database server is unavailable at the time of creating your instructions.

"DB2: DB2 worksheet: Transfer to a single database" on page 471
When you use the database transfer option, you can select the condition to create a single database in the Configuration Wizard. This worksheet highlights the fields and properties that you see in the Configuration Wizard when you select the single database condition.

"DB2: DB2 worksheet: Transfer to multiple databases" on page 472
When you use the database transfer option, you can use the condition to create multiple databases in the Configuration Wizard. This condition is selected by default. This worksheet highlights the fields and properties that you see in the Configuration Wizard when you use the multiple databases condition.

Transferring data to a different database server

Depending on your environment, the wizard generates a configuration process. The following steps reflect all possible steps in the configuration process. The steps do not represent a literal configuration. The steps are provided as a reference.

About this task

If you click **View Step Command**, you can see the task and properties associated with each step in the wizard.

Procedure

1. Back up the properties files that the wizard uses during the configuration.

   Condition

   None

ConfigEngine task
backup-property-files-for-dbxfer

2. Manual Step: Create the database users and groups.

   Condition
   None

   ConfigEngine task
   None

3. Create your databases.

   Condition
   None

   ConfigEngine task
   create-database

4. Manual Step: Download the script and run it on the database server to create your database.

   Condition
   Select to manually create your database.

   ConfigEngine task
   None

5. Set up your database.

   Condition
   None

   ConfigEngine task
   setup-database

6. Manual Step: Download the script and run it on the database server to set up your database.

   Condition
   Select to manually assign users permissions.

   ConfigEngine task
   None

7. Manual Step: Set up JCR collation for correct language locale order.

   Condition
   Select to have advanced database collation support.

   ConfigEngine task
   None

8. Manual Step: Restart the DB2 server.

   Condition
   None

   ConfigEngine task
   None

9. Validate the database connection and environment.

   Condition
   None

   ConfigEngine task
   validate-database validate-database-environment

10. Stop the portal server.

Condition
collation support

ConfigEngine task
stop-portal-server

11. Transfer the database.

    Condition
    None

    ConfigEngine task
    database-transfer enable-profiles-check-managed package-profiles

12. Grant privileges to the database runtime users.

    Condition
    None

    ConfigEngine task
    grant-runtime-db-user-privileges

13. Manual Step: Download the script and run it on the database server to grant privileges to the runtime user.

    Condition
    Select to manually create users and assign them permissions.

    ConfigEngine task
    None

14. Configure the JCR domain to support large files.

    Condition
    None

    ConfigEngine task
    datasource-enable-fully-materialize-lob-data

15. Start the portal server.

    Condition
    None

    ConfigEngine task
    start-portal-server

What to do next

You transferred your data from Apache Derby to your preferred database.

One quick way to test your database configuration is to log in and explore the site to validate the site is working as you expected.

Go to http://*host_name*:*port*/*context_root*/*default_portal_home*. For example, go to http://*host_name:10039*/*wps*/*portal*.

Next, you can use other options to configure your environment more.

If you are setting up a stand-alone server environment, you can use the Enable Federated Security option to add an LDAP user registry.

If you are setting up a cluster environment, you can use the Create a Deployment Manager option to create a deployment manager profile that is augmented with WebSphere Portal Express resources.

DB2: DB2 worksheet: Transfer to a single database

When you use the database transfer option, you can select the condition to create a single database in the Configuration Wizard. This worksheet highlights the fields and properties that you see in the Configuration Wizard when you select the single database condition.

Typical fields

When you use the database transfer option, you answer questions about your environment. Some fields are required. Other fields are required or removed based on your selections for environment conditions.

The following table lists the typical fields that display when you select the option to transfer your data to a single database. To see additional fields that apply to an advanced configuration, click **Advanced**.

Values that you enter for some typical fields apply across domains. Rather than entering a value multiple times, this value is copied to fields in the Advanced view and applied across the database domains.

Table 56. Transfer to a single database..

You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Condition	Your Value
Database name	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbName` properties.		
Data source	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DataSourceName` properties.		
Database URL	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbUrl` properties.		
Configuration user ID	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbUser` properties.		
Configuration password	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbPassword` properties.		
Database administrator ID	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DBA.DbUser` properties.		
Database administrator password	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DBA.DbPassword` properties.		

Table 56. Transfer to a single database. (continued).

You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Condition	Your Value
Runtime user	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbRuntimeUser` properties.	To see this field, continue to use the default selection of **Yes** for needing a runtime database user for day-to-day operations.	
Runtime password	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbRuntimeDbRuntimePassword` properties.		
IBM DB2 library	`Db2.DbLibrary`		

Field used for the database collation condition

Database collation is an optional condition available to you.

Table 57. Database collation

Field Label	Property	Conditions	Your Value
Temporary directory to be used for collation		To see this field, you must select the **Yes** option for advanced database collation support. The default selection is **No**.	

DB2: DB2 worksheet: Transfer to multiple databases

When you use the database transfer option, you can use the condition to create multiple databases in the Configuration Wizard. This condition is selected by default. This worksheet highlights the fields and properties that you see in the Configuration Wizard when you use the multiple databases condition.

Typical fields

When you use the database transfer option, you answer questions about your environment. Some fields are required. Other fields are required or removed based on your selections for environment conditions.

The following table lists the typical fields that display when you select the condition to transfer your data to multiple databases. To see additional fields that apply to an advanced configuration, click **Advanced**.

Depending on the conditions that you select, values that you enter for some typical fields might apply across domains. Rather than entering a value multiple times, this value is copied to fields in the Advanced view and applied across the database domains.

Table 58. Transfer to multiple database. You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Conditions	Your Value
Release database name	`release.DbName`		
Release data source	`release.DataSourceName`		
Release database URL	`release.DbUrl`		
Community database name	`community.DbName`		
Community data source	`community.DataSourceName`		
Community database URL	`community.DbUrl`		
Customization database name	`customization.DbName`		
Customization Data source	`customization.DataSourceName`		
Customization database URL	`customization.DbUrl`		
JCR database URL	`JCR.DbName`		
JCR database name	`JCR.DataSourceName`		
JCR data source	`JCR.DbUrl`		
Feedback database name	`feedback.DbName`		
Feedback data source	`feedback.DataSourceName`		
Feedback database URL	`feedback.DbUrl`		
Likeminds Database name	`likeminds.DbName`		
Likeminds data source	`likeminds.DataSourceName`		
Likeminds database URL	`likeminds.DbUrl`		
IBM DB2 library	`Db2.DbLibrary`		

Table 58. Transfer to multiple database (continued). You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Conditions	Your Value
Configuration user ID	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbUser` properties.	To see this field, continue to use the **Yes** selection for using the same user ID and passwords across portal database domains.	
Configuration password	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbPassword` properties.		
Database administrator ID	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DBA.DbUser` properties.		
Database administrator password	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DBA.DbPassword` properties.		

Table 58. Transfer to multiple database (continued). You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Conditions	Your Value
Release configuration user	**release.DbUser**	To see this field, select **No** for using the same user ID and passwords across portal database domains.	
Release configuration password	**release.DbPassword**		
Release database administrator	**release.DBA.DbUser**		
Release database administrator password	**release.DBA.DbPassword**		
Community configuration user	**community.DbUser**		
Community configuration password	**community.DbPassword**		
Community database administrator	**community.DBA.DbUser**		
Community database administrator password	**community.DBA.DbPassword**		
Customization configuration user	**customization.DbUser**		
Customization configuration password	**customization.DbPassword**		
Customization database administrator	**customization.DBA.DbUser**		
Customization database administrator password	**customization.DBA.DbPassword**		
JCR configuration user	**jcr.DbUser**		
JCR configuration password	**jcr.DbPassword**		
JCR database administrator	**jcr.DBA.DbUser**		
JCR database administrator password	**jcr.DBA.DbPassword**		
Feedback configuration user	**feedback.DbUser**		
Feedback configuration password	**feedback.DbPassword**		
Feedback database administrator	**feedback.DBA.DbUser**		
Feedback database administrator password	**feedback.DBA.DbPassword**		
Likeminds configuration user	**likeminds.DbUser**		
Likeminds configuration password	**likeminds.DbPassword**		
Likeminds database administrator	**likeminds.DBA.DbUser**		
Likeminds database administrator password	**likeminds.DBA.DbPassword**		

Table 58. Transfer to multiple database (continued). You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Conditions	Your Value
Runtime user	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbRuntimeUser` properties.	To see this field: • Continue to use the **Yes** selection for using the same user ID and passwords across portal database domains. • Continue to use the default selection of **Yes** for needing a runtime database user for day-to-day operations.	
Runtime password	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbRuntimeDbRuntimePassword` properties.		
Release runtime user	`release.DbRuntimeUser`	To see this field: • Select **No** for using the same user ID and passwords across portal database domains. • Continue to use the default selection of **Yes** for needing a runtime database user for day-to-day operations	
Release runtime password	`release.DbRuntimePassword`		
Community runtime user	`community.DbRuntimeUser`		
Community runtime password	`community.DbRuntimePassword`		
Customization runtime user	`customization.DbRuntimeUser`		
Customization runtime password	`customization.DbRuntimePassword`		
JCR runtime user	`jcr.DbRuntimeUser`		
JCR runtime password	`jcr.DbRuntimePassword`		
Feedback runtime user	`feedback.DbRuntimeUser`		
Feedback runtime password	`feedback.DbRuntimePassword`		
Likeminds runtime user	`likeminds.DbRuntimeUser`		
Likeminds runtime password	`likeminds.DbRuntimePassword`		

Field used for the database collation condition

Database collation is an optional condition available to you.

Table 59. Database collation

Field Label	Property	Conditions	Your Value
Temporary directory to be used for collation		To see this field, you must select the **Yes** option for advanced database collation support. The default selection is **No**.	

IBM DB2 for i: Database transfer

Your portal is installed with an Apache Derby database. The database that is immediately available for use is good for demonstration and portlet and theme

development environments. Otherwise, you must configure portal to use a production-level database. Use the Configuration Wizard to transfer the data and configure a new database server.

Configuration Wizard

The primary Configuration Wizard options are based on your target configuration topology, such as a stand-alone server or a cluster. The database transfer option is included with both **Set Up a Stand-alone Server** and **Set Up a Cluster**. For the stand-alone server topology, run the database transfer option before you run the federated security option. For the cluster option, run the database transfer before you create your deployment manager profile.

"IBM DB2 for i: IBM DB2 for i worksheet: Transfer your database" on page 478
This worksheet highlights the fields and properties that you see in the Configuration Wizard when you use the transfer database option.

Transferring data to a different database server

Depending on your environment, the wizard generates a configuration process. The following steps reflect all possible steps in the configuration process. The steps do not represent a literal configuration. The steps are provided as a reference.

About this task

If you click **View Step Command**, you can see the task and properties associated with each step in the wizard.

Procedure

1. Back up the properties files that the wizard uses during the configuration.

   Condition
   None

   ConfigEngine task
   backup-property-files-for-dbxfer
2. Manual Step: Create the database user profile on IBM DB2 for i.

   Condition
   None

   ConfigEngine task
   None
3. Manual Step: Create the database runtime users and groups.

   Condition
   None

   ConfigEngine task
   None
4. Create your databases.

   Condition
   None

   ConfigEngine task
   create-database
5. Validate the database connection and environment.

Condition
None

ConfigEngine task
validate-database validate-database-environment

6. Stop the portal server.

   Condition
   collation support

   ConfigEngine task
   stop-portal-server

7. Transfer the database.

   Condition
   None

   ConfigEngine task
   database-transfer enable-profiles-check-managed package-profiles

8. Grant privileges to the database runtime users.

   Condition
   None

   ConfigEngine task
   grant-runtime-db-user-privileges

9. Start the portal server.

   Condition
   None

   ConfigEngine task
   start-portal-server

What to do next

You transferred your data from Apache Derby to your preferred database.

One quick way to test your database configuration is to log in and explore the site to validate the site is working as you expected.

Go to http://*host_name*:*port*/*context_root*/*default_portal_home*. For example, go to http://*host_name:10039*/*wps*/*portal*.

Next, you can use other options to configure your environment more.

If you are setting up a stand-alone server environment, you can use the Enable Federated Security option to add an LDAP user registry.

If you are setting up a cluster environment, you can use the Create a Deployment Manager option to create a deployment manager profile that is augmented with WebSphere Portal Express resources.

IBM DB2 for i: IBM DB2 for i worksheet: Transfer your database

This worksheet highlights the fields and properties that you see in the Configuration Wizard when you use the transfer database option.

Typical fields

When you use the database transfer option, you answer questions about your environment. Some fields are required. Other fields are required or removed based on your selections for environment conditions.

The following table lists the typical fields that display when you select the database transfer option. To see additional fields that apply to an advanced configuration, click **Advanced**.

Table 60. Transfer database..

You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Condition	Your Value
Database name	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbName` properties.		
Data source	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DataSourceName` properties.		
Database URL	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbUrl` properties.		
Configuration user ID	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbUser` properties.		
Configuration password	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbPassword` properties.		
Database administrator ID	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DBA.DbUser` properties.		
Database administrator password	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DBA.DbPassword` properties.		
Runtime user	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbRuntimeUser` properties.	To see this field, continue to use the default selection of **Yes** for needing a runtime database user for day-to-day operations.	
Runtime password	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbRuntimeDbRuntimePassword` properties.		
IBM for DB2i connection type	`db2_iseries.DbDriverType`		

Table 60. Transfer database. (continued).

You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Condition	Your Value
IBM for DB2i library	**db2_iseries.DbLibrary**		

DB2 for z/OS: Database transfer

Your portal is installed with an Apache Derby database. The database that is immediately available for use is good for demonstration and portlet and theme development environments. Otherwise, you must configure portal to use a production-level database. Use the Configuration Wizard to transfer the data and configure a new database server.

Configuration Wizard

The primary Configuration Wizard options are based on your target configuration topology, such as a stand-alone server or a cluster. The database transfer option is included with both **Set Up a Stand-alone Server** and **Set Up a Cluster**. For the stand-alone server topology, run the database transfer option before you run the federated security option. For the cluster option, run the database transfer before you create your deployment manager profile.

Validation

Prevent a possible database transfer failure by validating your entries in the wizard. When you chose to validate settings for the Database Transfer option, field syntax and database connection validations are performed to prevent a possible failure before you run the configuration.

Syntax validation checks that you:

- Enter a valid port number in the range of 1 - 65535.
- Use the correct format for your host name.

The database connection validation checks that a connection can be made to your database server. The wizard can connect to your database server to validate the host name and port number values that you enter.

On the Database Settings panel, you can choose to turn validation on or off by selecting **Yes** or **No** to the **Connect to database server to validate settings** option. Select **No**, if you know that your parameters are correct and your database server is unavailable at the time of creating your instructions.

"DB2 for z/OS: DB2 for z/OS worksheet: Transfer your database" on page 482
This worksheet highlights the fields and properties that you see in the Configuration Wizard when you use the database transfer option.

Transferring data to a different database server

Depending on your environment, the wizard generates a configuration process. The following steps reflect all possible steps in the configuration process. The steps do not represent a literal configuration. The steps are provided as a reference.

About this task

If you click **View Step Command**, you can see the task and properties associated with each step in the wizard.

Procedure

1. Back up the properties files that the wizard uses during the configuration.

   Condition
   None

   ConfigEngine task
   backup-property-files-for-dbxfer

2. Manual Step: Create the database configuration users on DB2 for z/OS.

   Condition
   None

   ConfigEngine task
   None

3. Manual Step: Download the script and view instructions to delete existing databases.

   Condition
   None

   ConfigEngine task
   None

4. Manual Step: Download the script and view instructions to create your databases.

   Condition
   None

   ConfigEngine task
   None

5. Validate the database connection and environment.

   Condition
   None

   ConfigEngine task
   validate-database validate-database-environment

6. Stop the portal server.

   Condition
   collation support

   ConfigEngine task
   stop-portal-server

7. Transfer the database.

   Condition
   None

   ConfigEngine task
   database-transfer enable-profiles-check-managed package-profiles

8. Grant privileges to the database runtime users.

   Condition
   None

ConfigEngine task
`grant-runtime-db-user-privileges`

9. Connect to your databases.

   Condition
   None

   ConfigEngine task
   None

10. Manual Step: Download the script and view instructions to reset the check pending status on portal table spaces.

    Condition
    None

    ConfigEngine task
    None

11. Reset the web content manager event log.

    Condition
    None

    ConfigEngine task
    None

12. Start the portal server.

    Condition
    None

    ConfigEngine task
    `start-portal-server`

What to do next

You transferred your data from Apache Derby to your preferred database.

One quick way to test your database configuration is to log in and explore the site to validate the site is working as you expected.

Go to http://*host_name*:*port*/*context_root*/*default_portal_home*. For example, go to http://*host_name:10039*/*wps*/*portal*.

Next, you can use other options to configure your environment more.

If you are setting up a stand-alone server environment, you can use the Enable Federated Security option to add an LDAP user registry.

If you are setting up a cluster environment, you can use the Create a Deployment Manager option to create a deployment manager profile that is augmented with WebSphere Portal Express resources.

DB2 for z/OS: DB2 for z/OS worksheet: Transfer your database

This worksheet highlights the fields and properties that you see in the Configuration Wizard when you use the database transfer option.

Typical fields

When you use the database transfer option, you answer questions about your environment. Some fields are required. Other fields are required or removed based on your selections for environment conditions.

The following table lists the typical fields that display when you select the database transfer option. To see additional fields that apply to an advanced configuration, click **Advanced**.

Depending on the conditions that you select, values that you enter for some typical fields might apply across domains. Rather than entering a value multiple times, this value is copied to fields in the Advanced view and applied across the database domains.

Table 61. Fields that display in the Configuration Wizard.

You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Conditions	Your Value
DB2 SYSADM authorization ID:			
DB2 location name			
Release DB2 for z/OS database name	`release.DbNameOnZos`		
Release data source	`release.DataSourceName`		
Release database URL	`release.DbUrl`		
Release storage group	`release.DbStorageGroup`		
Release VCAT	`release.DbVCat`		
Release 32k buffer pool	`release.Db32KBufferPoolName`		
Release 4 K buffer pool	`release.Db4KBufferPoolName`		
Release 4 K buffer pool indexes	`release.DbIndex4KBufferPoolName`		
Community DB2 for z/OS database name	`community.DbNameOnZos`		
Community data source	`community.DataSourceName`		
Community database URL	`community.DbUrl`		
Community storage group	`community.DbStorageGroup`		
Community VCAT	`community.DbVCat`		
Community 32 K buffer pool	`community.Db32KBufferPoolName`		
Community 4 K buffer pool	`community.Db4KBufferPoolName`		
Community 4 K buffer pool indexes	`community.DbIndex4KBufferPoolName`		
Customization DB2 for z/OS database name	`customization.DbNameOnZos`		
Customization data source	`customization.DataSourceName`		
Customization database URL	`customization.DbUrl`		
Customization storage group	`customization.DbStorageGroup`		
Customization VCAT	`customization.DbVCat`		
Customization 32 K buffer pool	`customization.Db32KBufferPoolName`		

Table 61. Fields that display in the Configuration Wizard (continued).

You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Conditions	Your Value
Customization 4 K buffer pool	customization.Db4KBufferPoolName		
Customization 4 K buffer pool indexes	customization.DbIndex4KBufferPoolName		
JCR DB2 for z/OS database name	jcr.DbNameOnZos		
JCR data source	jcr.DataSourceName		
JCR database URL	jcr.DbUrl		
JCR storage group	jcr.DbStorageGroup		
JCR database host name	jcr.DbHost		
JCR domain	jcr.DbDomain		
JCR port	jcr.DbPort		
JCR storage group	jcr.DbStorageGroup		
JCR VCAT	jcr.DbVcat		
JCR 32 K buffer pool	jcr.Db32KBufferPoolName		
JCR 4 K buffer pool	jcr.Db4KBufferPoolName		
JCR 4 K buffer pool for indexes	jcr.DbIndex4KBufferPoolName		
Feedback database name	feedback.DbNameOnZos		
Feedback data source	feedback.DataSourceName		
Feedback database URL	feedback.DbUrl		
Feedback storage group	feedback.DbStorageGroup		
Feedback VCAT	feedback.DbVcat		
Feedback 32 K buffer pool	feedback.Db32KBufferPoolName		
Feedback 4 K buffer pool	feedback.Db4KBufferPoolName		
Likeminds database name	likeminds.DbNameOnZos		
Likeminds data source	likeminds.DataSourceName		
Likeminds database URL	likeminds.DbUrl		
Likeminds storage group	likeminds.DbStorageGroup		
Likeminds VCAT	likeminds.DbVcat		
Likeminds 32 K buffer pool	likeminds.Db32KBufferPoolName		
Likeminds 4 K buffer pool	likeminds.Db4KBufferPoolName		
DB2 subsystem name			
IBM DB2 for z/OS library	db2_zos.DbLibrary		
Database group			

Table 61. Fields that display in the Configuration Wizard (continued).

You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Conditions	Your Value
Configuration user ID	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbUser` properties.	To see this field, continue to use the **Yes** selection for using the same user ID and passwords across portal database domains.	
Configuration password	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbPassword` properties.		
Database administrator ID	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DBA.DbUser` properties.		
Database administrator password	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DBA.DbPassword` properties.		

Table 61. Fields that display in the Configuration Wizard (continued).

You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Conditions	Your Value
Release configuration user	**release.DbUser**	To see this field, select **No** for using the same user ID and passwords across portal database domains.	
Release configuration password	**release.DbPassword**		
Release database administrator	**release.DBA.DbUser**		
Release database administrator password	**release.DBA.DbPassword**		
Community configuration user	**community.DbUser**		
Community configuration password	**community.DbPassword**		
Community database administrator	**community.DBA.DbUser**		
Community database administrator password	**community.DBA.DbPassword**		
Customization configuration user	**customization.DbUser**		
Customization configuration password	**customization.DbPassword**		
Customization database administrator	**customization.DBA.DbUser**		
Customization database administrator password	**customization.DBA.DbPassword**		
JCR configuration user	**jcr.DbUser**		
JCR configuration password	**jcr.DbPassword**		
JCR database administrator	**jcr.DBA.DbUser**		
JCR database administrator password	**jcr.DBA.DbPassword**		
Feedback configuration user	**feedback.DbUser**		
Feedback configuration password	**feedback.DbPassword**		
Feedback database administrator	**feedback.DBA.DbUser**		
Feedback database administrator password	**feedback.DBA.DbPassword**		
Likeminds configuration user	**likeminds.DbUser**		
Likeminds configuration password	**likeminds.DbPassword**		
Likeminds database administrator	**likeminds.DBA.DbUser**		
Likeminds database administrator password	**likeminds.DBA.DbPassword**		

Table 61. Fields that display in the Configuration Wizard (continued).

You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Conditions	Your Value
Runtime user	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbRuntimeUser` properties.	To see this field: • Continue to use the **Yes** selection for using the same user ID and passwords across portal database domains. • Continue to use the default selection of **Yes** for needing a runtime database user for day-to-day operations.	
Runtime password	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbRuntimeDbRuntimePassword` properties.		
Release runtime user	`release.DbRuntimeUser`	To see this field: • Select **No** for using the same user ID and passwords across portal database domains. • Continue to use the default selection of **Yes** for needing a runtime database user for day-to-day operations	
Release runtime password	`release.DbRuntimePassword`		
Community runtime user	`community.DbRuntimeUser`		
Community runtime password	`community.DbRuntimePassword`		
Customization runtime user	`customization.DbRuntimeUser`		
Customization runtime password	`customization.DbRuntimePassword`		
JCR runtime user	`jcr.DbRuntimeUser`		
JCR runtime password	`jcr.DbRuntimePassword`		
Feedback runtime user	`feedback.DbRuntimeUser`		
Feedback runtime password	`feedback.DbRuntimePassword`		
Likeminds runtime user	`likeminds.DbRuntimeUser`		
Likeminds runtime password	`likeminds.DbRuntimePassword`		

SQL Server: Database transfer

Your portal is installed with an Apache Derby database. The database that is immediately available for use is good for demonstration and portlet and theme development environments. Otherwise, you must configure portal to use a production-level database. Use the Configuration Wizard to transfer the data and configure a new database server.

Configuration Wizard

The primary Configuration Wizard options are based on your target configuration topology, such as a stand-alone server or a cluster. The database transfer option is included with both **Set Up a Stand-alone Server** and **Set Up a Cluster**. For the stand-alone server topology, run the database transfer option before you run the federated security option. For the cluster option, run the database transfer before you create your deployment manager profile.

Validation

Prevent a possible database transfer failure by validating your entries in the wizard. When you chose to validate settings for the Database Transfer option, field syntax and database connection validations are performed to prevent a possible failure before you run the configuration.

Syntax validation checks that you:

- Enter a valid port number in the range of 1 - 65535.
- Use the correct format for your host name.

The database connection validation checks that a connection can be made to your database server. The wizard can connect to your database server to validate the host name and port number values that you enter.

On the Database Settings panel, you can choose to turn validation on or off by selecting **Yes** or **No** to the **Connect to database server to validate settings** option. Select **No**, if you know that your parameters are correct and your database server is unavailable at the time of creating your instructions.

"SQL Server: SQL Server worksheet: Transfer to a single database" on page 490
When you use the database transfer option, you can select the condition to create a single database in the Configuration Wizard. This worksheet highlights the fields and properties that you see in the Configuration Wizard when you select the single database condition.

"SQL Server: SQL Server worksheet: Transfer to multiple databases" on page 491
When you use the database transfer option, you can use the condition to create multiple databases in the Configuration Wizard. This condition is selected by default. This worksheet highlights the fields and properties that you see in the Configuration Wizard when you use the multiple databases condition.

Transferring data to a different database server

Depending on your environment, the wizard generates a configuration process. The following steps reflect all possible steps in the configuration process. The steps do not represent a literal configuration. The steps are provided as a reference.

About this task

If you click **View Step Command**, you can see the task and properties associated with each step in the wizard.

Procedure

1. Back up the properties files that the wizard uses during the configuration.

   Condition
   None

   ConfigEngine task
   backup-property-files-for-dbxfer

2. Create your databases.

   Condition
   None

   ConfigEngine task
   create-database

3. Manual Step: Download the script and run it on the database server to create your database.

   Condition
   Select to manually create your database.

   ConfigEngine task
   None

4. Set up your database.

   Condition
   None

   ConfigEngine task
   `setup-database`

5. Manual Step: Download the script and run it on the database server to set up your database.

   Condition
   Select to manually create users and assign them permissions.

   ConfigEngine task
   None

6. Validate the database connection and environment.

   Condition
   None

   ConfigEngine task
   `validate-database validate-database-environment`

7. Stop the portal server.

   Condition
   collation support

   ConfigEngine task
   `stop-portal-server`

8. Transfer the database.

   Condition
   None

   ConfigEngine task
   `database-transfer enable-profiles-check-managed package-profiles`

9. Grant privileges to the database runtime users.

   Condition
   None

   ConfigEngine task
   `grant-runtime-db-user-privileges`

10. Manual Step: Download the script and run it on the database server to grant privileges to database runtime users.

    Condition
    Select to manually create users and assign them permissions.

    ConfigEngine task
    None

11. Start the portal server.

    Condition
    None

    ConfigEngine task
    `start-portal-server`

What to do next

You transferred your data from Apache Derby to your preferred database.

One quick way to test your database configuration is to log in and explore the site to validate the site is working as you expected.

Go to http://*host_name*:*port*/*context_root*/*default_portal_home*. For example, go to http://*host_name:10039*/*wps*/*portal*.

Next, you can use other options to configure your environment more.

If you are setting up a stand-alone server environment, you can use the Enable Federated Security option to add an LDAP user registry.

If you are setting up a cluster environment, you can use the Create a Deployment Manager option to create a deployment manager profile that is augmented with WebSphere Portal Express resources.

SQL Server: SQL Server worksheet: Transfer to a single database

When you use the database transfer option, you can select the condition to create a single database in the Configuration Wizard. This worksheet highlights the fields and properties that you see in the Configuration Wizard when you select the single database condition.

Typical fields

When you use the database transfer option, you answer questions about your environment. Some fields are required. Other fields are required or removed based on your selections for environment conditions.

The following table lists the typical fields that display when you select the option to transfer your data to a single database. To see additional fields that apply to an advanced configuration, click **Advanced**.

Table 62. Transfer to a single database..

You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Condition	Your Value
Database name	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbName` properties.		
Data source	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DataSourceName` properties.		
Database URL	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbUrl` properties.		
Database home directory			
Configuration user ID	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbUser` properties.		

Table 62. Transfer to a single database. (continued).

You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Condition	Your Value
Configuration password	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbPassword` properties.		
Database administrator ID	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DBA.DbUser` properties.		
Database administrator password	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DBA.DbPassword` properties.		
Runtime user	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbRuntimeUser` properties.	To see this field, continue to use the default selection of **Yes** for needing a runtime database user for day-to-day operations.	
Runtime password	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbRuntimeDbRuntimePassword` properties.		
Release Admin URL			
Community Admin URL			
Customization Admin URL			
JCR Admin URL			
Feedback Admin URL			
Likeminds Admin URL			
Microsoft SQL Server library	`sqlserver2005.DbLibrary`		

SQL Server: SQL Server worksheet: Transfer to multiple databases

When you use the database transfer option, you can use the condition to create multiple databases in the Configuration Wizard. This condition is selected by default. This worksheet highlights the fields and properties that you see in the Configuration Wizard when you use the multiple databases condition.

Typical fields

When you use the database transfer option, you answer questions about your environment. Some fields are required. Other fields are required or removed based on your selections for environment conditions.

The following table lists the typical fields that display when you select the condition to transfer your data to multiple databases. To see additional fields that apply to an advanced configuration, click **Advanced**.

Table 63. Transfer to a multiple database.

You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Conditions	Your Value
Release database name	**release.DbName**		
Release data source	**release.DataSourceName**		
Release database URL	**release.DbUrl**		
Release Admin URL	**release.AdminUrl**		
Release database home directory	**release.DbHome**		
Community database name	**community.DbName**		
Community data source	**community.DataSourceName**		
Community database URL	**community.DbUrl**		
Community Admin URL	**community.AdminDbUrl**		
Community database home directory	**community.DbHome**		
Customization database name	**customization.DbName**		
Customization data source	**customization.DataSourceName**		
Customization database URL	**customization.DbUrl**		
Customization Admin URL	**customization.AdminUrl**		
Customization database home directory	**customization.DbHome**		
JCR database name	**JCR.DbName**		
JCR data source	**JCR.DataSourceName**		
JCR database URL	**JCR.DbUrl**		
JCR Admin URL	**JCR.AdminUrl**		
JCR database home directory	**JCR.DbHome**		
Feedback database name	**feedback.DbName**		
Feedback data source	**feedback.DataSourceName**		
Feedback database URL	**feedback.DbUrl**		
Feedback Admin URL	**feedback.AdminUrl**		
Feedback database home directory	**feedback.DbHome**		
Likeminds database name	**likeminds.DbName**		
Likeminds data source	**likeminds.DataSourceName**		
Likeminds database URL	**likeminds.DbUrl**		
Likeminds Admin URL	**likeminds.AdminUrl**		

Table 63. Transfer to a multiple database (continued).

You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Conditions	Your Value
Likeminds database home directory	**likeminds.DbHome**		
Microsoft SQL Server library	**sqlserver2005.DbLibrary**		
Configuration user ID	The value that you enter is copied to fields in the Advanced view for the *dbdomain*.**DbUser** properties.	To see this field, continue to use the **Yes** selection for using the same user ID and passwords across portal database domains.	
Configuration password	The value that you enter is copied to fields in the Advanced view for the *dbdomain*.**DbPassword** properties.		
Database administrator ID	The value that you enter is copied to fields in the Advanced view for the *dbdomain*.**DBA.DbUser** properties.		
Database administrator password	The value that you enter is copied to fields in the Advanced view for the *dbdomain*.**DBA.DbPassword** properties.		

Table 63. Transfer to a multiple database (continued).

You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Conditions	Your Value
Release configuration user	**release.DbUser**	To see this field, select **No** for using the same user ID and passwords across portal database domains.	
Release configuration password	**release.DbPassword**		
Release database administrator	**release.DBA.DbUser**		
Release database administrator password	**release.DBA.DbPassword**		
Community configuration user	**community.DbUser**		
Community configuration password	**community.DbPassword**		
Community database administrator	**community.DBA.DbUser**		
Community database administrator password	**community.DBA.DbPassword**		
Customization configuration user	**customization.DbUser**		
Customization configuration password	**customization.DbPassword**		
Customization database administrator	**customization.DBA.DbUser**		
Customization database administrator password	**customization.DBA.DbPassword**		
JCR configuration user	**jcr.DbUser**		
JCR configuration password	**jcr.DbPassword**		
JCR database administrator	**jcr.DBA.DbUser**		
JCR database administrator password	**jcr.DBA.DbPassword**		
Feedback configuration user	**feedback.DbUser**		
Feedback configuration password	**feedback.DbPassword**		
Feedback database administrator	**feedback.DBA.DbUser**		
Feedback database administrator password	**feedback.DBA.DbPassword**		
Likeminds configuration user	**likeminds.DbUser**		
Likeminds configuration password	**likeminds.DbPassword**		
Likeminds database administrator	**likeminds.DBA.DbUser**		
Likeminds database administrator password	**likeminds.DBA.DbPassword**		

Table 63. Transfer to a multiple database (continued).

You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Conditions	Your Value
Runtime user	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbRuntimeUser` properties.	To see this field: • Continue to use the **Yes** selection for using the same user ID and passwords across portal database domains. • Continue to use the default selection of **Yes** for needing a runtime database user for day-to-day operations.	
Runtime password	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbRuntimeDbRuntimePassword` properties.		
Release runtime user	`release.DbRuntimeUser`	To see this field: • Select **No** for using the same user ID and passwords across portal database domains. • Continue to use the default selection of **Yes** for needing a runtime database user for day-to-day operations	
Release runtime password	`release.DbRuntimePassword`		
Community runtime user	`community.DbRuntimeUser`		
Community runtime password	`community.DbRuntimePassword`		
Customization runtime user	`customization.DbRuntimeUser`		
Customization runtime password	`customization.DbRuntimePassword`		
JCR runtime user	`jcr.DbRuntimeUser`		
JCR runtime password	`jcr.DbRuntimePassword`		
Feedback runtime user	`feedback.DbRuntimeUser`		
Feedback runtime password	`feedback.DbRuntimePassword`		
Likeminds runtime user	`likeminds.DbRuntimeUser`		
Likeminds runtime password	`likeminds.DbRuntimePassword`		

Oracle: Database transfer

Your portal is installed with an Apache Derby database. The database that is immediately available for use is good for demonstration and portlet and theme development environments. Otherwise, you must configure portal to use a production-level database. Use the Configuration Wizard to transfer the data and configure a new database server.

Configuration Wizard

The primary Configuration Wizard options are based on your target configuration topology, such as a stand-alone server or a cluster. The database transfer option is included with both **Set Up a Stand-alone Server** and **Set Up a Cluster**. For the stand-alone server topology, run the database transfer option before you run the

federated security option. For the cluster option, run the database transfer before you create your deployment manager profile.

For the **Do you want to transfer to one database or multiple databases or schemas** option, select **Single database** to configure one database with a single database user and a single data source. Select **Multiple databases or schemas** to configure multiple databases with different data sources and different users per schema. You can also use the multiple database option to configure a single database with different data sources and different users.

Validation

Prevent a possible database transfer failure by validating your entries in the wizard. When you chose to validate settings for the Database Transfer option, field syntax and database connection validations are performed to prevent a possible failure before you run the configuration.

Syntax validation checks that you:

- Enter a valid port number in the range of 1 - 65535.
- Use the correct format for your host name.
- Enter a valid context root value

The database connection validation checks that a connection can be made to your database server. The wizard can connect to your database server to validate the host name and port number values that you enter.

On the Database Settings panel, you can choose to turn validation on or off by selecting **Yes** or **No** to the **Connect to database server to validate settings** option. Select **No**, if you know that your parameters are correct and your database server is unavailable at the time of creating your instructions.

"Oracle: Oracle worksheet: Transfer your database" on page 498
This worksheet highlights the fields and properties that you see in the Configuration Wizard when you use the transfer database option.

"Oracle: Oracle worksheet: Transfer to multiple databases" on page 500
When you use the database transfer option, you can use the condition to create multiple databases in the Configuration Wizard. This condition is selected by default. This worksheet highlights the fields and properties that you see in the Configuration Wizard when you use the multiple databases condition.

Transferring data to a different database server

Depending on your environment, the wizard generates a configuration process. The following steps reflect all possible steps in the configuration process. The steps do not represent a literal configuration. The steps are provided as a reference.

About this task

If you click **View Step Command**, you can see the task and properties associated with each step in the wizard.

Procedure

1. Back up the properties files that the wizard uses during the configuration.

   Condition
   None

ConfigEngine task
backup-property-files-for-dbxfer

2. Manual Step: Create your Oracle database.

   Condition
   None

   ConfigEngine task
   None

3. Manual Step: Create the data directory, `data`, and the index directory, `index`, on your database server.

   Condition
   Select to manually create users and assign them permissions.

   ConfigEngine task
   None

4. Set up your database.

   (Automatic Storage Management Users only): If you have configured your database with Automatic Storage Management, you must perform additional manual instructions after you perform this step. Run the setup database script to create database schemas and users and grant privileges to database users . Then, go to "Oracle: Creating JCR table spaces (Automatic Storage Management)" on page 554 to perform additional manual instructions.

   Condition
   None

   ConfigEngine task
   setup-database

5. Manual Step: Download the script and run it on the database server to set up your database.

   (Automatic Storage Management Users only): If you have configured your database with Automatic Storage Management, you must edit the script that you download for your environment before you run it on the database server.

   Condition
   Select to manually create users and assign them permissions.

   ConfigEngine task
   None

6. Validate the database connection and environment.

   Condition
   None

   ConfigEngine task
   validate-database validate-database-environment

7. Stop the portal server.

   Condition
   collation support

   ConfigEngine task
   stop-portal-server

8. Transfer the database.

   Condition
   None

ConfigEngine task
database-transfer enable-profiles-check-managed package-profiles

9. Grant privileges to the database runtime users.

   Condition
   None

   ConfigEngine task
   grant-runtime-db-user-privileges

10. Manual Step: Download the script and run it to grant the database runtime user the appropriate privileges to work with database tables.

    Condition
    Select to manually create users and assign them permissions.

    ConfigEngine task
    None

11. Manual Step: Improve database response time for your database that contains the JCR domain.

    Condition
    None

    ConfigEngine task
    None

12. Start the portal server.

    Condition
    None

    ConfigEngine task
    start-portal-server

What to do next

You transferred your data from Apache Derby to your preferred database.

One quick way to test your database configuration is to log in and explore the site to validate the site is working as you expected.

Go to http://*host_name*:*port*/*context_root*/*default_portal_home*. For example, go to http://*host_name:10039*/*wps*/*portal*.

Next, you can use other options to configure your environment more.

If you are setting up a stand-alone server environment, you can use the Enable Federated Security option to add an LDAP user registry.

If you are setting up a cluster environment, you can use the Create a Deployment Manager option to create a deployment manager profile that is augmented with WebSphere Portal Express resources.

Oracle: Oracle worksheet: Transfer your database

This worksheet highlights the fields and properties that you see in the Configuration Wizard when you use the transfer database option.

Typical fields

When you use the database transfer option, you answer questions about your environment. Some fields are required. Other fields are required or removed based on your selections for environment conditions.

The following table lists the required fields that display when you select the database transfer option. To see additional fields that apply to an advanced configuration, click **Advanced**.

Important: To successfully complete the database transfer, the Oracle Thin Type-4 JDBC is required for Linux.

Table 64. Transfer database.

You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Condition	Your Value
Database name	This field corresponds to the *dbdomain*`.DbName` properties in the `wkplc_dbdomain.properties` file.		
Data source	This field corresponds to the *dbdomain*`.DataSourceName` properties in the `wkplc_dbdomain`.properties file.		
Database URL	This field corresponds to the *dbdomain*`.DbUrl` properties in the `wkplc_dbdomain.properties` file.		
Database home directory			
Configuration user ID	This field corresponds to the *dbdomain*`.DbUser` properties in the `wkplc_dbdomain.properties` file.		
Configuration password	This field corresponds to the *dbdomain*`.DbPassword` properties in the `wkplc_dbdomain.properties` file.		
Database administrator ID	This field corresponds to the *dbdomain*`.DBA.DbUser` properties in the `wkplc_dbdomain.properties` file.		
Database administrator password	This field corresponds to the *dbdomain*`.DBA.DbPassword` properties in the `wkplc_dbdomain.properties` file.		

Table 64. Transfer database (continued).

You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Condition	Your Value
Runtime user	This field corresponds to the *dbdomain*`.DbRuntimeUser` properties in the `wkplc_dbdomain.properties` file.	To see this field, continue to use the default selection of **Yes** for needing a runtime database user for day-to-day operations.	
Runtime password	This field corresponds to the *dbdomain*`.DbRuntimeDbRuntimePassword` properties in the `wkplc_dbtype.properties` file.		
Oracle Database library	`oracle.DbLibrary`		

Oracle: Oracle worksheet: Transfer to multiple databases

CF06

When you use the database transfer option, you can use the condition to create multiple databases in the Configuration Wizard. This condition is selected by default. This worksheet highlights the fields and properties that you see in the Configuration Wizard when you use the multiple databases condition.

Typical fields

When you use the database transfer option, you answer questions about your environment. Some fields are required. Other fields are required or removed based on your selections for environment conditions.

The following table lists the typical fields that display when you select the condition to transfer your data to multiple databases. To see additional fields that apply to an advanced configuration, click **Advanced**.

Table 65. Transfer to a multiple database.

You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Conditions	Your Value
Release database name	`release.DbName`		
Release data source	`release.DataSourceName`		
Release database URL	`release.DbUrl`		
Release Admin URL	`release.AdminUrl`		
Release database home directory	`release.DbHome`		
Community database name	`community.DbName`		
Community data source	`community.DataSourceName`		
Community database URL	`community.DbUrl`		

Table 65. Transfer to a multiple database (continued).

You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Conditions	Your Value
Community Admin URL	**community.AdminDbUrl**		
Community database home directory	**community.DbHome**		
Customization database name	**customization.DbName**		
Customization data source	**customization.DataSourceName**		
Customization database URL	**customization.DbUrl**		
Customization Admin URL	**customization.AdminUrl**		
Customization database home directory	**customization.DbHome**		
JCR database name	**JCR.DbName**		
JCR data source	**JCR.DataSourceName**		
JCR database URL	**JCR.DbUrl**		
JCR Admin URL	**JCR.AdminUrl**		
JCR database home directory	**JCR.DbHome**		
Feedback database name	**feedback.DbName**		
Feedback data source	**feedback.DataSourceName**		
Feedback database URL	**feedback.DbUrl**		
Feedback Admin URL	**feedback.AdminUrl**		
Feedback database home directory	**feedback.DbHome**		
Likeminds database name	**likeminds.DbName**		
Likeminds data source	**likeminds.DataSourceName**		
Likeminds database URL	**likeminds.DbUrl**		
Likeminds Admin URL	**likeminds.AdminUrl**		
Likeminds database home directory	**likeminds.DbHome**		
Oracle library	**oracle.DbLibrary**		

Table 65. Transfer to a multiple database (continued).

You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Conditions	Your Value
Configuration user ID	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbUser` properties.	To see this field, continue to use the **Yes** selection for using the same user ID and passwords across portal database domains.	
Configuration password	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbPassword` properties.		
Database administrator ID	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DBA.DbUser` properties.		
Database administrator password	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DBA.DbPassword` properties.		

Table 65. Transfer to a multiple database (continued).

You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Conditions	Your Value
Release configuration user	**release.DbUser**	To see this field, select **No** for using the same user ID and passwords across portal database domains.	
Release configuration password	**release.DbPassword**		
Release database administrator	**release.DBA.DbUser**		
Release database administrator password	**release.DBA.DbPassword**		
Community configuration user	**community.DbUser**		
Community configuration password	**community.DbPassword**		
Community database administrator	**community.DBA.DbUser**		
Community database administrator password	**community.DBA.DbPassword**		
Customization configuration user	**customization.DbUser**		
Customization configuration password	**customization.DbPassword**		
Customization database administrator	**customization.DBA.DbUser**		
Customization database administrator password	**customization.DBA.DbPassword**		
JCR configuration user	**jcr.DbUser**		
JCR configuration password	**jcr.DbPassword**		
JCR database administrator	**jcr.DBA.DbUser**		
JCR database administrator password	**jcr.DBA.DbPassword**		
Feedback configuration user	**feedback.DbUser**		
Feedback configuration password	**feedback.DbPassword**		
Feedback database administrator	**feedback.DBA.DbUser**		
Feedback database administrator password	**feedback.DBA.DbPassword**		
Likeminds configuration user	**likeminds.DbUser**		
Likeminds configuration password	**likeminds.DbPassword**		
Likeminds database administrator	**likeminds.DBA.DbUser**		
Likeminds database administrator password	**likeminds.DBA.DbPassword**		

Table 65. Transfer to a multiple database (continued).

You might not see all of the fields listed in this table. Use the Conditions column in this table to find out what selections are required to see the fields. If you do not see conditions listed for a field, the field always appears. A selection might result in multiple fields displaying in the wizard. For this type of selection, a selection listed in the Conditions column spans multiple field label rows.

Field Label	Property	Conditions	Your Value
Runtime user	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbRuntimeUser` properties.	To see this field: • Continue to use the **Yes** selection for using the same user ID and passwords across portal database domains. • Continue to use the default selection of **Yes** for needing a runtime database user for day-to-day operations.	
Runtime password	The value that you enter is copied to fields in the Advanced view for the *dbdomain*`.DbRuntimeDbRuntimePassword` properties.		
Release runtime user	`release.DbRuntimeUser`	To see this field: • Select **No** for using the same user ID and passwords across portal database domains. • Continue to use the default selection of **Yes** for needing a runtime database user for day-to-day operations	
Release runtime password	`release.DbRuntimePassword`		
Community runtime user	`community.DbRuntimeUser`		
Community runtime password	`community.DbRuntimePassword`		
Customization runtime user	`customization.DbRuntimeUser`		
Customization runtime password	`customization.DbRuntimePassword`		
JCR runtime user	`jcr.DbRuntimeUser`		
JCR runtime password	`jcr.DbRuntimePassword`		
Feedback runtime user	`feedback.DbRuntimeUser`		
Feedback runtime password	`feedback.DbRuntimePassword`		
Likeminds runtime user	`likeminds.DbRuntimeUser`		
Likeminds runtime password	`likeminds.DbRuntimePassword`		

Manual Steps: Database Transfer option in the Configuration Wizard

The database transfer option in the Configuration Wizard includes manual steps. Some manual steps include scripts for you or your database administrator to run. For reference only, you can see the manual steps in this section of the product documentation. Use the Configuration Wizard to complete the deployment configuration. The wizard generates specific instructions and scripts based on your unique input.

"Database transfer: Create database users and groups" on page 511
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the

wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Create database runtime users and groups" on page 512
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Create database users and groups for DB2 for i" on page 512
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database Transfer: Granting privileges to database users for DB2 for i" on page 514
Configuration and runtime database users are granted a different set of privileges, depending on whether these users are schema owners or not. You can create a copy of the SQL scripts and edit this copy to manually grant permissions to configuration and runtime database users.

"Database transfer: Create database configuration users and groups for DB2 for z/OS" on page 517
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Download the script and view instructions to delete existing databases for DB2 for z/OS" on page 518
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Download the script and create the database on DB2 for z/OS server" on page 519
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Download the script and reset the check pending status on portal table spaces for DB2 for z/OS" on page 521
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the

wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Create DB2 database" on page 521
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Download the script and setup your DB2 database " on page 523
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Download the script and grant permissions for the DB2 database users" on page 525
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Enable the DB2 pureScale load balancing feature" on page 531
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Download a script and create your MS SQL database" on page 532
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Download the script and setup your MS SQL database" on page 532
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Download the script and grant permissions for the MS SQL database users" on page 535
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your

deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Create Oracle database" on page 540
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Download the script and setup your Oracle database" on page 542
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Download the script and grant permissions for the Oracle database users" on page 544
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Set up the data and index directory for JCR collation " on page 551
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Set up JCR collation" on page 551
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Improve database response time for your database domains" on page 553
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Create database users and groups" on page 511
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the

wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Create database runtime users and groups" on page 512
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Create database users and groups for DB2 for i" on page 512
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database Transfer: Granting privileges to database users for DB2 for i" on page 514
Configuration and runtime database users are granted a different set of privileges, depending on whether these users are schema owners or not. You can create a copy of the SQL scripts and edit this copy to manually grant permissions to configuration and runtime database users.

"Database transfer: Create database configuration users and groups for DB2 for z/OS" on page 517
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Download the script and view instructions to delete existing databases for DB2 for z/OS" on page 518
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Download the script and create the database on DB2 for z/OS server" on page 519
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Download the script and reset the check pending status on portal table spaces for DB2 for z/OS" on page 521
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the

wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Create DB2 database" on page 521
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Download the script and setup your DB2 database " on page 523
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Download the script and grant permissions for the DB2 database users" on page 525
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Enable the DB2 pureScale load balancing feature" on page 531
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Download a script and create your MS SQL database" on page 532
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Download the script and setup your MS SQL database" on page 532
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Download the script and grant permissions for the MS SQL database users" on page 535
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your

deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Create Oracle database" on page 540
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Download the script and setup your Oracle database" on page 542
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Download the script and grant permissions for the Oracle database users" on page 544
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Set up the data and index directory for JCR collation " on page 551
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Set up JCR collation" on page 551
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

"Database transfer: Improve database response time for your database domains" on page 553
Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database,

migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Database transfer: Create database users and groups

Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

About this task

Attention: All steps for all database environments are included without consideration for your environment. Use the Configuration Wizard to generate custom instructions for your environment.

User and group names must comply with both the database management system software requirements and WebSphere Portal Express requirements.

The procedure is repeated for each portal database domain: release, community customization, jcr, feedback, and likeminds. The variable *dbdomain* represents the actual portal database domain name. You must also create group for each role. Configuration

Procedure

1. Create the database configuration users for the database domains on the operating system.
2. Create the *dbdomain* database configuration user. Use *dbdomain*.DbUser as the user ID when you create this user.
3. Create the database configuration groups for the database domains on the operating system where your database is installed. See your operating system instructions for details on creating groups.
4. Create the *dbdomain* database configuration group. Use *dbdomain*.DbConfigRoleName as the group name on your operating system.
5. Assign your database configuration users to the database configuration groups that you created. See your operating system instructions for details on adding users to groups.
6. Assign *dbdomain*.DbUser to the *dbdomain*.DbConfigRoleName.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Database transfer: Create database runtime users and groups

Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

About this task

Attention: All steps for all database environments are included without consideration for your environment. Use the Configuration Wizard to generate custom instructions for your environment.

User and group names must comply with both the database management system software requirements and WebSphere Portal Express requirements.

Procedure

1. Create the database runtime users for the database domains on the operating system.
2. Create the *dbdomain* database runtime user. Use *dbdomain*.DbRuntimeUser as the user ID when you create this user.
3. Create the *dbdomain* database runtime group. Use *dbdomain*.DbConfigRoleName as the group name on your operating system.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Database transfer: Create database users and groups for DB2 for i

Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

About this task

Attention: All steps for all database environments are included without consideration for your environment. Use the Configuration Wizard to generate custom instructions for your environment.

User and group names must comply with both the database management system software requirements and WebSphere Portal Express requirements.

The procedure is repeated for each portal database domain: release, community customization, jcr, feedback, and likeminds. The variable *dbdomain* represents the actual portal database domain name. You must also create group for each role. Configuration

Procedure

Create the database users and groups.
Example of instructions that the Configuration Wizard generates.

```
Create the database user profiles for the database domains on the operating system where
IBM DB2 for i is installed. The wizard refers to the database user profile as the database configu

The database user profile (database configuration user) must at least have *IOSYSCFG and *JOBCTL s
with user class as *USER on the server where IBM DB2 for i is installed.
Use a different value for your database user profile than used to install WebSphere Portal. The ac
profile might have more authority than is required and usually belongs to an individual. Ensure th
group names comply with both the database management system software requirements and WebSphere Po

Create the database configuration users for the database domains on the operating system.

    Create the release database configuration user. Use config_ID as the user ID when you create t

    Create the community database configuration user. Use config_ID as the user ID when you create

    Create the customization database configuration user. Use config_ID as the user ID when you cr

    Create the jcr database configuration user. Use config_ID as the user ID when you create this

    Create the feedback database configuration user. Use config_ID as the user ID when you create

    Create the likeminds database configuration user. Use config_ID as the user ID when you create

Create the database configuration groups for the database domains on the operating system where IB
installed. See your operating system instructions for details on creating groups.

    Create the release database configuration group. Use WPBASCFG as the group name on your operat

    Create the community database configuration group. Use WPBASCFG as the group name on your oper

    Create the customization database configuration group. Use WPBASCFG as the group name on your

    Create the jcr database configuration group. Use WPJCRCFG as the group name on your operating

    Create the feedback database configuration group. Use WPPZNCFG as the group name on your opera

    Create the likeminds database configuration group. Use WPPZNCFG as the group name on your oper

Assign your database user profile users (database configuration users) to the database configurati
you created. See your operating system instructions for details on adding users to groups.

Create the database runtime users for the database domains on the operating system. Note: The data
does not need any special authority with user class as *USER on the IBM i server where the DB2 is

    Create the release database runtime user. Use runtime_ID as the user ID when you create this u

    Create the community database runtime user. Use runtime_ID as the user ID when you create this

    Create the customization database runtime user. Use runtime_ID as the user ID when you create

    Create the jcr database runtime user. Use runtime_ID as the user ID when you create this user.

    Create the feedback database runtime user. Use runtime_ID as the user ID when you create this

    Create the likeminds database runtime user. Use runtime_ID as the user ID when you create this
```

```
Create the database runtime groups for the database domains on the operating system where DB2 is ins
See your operating system instructions for details on creating groups.

    Create the release database runtime group. Use WPBASRT as the group name on your operating syste

    Create the community database runtime group. Use WPBASRT as the group name on your operating sys

    Create the customization database runtime group. Use WPBASRT as the group name on your operating

    Create the jcr database runtime group. Use WPJCRRT as the group name on your operating system.

    Create the feedback database runtime group. Use WPPZNRT as the group name on your operating syst

    Create the likeminds database runtime group. Use WPPZNRT as the group name on your operating sys

Assign your database runtime users to the database runtime groups that you created. See your operati
system instructions for details on adding users to groups.
```

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Database Transfer: Granting privileges to database users for DB2 for i

Configuration and runtime database users are granted a different set of privileges, depending on whether these users are schema owners or not. You can create a copy of the SQL scripts and edit this copy to manually grant permissions to configuration and runtime database users.

Note: The wizard refers to the database user profile as the database configuration user.

Required privileges of the configuration database user

When a configuration database user (database user profile) is a schema owner, the **domain.DbUser** property is assigned the same value as the **domain.DbSchema** property, and a role is created for a configuration user in each database domain. This role is created and assigned automatically when you create your database using the configuration wizard or when you run the `create-database` configuration task.

To learn more about the specific permissions granted to the configuration database user, navigate to the SQL script templates in the installation directory of IBM WebSphere Portal Express. These read-only templates should not be modified. To grant these privileges, you can create a copy of the SQL scripts and use this copy to grant permissions manually.

Refer to the following locations of the SQL script templates to learn more about the specific permissions granted to the schema-owning configuration database user:

Table 66. Location of SQL script templates by database domain for information about specific permissions granted to schema-owning configuration database users

Database domain	Location of template
Release	*PortalServer_root*/base/wp.db.impl/config/templates/setupdb/db2_iseries/release/createConfigRoleForSameSchema.sql
Community	*PortalServer_root*/base/wp.db.impl/config/templates/setupdb/db2_iseries/community/createConfigRoleForSameSchema.sql
Customization	*PortalServer_root*/base/wp.db.impl/config/templates/setupdb/db2_iseries/customization/createConfigRoleForSameSchema.sql
JCR	*PortalServer_root*/base/wp.db.impl/config/templates/setupdb/db2_iseries/jcr/createConfigRoleForSameSchema.sql
Feedback	*PortalServer_root*/pzn/prereq.pzn/config/templates/setupdb/db2_iseries/feedback/createConfigRoleForSameSchema.sql
Likeminds	*PortalServer_root*/pzn/prereq.pzn/config/templates/setupdb/db2_iseries/likeminds/createConfigRoleForSameSchema.sql

Refer to the following locations of the SQL script templates for non-schema-owning configuration database user:

Table 67. Location of SQL script templates by database domain for information about specific permissions granted to non-schema-owning configuration database users

Database domain	Location of template
Release	*PortalServer_root*/base/wp.db.impl/config/templates/setupdb/db2_iseries/release/createConfigRoleForDifferentSchema.sql
Community	*PortalServer_root*/base/wp.db.impl/config/templates/setupdb/db2_iseries/community/createConfigRoleForDifferentSchema.sql
Customization	*PortalServer_root*/base/wp.db.impl/config/templates/setupdb/db2_iseries/customization/createConfigRoleForDifferentSchema.sql
JCR	*PortalServer_root*/base/wp.db.impl/config/templates/setupdb/db2_iseries/jcr/createConfigRoleForDifferentSchema.sql
Feedback	*PortalServer_root*/pzn/prereq.pzn/config/templates/setupdb/db2_iseries/feedback/createConfigRoleForDifferentSchema.sql
Likeminds	*PortalServer_root*/pzn/prereq.pzn/config/templates/setupdb/db2_iseries/likeminds/createConfigRoleForDifferentSchema.sql

Required privileges for the runtime database user

When the runtime database user is a schema owner, the **domain.DbUser** property is assigned the same value as the properties **domain.DbRuntimeUser** and **domain.DbSchema**. The runtime database user typically does not create tables used to query and manipulate data and does not by default have access to these tables. To grant minimum privileges to a runtime database user to work with these tables, access needs to be provided for the objects individually. A role is created for runtime database users in each database domain. These roles are created and assigned automatically when you run the following configuration tasks:

```
create-database
grant-runtime-db-user-privileges
```

Before you run these configuration tasks, the runtime database user can only access the database to validate configurations. To learn more about the specific permissions granted to the runtime database user, navigate to the SQL script

templates in the installation directory of WebSphere Portal Express. These read-only templates should not be modified. To grant these privileges, you can create a copy of the SQL scripts and use this copy to grant permissions manually.

Refer to the following locations of the SQL script templates to learn more about the specific permissions granted to the schema-owning configuration database user:

Table 68. Location of SQL script templates by database domain for information about specific permissions granted to schema-owning configuration runtime database users

Database domain	Location of template
Release	*PortalServer_root*/base/wp.db.impl/config/templates/setupdb/db2_iseries/release/createRuntimeRoleForSameSchema.sql
Community	*PortalServer_root*/base/wp.db.impl/config/templates/setupdb/db2_iseries/community/createRuntimeRoleForSameSchema.sql
Customization	*PortalServer_root*/base/wp.db.impl/config/templates/setupdb/db2_iseries/customization/createRuntimeRoleForSameSchema.sql
JCR	*PortalServer_root*/base/wp.db.impl/config/templates/setupdb/db2_iseries/jcr/createRuntimeRoleForSameSchema.sql *PortalServer_root*/jcr/wp.content.repository.install/config/templates/setupdb/db2_iseries/jcr/grantPermissionsToRuntimeRoleStatic.sql
Feedback	*PortalServer_root*/pzn/prereq.pzn/config/templates/setupdb/db2_iseries/feedback/createRuntimeRoleForSameSchema.sql
Likeminds	*PortalServer_root*/pzn/prereq.pzn/config/templates/setupdb/db2_iseries/likeminds/createRuntimeRoleForSameSchema.sql

Refer to the following locations of the SQL script templates to learn more about the specific permissions granted to the non-schema-owning runtime database user:

Table 69. Location of SQL script templates by database domain for information about specific permissions granted to non-schema-owning runtime database users

Database domain	Location of template
Release	*PortalServer_root*base/wp.db.impl/config/templates/setupdb/db2_iseries/release/createRuntimeRoleForDifferentSchema.sql
Community	*PortalServer_root*/base/wp.db.impl/config/templates/setupdb/db2_iseries/community/createRuntimeRoleForDifferentSchema.sql
Customization	*PortalServer_root*/base/wp.db.impl/config/templates/setupdb/db2_iseries/customization/createRuntimeRoleForDifferentSchema.sql
JCR	*PortalServer_root*/base/wp.db.impl/config/templates/setupdb/db2_iseries/jcr/createRuntimeRoleForDifferentSchema.sql *PortalServer_root*/jcr/wp.content.repository.install/config/templates/setupdb/db2_iseries/jcr/grantPermissionsToRuntimeRole.sql
Feedback	*PortalServer_root*/pzn/prereq.pzn/config/templates/setupdb/db2_iseries/feedback/createRuntimeRoleForDifferentSchema.sql
Likeminds	*PortalServer_root*/pzn/prereq.pzn/config/templates/setupdb/db2_iseries/likeminds/createRuntimeRoleForDifferentSchema.sql

Related concepts:

"Database users" on page 121
There are two types of database users: database configuration users and database runtime users. Become familiar with the privileges required for each user type to work with the database domains of IBM WebSphere Portal Express and the commands for creating database configuration users and granting privileges.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Database transfer: Create database configuration users and groups for DB2 for z/OS

Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

About this task

Attention: All steps for all database environments are included without consideration for your environment. Use the Configuration Wizard to generate custom instructions for your environment.

Procedure

1. Download the script. This script is contained in a compressed file. Extract the file.
2. Review the RACF commands that are contained in this script with your RACF administrator. Edit these commands as needed. For example, the RACF administrator might need to remove the ADDUSER commands if you already have database user IDs that are defined in your RACF.
3. If you have a security product other than RACF, you must change the syntax in the RACF commands into the appropriate syntax for your security product.
4. Run the RACF commands on z/OS in the order that is listed in the script.

   Example of the script that the Configuration Wizard generates.

```
//*  RACF setup for database user IDs and schemas:
ADDGROUP WPBASRT;
ADDGROUP WPJCRRT;
ADDGROUP WPPZNRT;
ADDUSER runtime_ID DFLTGRP(WPBASRT) NAME('WAS DB2 ACCESS USER')
PW USER(runtime_ID) NOINTERVAL
ALU runtime_ID PASSWORD(runtime_pwd) NOEXPIRED
CONNECT runtime_ID GROUP(WPJCRRT)
CONNECT runtime_ID GROUP(WPPZNRT)
ADDGROUP jcr SUPGROUP(group)
CONNECT dbconfig_ID GROUP(jcr)
CONNECT runtime_ID GROUP(jcr)
```

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Database transfer: Download the script and view instructions to delete existing databases for DB2 for z/OS

Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

About this task

Attention: All steps for all database environments are included without consideration for your environment. Use the Configuration Wizard to generate custom instructions for your environment.

Procedure

1. Copy the downloaded script to the database server and extract it.
2. Use your preferred SQL processor, such as SPUFI, to run the script on the database server. Run the DB2 commands on z/OS in the order that is listed in the script.

   Note: When you run this script when no other database exists, you might receive a message that this script is unsuccessful. You can ignore this message.

   Example of the script that the Configuration Wizard generates.

```
SET CURRENT SQLID = 'dbadmin';
REVOKE CREATEIN, DROPIN ON SCHEMA jcr
FROM dbconfig_ID;
REVOKE SELECT ON SYSIBM.SYSTABLESPACE FROM dbconfig_ID;
REVOKE SELECT ON SYSIBM.SYSVIEWS FROM dbconfig_ID;
REVOKE SELECT ON SYSIBM.SYSDUMMY1 FROM dbconfig_ID;
REVOKE SELECT ON SYSIBM.SYSTRIGGERS FROM dbconfig_ID;
REVOKE SELECT ON SYSIBM.SYSINDEXPART FROM dbconfig_ID;
REVOKE SELECT ON SYSIBM.SYSINDEXES FROM dbconfig_ID;
REVOKE SELECT ON SYSIBM.SYSSYNONYMS FROM dbconfig_ID;
REVOKE SELECT ON SYSIBM.SYSDATABASE FROM dbconfig_ID;
COMMIT;
DROP TABLESPACE WPFDBK.fdbkdbts;
COMMIT;
DROP TABLESPACE WPLM.lmdbts;
COMMIT;
DROP DATABASE WPREL;
COMMIT;
DROP DATABASE WPCOMM;
COMMIT;
DROP DATABASE WPCUST;
COMMIT;
DROP DATABASE WPJCR;
```

```
COMMIT;
DROP DATABASE WPFDBK;
COMMIT;
DROP DATABASE WPLM;
COMMIT;
DROP STOGROUP WPSSG;
COMMIT;
REVOKE SELECT ON SYSIBM.SYSCOLUMNS FROM dbconfig_ID;
REVOKE SELECT ON SYSIBM.SYSTABLES FROM dbconfig_ID;
REVOKE SELECT ON SYSIBM.SYSRELS FROM dbconfig_ID;
REVOKE SELECT ON SYSIBM.SYSFOREIGNKEYS FROM dbconfig_ID;
COMMIT;
REVOKE USE OF BUFFERPOOL BP2 FROM dbconfig_ID;
COMMIT;
REVOKE USE OF BUFFERPOOL BP32K1 FROM dbconfig_ID;
COMMIT;
REVOKE USE OF BUFFERPOOL BP3 FROM dbconfig_ID;
COMMIT;
```

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Database transfer: Download the script and create the database on DB2 for z/OS server

Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

About this task

Attention: All steps for all database environments are included without consideration for your environment. Use the Configuration Wizard to generate custom instructions for your environment.

Procedure

1. Copy the downloaded script to the database server and extract it.
2. Use your preferred SQL processor, such as SPUFI, to run the script on the database server. Run the DB2 commands on z/OS in the order that is listed in the script.

   Example of the script that the Configuration Wizard generates.

```
SET CURRENT SQLID = 'dbadmin';
CREATE STOGROUP WPSSG
       VOLUMES('*')
       VCAT DSN910;
COMMIT;
CREATE DATABASE WPREL
       STOGROUP WPSSG
       BUFFERPOOL BP2
       INDEXBP BP3
       CCSID UNICODE;
COMMIT;
```

```
CREATE DATABASE WPCOMM
       STOGROUP WPSSG
       BUFFERPOOL BP2
       INDEXBP BP3
       CCSID UNICODE;
COMMIT;
CREATE DATABASE WPCUST
       STOGROUP WPSSG
       BUFFERPOOL BP2
       INDEXBP BP3
       CCSID UNICODE;
COMMIT;
CREATE DATABASE WPJCR
       STOGROUP WPSSG
       BUFFERPOOL BP2
       INDEXBP BP3
       CCSID UNICODE;
COMMIT;
CREATE DATABASE WPFDBK
       STOGROUP WPSSG
       BUFFERPOOL BP2
       CCSID UNICODE;
COMMIT;
CREATE DATABASE WPLM
       STOGROUP WPSSG
       BUFFERPOOL BP2
       CCSID UNICODE;
COMMIT;
GRANT CREATEIN, DROPIN ON SCHEMA jcr
TO dbconfig_ID;
GRANT SELECT ON SYSIBM.SYSTABLESPACE TO dbconfig_ID;
GRANT SELECT ON SYSIBM.SYSVIEWS TO dbconfig_ID;
GRANT SELECT ON SYSIBM.SYSDUMMY1 TO dbconfig_ID;
GRANT SELECT ON SYSIBM.SYSTRIGGERS TO dbconfig_ID;
GRANT SELECT ON SYSIBM.SYSINDEXPART TO dbconfig_ID;
GRANT SELECT ON SYSIBM.SYSINDEXES TO dbconfig_ID;
GRANT SELECT ON SYSIBM.SYSSYNONYMS TO dbconfig_ID;
GRANT SELECT ON SYSIBM.SYSDATABASE TO dbconfig_ID;
COMMIT;
CREATE TABLESPACE fdbkdbts IN WPFDBK
       BUFFERPOOL BP2
       LOCKSIZE ROW
       SEGSIZE 4
       DEFINE NO;
COMMIT;
CREATE TABLESPACE lmdbts IN WPLM
       BUFFERPOOL BP2
       LOCKSIZE ROW
       SEGSIZE 4
       DEFINE NO;
COMMIT;
GRANT SELECT ON SYSIBM.SYSCOLUMNS TO dbconfig_ID;
GRANT SELECT ON SYSIBM.SYSTABLES TO dbconfig_ID;
GRANT SELECT ON SYSIBM.SYSRELS TO dbconfig_ID;
GRANT SELECT ON SYSIBM.SYSFOREIGNKEYS TO dbconfig_ID;
COMMIT;
GRANT DBADM ON DATABASE WPREL
TO dbconfig_ID WITH GRANT OPTION;
COMMIT;
GRANT DBADM ON DATABASE WPCOMM
TO dbconfig_ID WITH GRANT OPTION;
COMMIT;
GRANT DBADM ON DATABASE WPCUST
TO dbconfig_ID WITH GRANT OPTION;
COMMIT;
GRANT DBADM ON DATABASE WPJCR
TO dbconfig_ID WITH GRANT OPTION;
COMMIT;
GRANT DBADM ON DATABASE WPFDBK
TO dbconfig_ID WITH GRANT OPTION;
COMMIT;
GRANT DBADM ON DATABASE WPLM
TO dbconfig_ID WITH GRANT OPTION;
COMMIT;
GRANT USE OF STOGROUP WPSSG
TO dbconfig_ID;
COMMIT;
GRANT USE OF BUFFERPOOL BP2 TO dbconfig_ID;
COMMIT;
GRANT USE OF BUFFERPOOL BP32K1 TO dbconfig_ID;
COMMIT;
GRANT USE OF BUFFERPOOL BP3 TO dbconfig_ID;
COMMIT;
```

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Database transfer: Download the script and reset the check pending status on portal table spaces for DB2 for z/OS

Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

About this task

Attention: All steps for all database environments are included without consideration for your environment. Use the Configuration Wizard to generate custom instructions for your environment.

Procedure

1. Copy the downloaded script to the database server and extract it.
2. Review the DB2 utilities that are contained in this script with your database administrator. Edit these utilities as needed.
3. Run the DB2 utilities that are listed in this script.
4. Use the SQL SELECT query that is created after you run the CHECK DATA commands to confirm that all of the CHECK PENDING states are resolved.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Database transfer: Create DB2 database

Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

Before you begin

Ensure your DB2 server is already installed and running.

About this task

Attention: All steps for all database environments are included without consideration for your environment. Use the Configuration Wizard to generate custom instructions for your environment.

Procedure

Instructions for AIX, HP-UX, Linux, and Solaris

1. Copy the JDBC Driver JAR files from your database server to your portal server.

   Example location:

   `/opt/ibm/db2/V10.5/java/db2jcc4.jar`

   `/opt/ibm/db2/V10.5/java/db2jcc_license_cu.jar`

   Place the files on the portal server in the directory that you specified for database library files.

   db2.DbLibrary

2. Log in as a DB2 system administrator with the required database privileges (SYSADM or at a minimum SYSCTRL).

   `su - domain.DBA.DbUser`

3. Copy the script that you downloaded for this step to the database server, and extract the file.
4. Rename the file that you downloaded to include a `.sh` extension.
5. Run the shell file on your database server.

   /your_database_server_file_location/`CreateDB2Database.sh`

   Example of the script that the Configuration Wizard generates.

```
db2set DB2COMM=TCPIP
db2set DB2_EVALUNCOMMITTED=YES
db2set DB2_INLIST_TO_NLJN=YES
db2 "UPDATE DBM CFG USING sheapthres 0"


db2 "CREATE DB WPSDB using codeset UTF-8 territory us PAGESIZE 8192"
db2 "UPDATE DB CFG FOR WPSDB USING locktimeout 30"

db2 "CONNECT TO WPSDB USER dba_ID USING dba_pwd"
db2 "CREATE BUFFERPOOL ICMLSFREQBP4 SIZE 1000 AUTOMATIC PAGESIZE 4K"
db2 "CREATE BUFFERPOOL ICMLSVOLATILEBP4 SIZE 16000 AUTOMATIC PAGESIZE 4K"
db2 "CREATE BUFFERPOOL ICMLSMAINBP32 SIZE 16000 AUTOMATIC PAGESIZE 32K"
db2 "CREATE BUFFERPOOL CMBMAIN4 SIZE 1000 AUTOMATIC PAGESIZE 4K"
db2 "CREATE REGULAR TABLESPACE ICMLFQ32 PAGESIZE 32K BUFFERPOOL ICMLSMAINBP32"
db2 "CREATE REGULAR TABLESPACE ICMLNF32 PAGESIZE 32K BUFFERPOOL ICMLSMAINBP32"
db2 "CREATE REGULAR TABLESPACE ICMVFQ04 PAGESIZE 4K BUFFERPOOL ICMLSVOLATILEBP4"
db2 "CREATE REGULAR TABLESPACE ICMSFQ04 PAGESIZE 4K BUFFERPOOL ICMLSFREQBP4"
db2 "CREATE REGULAR TABLESPACE CMBINV04 PAGESIZE 4K BUFFERPOOL CMBMAIN4"
db2 "CREATE SYSTEM TEMPORARY TABLESPACE ICMLSSYSTSPACE32 PAGESIZE 32K BUFFERPOOL ICMLSMAINBP32"
db2 "CREATE SYSTEM TEMPORARY TABLESPACE ICMLSSYSTSPACE4 PAGESIZE 4K BUFFERPOOL ICMLSVOLATILEBP4"
db2 "CREATE USER TEMPORARY TABLESPACE ICMLSUSRTSPACE4 PAGESIZE 4K BUFFERPOOL ICMLSVOLATILEBP4"
db2 "DISCONNECT WPSDB"
db2 "TERMINATE"
db2 "UPDATE DB CFG FOR WPSDB USING logfilsiz 16000"
db2 "UPDATE DB CFG FOR WPSDB USING logprimary 20"
db2 "UPDATE DB CFG FOR WPSDB USING logsecond 50"
db2 "UPDATE DB CFG FOR WPSDB USING logbufsz 500"
db2 "UPDATE DB CFG FOR WPSDB USING DFT_QUERYOPT 5"
```

Instructions for Windows

6. Copy the JDBC Driver JAR files from your database server to your portal server.

   Example location:

   `C:\\Program Files\\IBM\\SQLLIB\\java\\db2jcc4.jar`

   `C:\\Program Files\\IBM\\SQLLIB\\java\\db2jcc_license_cu.jar`

Place the files on the portal server in the directory that you specified for database library files.

db2.DbLibrary

7. Log in as a DB2 system administrator. This DB2 system administrator must have the required database privileges (SYSADM or at a minimum SYSCTRL) to create your database. This user is typically a member of both the Administrators and DB2ADMNS groups.
8. Copy the script that you downloaded for this step to the database server, and extract the file.
9. Rename the file that you downloaded to include a `.bat` extension.
10. Run the batch file on your database server.

    ```
    db2cmdadmin.exe -w C:\\your_database_server_file_location\\
    CreateDB2Database.bat
    ```

    Example of the script is not available for Windows.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Database transfer: Download the script and setup your DB2 database

Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

About this task

Attention: All steps for all database environments are included without consideration for your environment. Use the Configuration Wizard to generate custom instructions for your environment.

Procedure

1. Copy the downloaded script to the database server and extract it.
2. Run the SQL statements that are listed in this script in your SQL processor. db2 -tvf SetupDB2Database.sql

```
CONNECT TO WPSDB USER dba_ID USING dba_pwd;
CREATE SCHEMA release AUTHORIZATION configID;
COMMIT;

GRANT CONNECT, CREATETAB ON DATABASE TO GROUP WP_BASE_CONFIG_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSH200 TO GROUP WP_BASE_CONFIG_USERS;
GRANT ALTERIN, CREATEIN, DROPIN ON SCHEMA release TO GROUP WP_BASE_CONFIG_USERS;
GRANT USE OF TABLESPACE USERSPACE1 TO GROUP WP_BASE_CONFIG_USERS;
GRANT USAGE ON WORKLOAD SYSDEFAULTUSERWORKLOAD TO GROUP WP_BASE_CONFIG_USERS;



GRANT CONNECT ON DATABASE TO GROUP WP_BASE_RUNTIME_USERS;
```

```
GRANT EXECUTE ON PACKAGE NULLID.SYSSN100 TO GROUP WP_BASE_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN200 TO GROUP WP_BASE_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN300 TO GROUP WP_BASE_RUNTIME_USERS;
GRANT USAGE ON WORKLOAD SYSDEFAULTUSERWORKLOAD TO GROUP WP_BASE_RUNTIME_USERS;

CONNECT RESET;

CONNECT TO WPSDB USER dba_ID USING dba_pwd;
CREATE SCHEMA community AUTHORIZATION configID;
COMMIT;

GRANT CONNECT, CREATETAB ON DATABASE TO GROUP WP_BASE_CONFIG_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSH200 TO GROUP WP_BASE_CONFIG_USERS;
GRANT ALTERIN, CREATEIN, DROPIN ON SCHEMA community TO GROUP WP_BASE_CONFIG_USERS;
GRANT USE OF TABLESPACE USERSPACE1 TO GROUP WP_BASE_CONFIG_USERS;
GRANT USAGE ON WORKLOAD SYSDEFAULTUSERWORKLOAD TO GROUP WP_BASE_CONFIG_USERS;



GRANT CONNECT ON DATABASE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN100 TO GROUP WP_BASE_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN200 TO GROUP WP_BASE_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN300 TO GROUP WP_BASE_RUNTIME_USERS;
GRANT USAGE ON WORKLOAD SYSDEFAULTUSERWORKLOAD TO GROUP WP_BASE_RUNTIME_USERS;

CONNECT RESET;

CONNECT TO WPSDB USER dba_ID USING dba_pwd;
CREATE SCHEMA customization AUTHORIZATION configID;
COMMIT;

GRANT CONNECT, CREATETAB ON DATABASE TO GROUP WP_BASE_CONFIG_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSH200 TO GROUP WP_BASE_CONFIG_USERS;
GRANT ALTERIN, CREATEIN, DROPIN ON SCHEMA customization TO GROUP WP_BASE_CONFIG_USERS;
GRANT USE OF TABLESPACE USERSPACE1 TO GROUP WP_BASE_CONFIG_USERS;
GRANT USAGE ON WORKLOAD SYSDEFAULTUSERWORKLOAD TO GROUP WP_BASE_CONFIG_USERS;



GRANT CONNECT ON DATABASE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN100 TO GROUP WP_BASE_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN200 TO GROUP WP_BASE_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN300 TO GROUP WP_BASE_RUNTIME_USERS;
GRANT USAGE ON WORKLOAD SYSDEFAULTUSERWORKLOAD TO GROUP WP_BASE_RUNTIME_USERS;

CONNECT RESET;

CONNECT TO WPSDB USER dba_ID USING dba_pwd;
CREATE SCHEMA jcr AUTHORIZATION configID;
COMMIT;

GRANT CONNECT, CREATETAB ON DATABASE TO GROUP WP_JCR_CONFIG_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSH200 TO GROUP WP_JCR_CONFIG_USERS;
GRANT ALTERIN, CREATEIN, DROPIN ON SCHEMA jcr TO GROUP WP_JCR_CONFIG_USERS;
GRANT USE OF TABLESPACE USERSPACE1 TO GROUP WP_JCR_CONFIG_USERS;
GRANT USAGE ON WORKLOAD SYSDEFAULTUSERWORKLOAD TO GROUP WP_JCR_CONFIG_USERS;


GRANT USE OF TABLESPACE ICMLFQ32 TO GROUP WP_JCR_CONFIG_USERS;
GRANT USE OF TABLESPACE ICMLNF32 TO GROUP WP_JCR_CONFIG_USERS;
GRANT USE OF TABLESPACE ICMVFQ04 TO GROUP WP_JCR_CONFIG_USERS;
GRANT USE OF TABLESPACE ICMSFQ04 TO GROUP WP_JCR_CONFIG_USERS;
GRANT USE OF TABLESPACE CMBINV04 TO GROUP WP_JCR_CONFIG_USERS;
GRANT USE OF TABLESPACE ICMLSUSRTSPACE4 TO GROUP WP_JCR_CONFIG_USERS;


GRANT CONNECT ON DATABASE TO GROUP WP_JCR_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN100 TO GROUP WP_JCR_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN200 TO GROUP WP_JCR_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN300 TO GROUP WP_JCR_RUNTIME_USERS;
GRANT USAGE ON WORKLOAD SYSDEFAULTUSERWORKLOAD TO GROUP WP_JCR_RUNTIME_USERS;

CONNECT RESET;

CONNECT TO WPSDB USER dba_ID USING dba_pwd;
CREATE SCHEMA feedback AUTHORIZATION configID;
COMMIT;

GRANT CONNECT, CREATETAB ON DATABASE TO GROUP WP_PZN_CONFIG_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSH200 TO GROUP WP_PZN_CONFIG_USERS;
GRANT ALTERIN, CREATEIN, DROPIN ON SCHEMA feedback TO GROUP WP_PZN_CONFIG_USERS;
GRANT USE OF TABLESPACE USERSPACE1 TO GROUP WP_PZN_CONFIG_USERS;
GRANT USAGE ON WORKLOAD SYSDEFAULTUSERWORKLOAD TO GROUP WP_PZN_CONFIG_USERS;



GRANT CONNECT ON DATABASE TO GROUP WP_PZN_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN100 TO GROUP WP_PZN_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN200 TO GROUP WP_PZN_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN300 TO GROUP WP_PZN_RUNTIME_USERS;
GRANT USAGE ON WORKLOAD SYSDEFAULTUSERWORKLOAD TO GROUP WP_PZN_RUNTIME_USERS;

CONNECT RESET;

CONNECT TO WPSDB USER dba_ID USING dba–pwd;
```

```
CREATE SCHEMA likeminds AUTHORIZATION configID;
COMMIT;

GRANT CONNECT, CREATETAB ON DATABASE TO GROUP WP_PZN_CONFIG_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSH200 TO GROUP WP_PZN_CONFIG_USERS;
GRANT ALTERIN, CREATEIN, DROPIN ON SCHEMA likeminds TO GROUP WP_PZN_CONFIG_USERS;
GRANT USE OF TABLESPACE USERSPACE1 TO GROUP WP_PZN_CONFIG_USERS;
GRANT USAGE ON WORKLOAD SYSDEFAULTUSERWORKLOAD TO GROUP WP_PZN_CONFIG_USERS;



GRANT CONNECT ON DATABASE TO GROUP WP_PZN_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN100 TO GROUP WP_PZN_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN200 TO GROUP WP_PZN_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN300 TO GROUP WP_PZN_RUNTIME_USERS;
GRANT USAGE ON WORKLOAD SYSDEFAULTUSERWORKLOAD TO GROUP WP_PZN_RUNTIME_USERS;

CONNECT RESET;
```

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Database transfer: Download the script and grant permissions for the DB2 database users

Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

About this task

Attention: All steps for all database environments are included without consideration for your environment. Use the Configuration Wizard to generate custom instructions for your environment.

Procedure

1. Copy the downloaded script to the database server and extract it.
2. Run the SQL statements that are listed in this script in your SQL processor. `db2 -tvf SetupDB2Database.sql`

   Example of the script that is generated by the Configuration Wizard.

```
CONNECT TO WPSDB USER dba_ID USING dba_pwd;
GRANT CONNECT ON DATABASE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN100 TO GROUP WP_BASE_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN200 TO GROUP WP_BASE_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN300 TO GROUP WP_BASE_RUNTIME_USERS;
GRANT USAGE ON WORKLOAD SYSDEFAULTUSERWORKLOAD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.ACL TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.ACL_ENTRY TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.ACTION_DESC TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.ACTION_DESC_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.ACTION_SET TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.ACTION_SET_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.APP_DESC TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.APP_DESC_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.APP_DESC_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.AUTH_LEVEL TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.CLIENT_DESC TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.CLIENT_DESC_CAPS TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.COMP_INST TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.COMP_INST_DD TO GROUP WP_BASE_RUNTIME_USERS;
```

```
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.COMP_INST_DRAFT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.COMP_INST_DRAFT_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.COMP_INST_DRAFT_MAD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.COMP_INST_MAD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.CONTENT_MAPPING TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.CONTENT_MAPPING_DRAFT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.CP_FILTER_DATA TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.CP_FILTER_INST TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.CRED_SEGMENT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.CRED_SLOT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.CRED_SLOT_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.DEVICE_CLASS_DESC TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.DEVICE_CLASS_DESC_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.EVT_HANDLER TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.EVT_INTERFACE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.LANG_DESC TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.LANG_DESC_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.LNK_PAGE_DC TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.LNK_PAGE_DC_DRAFT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.LNK_PAGE_PORT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.LNK_PAGE_PORT_DRAFT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.LNK_RES_TYPE_AS TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.LNK_RES_TYPE_RES TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.LNK_RESTYPE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.LNK_THEME_SKIN TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.LNK_USER_ROLE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.LNK_USER_ROLE_DRAFT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.MARKUP_DESC TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.MARKUP_DESC_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.OUTBOUND_CONFIG TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.OUTBOUND_CONFIG_VAR TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.OUTBOUND_COOKIE_RULE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.OUTBOUND_COOKIE_RULE_CN TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.OUTBOUND_COOKIE_RULE_TR TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.OUTBOUND_MAPPING TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.OUTBOUND_MAPPING_IP TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.OUTBOUND_MAPPING_MD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.OUTBOUND_POLICY TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.OUTBOUND_POLICY_AM TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.OUTBOUND_POLICY_FC TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.OUTBOUND_POLICY_HD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.OUTBOUND_POLICY_MD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.OUTBOUND_POLICY_MT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PAGE_ACT_ASC TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PAGE_ACT_ASC_DRAFT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PAGE_ACT_ASC_DRAFT_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PAGE_ACT_ASC_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PAGE_INST TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PAGE_INST_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PAGE_INST_DRAFT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PAGE_INST_DRAFT_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PAGE_INST_DRAFT_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PAGE_INST_DRAFT_MAD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PAGE_INST_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PAGE_INST_MAD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PAGE_LAYOUT_DESC TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PAGE_LAYOUT_DESC_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PAGE_LAYOUT_DESC_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PAGE_LAYOUT_DESC_MAD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PCM_DIALOG TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PCM_DIALOG_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PCM_DIALOG_MD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PCM_TRANSITION TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PCM_TRANSITION_ENDPOINT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PCM_TRANSITION_ENDPOINT_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PCM_TRANSITION_ENDPOINT_MD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PCM_TRANSITION_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PCM_TRANSITION_MD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_ACT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_ACT_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_ACT_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_ACT_MD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_DESC TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_DESC_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_DESC_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_DESC_PREF TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_INST TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_INST_DRAFT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_INST_DRAFT_PREF TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_INST_PREF TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_PARM TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_PARM_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_PROP TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_PROP_ALIAS TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_PROP_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_WIRE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_WIRE_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_WIRE_DRAFT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_WIRE_DRAFT_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_WIRE_DRAFT_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_WIRE_DRAFT_MD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_WIRE_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORT_WIRE_MD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORTAL_URL TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PORTAL_URL_DD TO GROUP WP_BASE_RUNTIME_USERS;
```

```
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PROT_RES TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PROT_RES_DEP TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PROT_RES_DRAFT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PSE_CATEGORY TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PSE_CATEGORY_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PSE_CATEGORY_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PSE_DESC TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PSE_DESC_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PSE_SOURCE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PSE_SOURCE_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PSE_SOURCE_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PUB_REND_PARM TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PUB_REND_PARM_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.PUB_REND_PARM_QN TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.RES_TYPE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.RES_TYPE_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.ROLE_INST TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.ROLE_INST_DRAFT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.SEARCH_REG TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.SEARCH_REG_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.SERV_DESC TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.SERV_DESC_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.SERV_DESC_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.SERV_DESC_MAD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.SERV_DESC_PREF TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.SERVER_INFO TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.SERVER_INFO_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.SERVER_INFO_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.SKIN_DESC TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.SKIN_DESC_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.SKIN_DESC_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.SKIN_DESC_MAD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.THEME_DESC TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.THEME_DESC_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.THEME_DESC_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.THEME_DESC_MAD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.UNIQUE_NAME TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.UNIQUE_NAME_DRAFT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.URL_MAP_CTX TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.URL_MAP_CTX_LBLS TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.USER_DESC TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.USER_DESC_STATE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.VANITY_URL TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.VANITY_URL_DRAFT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.VAULT_DATA TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.VAULT_RESOURCES TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.VP_DESC TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.VP_DESC_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.WEB_MOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.WEB_MOD_UATTR TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.WF_DATA TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.WF_DEF TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.WF_DEF_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.WF_TP_MAPPING TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.WF_TP_MAPPING_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.WP_SEARCH_REG TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.WPS_SCDLMGR TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.WPS_SCDLMPR TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.WPS_SCDTASK TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.WPS_SCDTREG TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.WSRP_PROD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.WSRP_PROD_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.WSRP_PROD_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.WSRP_PROD_PORT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE release.WSRP_PROD_UATTR TO GROUP WP_BASE_RUNTIME_USERS;


CONNECT RESET;
CONNECT TO WPSDB USER dba_ID USING dba_pwd;
GRANT CONNECT ON DATABASE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN100 TO GROUP WP_BASE_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN200 TO GROUP WP_BASE_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN300 TO GROUP WP_BASE_RUNTIME_USERS;
GRANT USAGE ON WORKLOAD SYSDEFAULTUSERWORKLOAD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.ACL TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.ACL_ENTRY TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.AUTH_LEVEL TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.CATA TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.CATA_ENTRY TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.CATA_ENTRY_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.CATA_ENTRY_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.CATA_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.CATE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.CATE_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.COMP_INST TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.COMP_INST_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.COMP_INST_MAD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.CONTACTS_GROUP TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.CONTACTS_MEMBER TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.CONTACTS_MEMBER_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.CONTENT_MAPPING TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.CP_RESOURCE_RATING TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.CP_RESOURCE_TAG TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.CRED_SEGMENT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.CRED_SLOT TO GROUP WP_BASE_RUNTIME_USERS;
```

```
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.CRED_SLOT_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.LNK_MEMBER_GROUP TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.LNK_USER_ROLE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PAGE_ACT_ASC TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PAGE_ACT_ASC_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PAGE_INST TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PAGE_INST_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PAGE_INST_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PAGE_INST_MAD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PORT_ACT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PORT_ACT_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PORT_ACT_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PORT_ACT_MD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PORT_INST TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PORT_INST_PREF TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PORT_PARM TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PORT_PARM_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PORT_PROP TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PORT_PROP_ALIAS TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PORT_PROP_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PORT_WIRE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PORT_WIRE_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PORT_WIRE_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PORT_WIRE_MD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PROT_RES TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.PROT_RES_DEP TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.ROLE_INST TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.UNIQUE_NAME TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.VAULT_DATA TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE customization.VAULT_RESOURCES TO GROUP WP_BASE_RUNTIME_USERS;


CONNECT RESET;
CONNECT TO WPSDB USER dba_ID USING dba_pwd;
GRANT CONNECT ON DATABASE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN100 TO GROUP WP_BASE_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN200 TO GROUP WP_BASE_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN300 TO GROUP WP_BASE_RUNTIME_USERS;
GRANT USAGE ON WORKLOAD SYSDEFAULTUSERWORKLOAD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.ACL TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.ACL_ENTRY TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AI_APP TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AI_APP_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AI_APP_METADATA TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AI_APP_WIRE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AI_BC TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AI_BC_META_WIRE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AI_BC_METADATA TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AI_BC_VAR TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AI_BC_WIRE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AI_CATEGORY TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AI_CATEGORY_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AI_PARM TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AI_PARM_BCMD_WIRE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AI_PARM_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AI_RSTR_PT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AI_TEMP_METADATA TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AI_TEMPLATE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AI_TEMPLATE_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AI_TEMPLATE_ZIP TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AI_TFOLDER TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AI_TFOLDER_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AI_WIRE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.APP_DESC TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.APP_DESC_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.APP_DESC_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.AUTH_LEVEL TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.COMP_INST TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.COMP_INST_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.COMP_INST_MAD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.CONTENT_MAPPING TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.CP_CATEGORY TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.CP_CUSTOMRESOURCE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.CP_RESOURCE_RATING TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.CP_RESOURCE_TAG TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.CP_RESOURCE_UPDATED TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.HASHED_BLOB TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.LNK_USER_ROLE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.LOC_DATA TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.LOC_DATA_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PAGE_ACT_ASC TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PAGE_ACT_ASC_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PAGE_INST TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PAGE_INST_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PAGE_INST_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PAGE_INST_MAD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PORT_ACT TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PORT_ACT_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PORT_ACT_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PORT_ACT_MD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PORT_DESC TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PORT_DESC_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PORT_DESC_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PORT_DESC_PREF TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PORT_INST TO GROUP WP_BASE_RUNTIME_USERS;
```

```
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PORT_INST_PREF TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PORT_PARM TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PORT_PARM_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PORT_PROP TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PORT_PROP_ALIAS TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PORT_PROP_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PORT_WIRE TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PORT_WIRE_DD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PORT_WIRE_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PORT_WIRE_MD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PROT_RES TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.PROT_RES_DEP TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.ROLE_INST TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.UNIQUE_NAME TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.WF_DATA TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.WF_DEF TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.WF_DEF_LOD TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.WF_TP_MAPPING TO GROUP WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE community.WF_TP_MAPPING_DD TO GROUP WP_BASE_RUNTIME_USERS;


CONNECT RESET;
CONNECT TO WPSDB USER dba_ID USING dba_pwd;
GRANT CONNECT ON DATABASE TO GROUP WP_JCR_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN100 TO GROUP WP_JCR_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN200 TO GROUP WP_JCR_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN300 TO GROUP WP_JCR_RUNTIME_USERS;
GRANT USAGE ON WORKLOAD SYSDEFAULTUSERWORKLOAD TO GROUP WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE jcr.ACL TO GROUP WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE jcr.ACL_ENTRY TO GROUP WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE jcr.CONTENT_MAPPING TO GROUP WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE jcr.LNK_USER_ROLE TO GROUP WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE jcr.PROT_RES TO GROUP WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE jcr.PROT_RES_DEP TO GROUP WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE jcr.ROLE_INST TO GROUP WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON TABLE jcr.UNIQUE_NAME TO GROUP WP_JCR_RUNTIME_USERS;


CONNECT RESET;
CONNECT TO WPSDB USER dba_ID USING dba_pwd;
GRANT CONNECT ON DATABASE TO GROUP WP_PZN_RUNTIME_USERS;
GRANT DBADM ON DATABASE TO GROUP WP_PZN_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN100 TO GROUP WP_PZN_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN200 TO GROUP WP_PZN_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN300 TO GROUP WP_PZN_RUNTIME_USERS;
GRANT USAGE ON WORKLOAD SYSDEFAULTUSERWORKLOAD TO GROUP WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.AGGREGATEKEY TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.AGGREGATES TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.AGGREGATESTATUS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.AGGREGATE_CONTENT TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.BROWSERS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.CALENDAR TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.CATEGORIES TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.CATEGORYMAP TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.CATEGORY_PATTERNS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.CATEGORY_SETS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.COOKIESSTATUS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.DOMAIN TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.ENTITIES TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.ENTITYTRAVERSAL TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.HITPARMS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.HIT_FACTS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.HTTPVERSION TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.IMPORTHISTORY TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.JAVASCRIPTSTATUS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.KEY TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.KEY_VALUE_COMBO TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.KEY_VALUE_PAIR TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.LINKAGE TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.LOGCONTROL TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.LOGS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.LOG_FILE_STATUS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.NETWORKS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.PARMS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.PLATFORMS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.PROTOCOLS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.REFERRER TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.REFERRERHOST TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.REFERRERURL TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.RESETSTATUS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.RESOURCES TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.RESULT TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.RETURNCODES TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.SERVERNODES TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.SERVERS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.SESSIONPARMS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.SESSION_FACTS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.SUBDOMAINS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.TIMEOFDAY TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.TIMESPAN TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.USERAGENTS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.USERS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.VALUE TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.WEB_NODES TO WP_PZN_RUNTIME_USERS;
```

```
CONNECT RESET;
CONNECT TO WPSDB USER dba_ID USING dba_pwd;
GRANT CONNECT ON DATABASE TO GROUP WP_PZN_RUNTIME_USERS;
GRANT DBADM ON DATABASE TO GROUP WP_PZN_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN100 TO GROUP WP_PZN_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN200 TO GROUP WP_PZN_RUNTIME_USERS;
GRANT EXECUTE ON PACKAGE NULLID.SYSSN300 TO GROUP WP_PZN_RUNTIME_USERS;
GRANT USAGE ON WORKLOAD SYSDEFAULTUSERWORKLOAD TO GROUP WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_ADMIN_USERS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_CFG TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_ITEM_DATA TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_MBA_SCORED TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_RTG_POOL TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_TRX_MENTOR TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_TRX_POOL TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_TRX_TYPE TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_USER_DATA TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_USER_MENTOR TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_USER_RATING TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_USER_TRX TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_VISIT_LIST TO WP_PZN_RUNTIME_USERS;


CONNECT RESET;
CONNECT TO WPSDB USER dba_ID USING dba_pwd;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_OBJECTS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_TIMERS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.JOBS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.OBJECTS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.TASKJOBMAPPING TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.TASKS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_USER_SHORTCUT TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_CONTROLLABLES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_LIBRARIES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_PROJECTS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_PROJLIBS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_SYNDICATORS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_SYNDLIBS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_VERSIONS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_VPORTALS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_SYND_FAILURE TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_SYND_SESSION TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_SYND_EXTRA_DATA TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.OPENJPA_SEQUENCE_TABLE TO WP_JCR_RUNTIME_USERS;


GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTADMINDOMAINS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTACCESSCODES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTCOLLNAME TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTSYSCONTROL TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTATTRDEFS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTATTRGROUP TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTNLSLANGUAGES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTMAXKEYWORD TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTNLSKEYWORDS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTITEMTYPEDEFS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTITVIEWDEFS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTITVIEWID TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTCOMPDEFS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTCOMPATTRS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTCOMPATTRSFK TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTCOMPVIEWDEFS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTCOMPVIEWATTRS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTITEMS001001 TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTITEMVER001001 TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTLINKS001001 TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTMIMETYPES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTITEMSTODELETE TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTXDOOBJECTS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTRI001001 TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTVIEWACCESS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMUT00600001 TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRNODELOCKS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRWS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRNSURIS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRNSPREFIXES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMUTSWIDE0 TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMUTMWIDE0 TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRWSNODES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRLINKS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRDELTALINKS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRLINKREL TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRDELTALR TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRGLBLPROPS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRADDTLPROPS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRNODETYPES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRPROPDEFS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRNODEDEFS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRRNODETYPES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRSUPERTYPES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRTIMESTAMPS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRIDS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRDUMMY TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRMAXSNSIDX TO WP_JCR_RUNTIME_USERS;
```

```
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRGENDDL TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRINDEXNAMES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRINDEXES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRTSPENDING TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRTSERRORS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRTSINDEXES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRTSPATHINFO TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRTSPOSCOR TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRTSNEGCOR TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRAPPL TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRNODEREGISTER TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRTSFULLCRAWLTOPICS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRTSSUBSCRIPTIONMANAGER TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRTSSEEDLISTPENDING TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRTSINCSUBSMSGS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCREDITIONINFO TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCREDITIONS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRREMOVEHLP TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRIMPORTREF TO WP_JCR_RUNTIME_USERS;


CONNECT RESET;
```

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Database transfer: Enable the DB2 pureScale load balancing feature

Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

About this task

Attention: All steps for all database environments are included without consideration for your environment. Use the Configuration Wizard to generate custom instructions for your environment.

This step applies only to DB2 pureScale.

For multithreaded Java applications, set enableSysplexWLB to true in the connection string to use the transaction level workload balancing feature for DB2 pureScale. As more members are started, clients automatically route to the new member without any interruption of service. Members can be stopped without the application knowing.

Procedure

1. Access the WebSphere Integrated Solutions Console. Go to `http://`*`host_name`*`:`*`your_server_admin_port`*`/ibm/console/`. For example, go to `http://`*`host_name`*`:10038/ibm/console/`.
2. Click **Resources** > **JDBC** > **Data Sources**.
3. Click **Data sources** > **data source name** > **Custom properties**.

4. Search for enableSysplexWLB.
5. Click **enableSysplexWLB**, and enter `true` in the **Value** field.
6. Click **OK**.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Database transfer: Download a script and create your MS SQL database

Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

About this task

Attention: All steps for all database environments are included without consideration for your environment. Use the Configuration Wizard to generate custom instructions for your environment.

Procedure

1. Copy the downloaded script to the database server and extract it.
2. Run the SQL statements that are listed in this script in your SQL processor.

   Example of the script that is created by the Configuration Wizard

```
CREATE DATABASE WPSDB ON (NAME=RELDB_DATA, FILENAME='C:\\Microsoft SQL Server\\instance\\MSSQL/Data/WPSDB_Data.MDF', SIZE=10MB, MAXSIZE=U
LOG ON (NAME=DB_LOG, FILENAME='C:\\Microsoft SQL Server\\instance\\MSSQL/Data/WPSDB_Log.LDF', SIZE=100MB, MAXSIZE=30000MB, FILEGROWTH=100
collate SQL_Latin1_General_CP1_CS_AS
;
ALTER DATABASE WPSDB SET READ_COMMITTED_SNAPSHOT ON;
```

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Database transfer: Download the script and setup your MS SQL database

Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and

steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

About this task

Attention: All steps for all database environments are included without consideration for your environment. Use the Configuration Wizard to generate custom instructions for your environment.

Procedure

1. Copy the downloaded script to the database server and extract it.
2. Run the SQL statements that are listed in this script in your SQL processor. db2 -tvf SetupDB2Database.sql

   Example of the script that the Configuration Wizard generates

```
-- Create the schema
USE [WPSDB];
GO
CREATE SCHEMA release;
GO
-- Create the configuration user
USE [WPSDB];
EXEC sp_addlogin 'config_ID', 'config_pwd', 'WPSDB';
-- Add a role to the user for XA transactions
USE [master];
EXEC sp_addrolemember N'SqlJDBCXAUser', N'config_ID';
GO
-- Add the configuration user to the database
USE [WPSDB];
EXEC sp_grantdbaccess 'config_ID';
GO
-- Create the runtime user
USE [WPSDB];
EXEC sp_addlogin 'runtime_ID', 'run_pwd', 'WPSDB';
-- Add a role to the user for XA transactions
USE [master];
EXEC sp_addrolemember N'SqlJDBCXAUser', N'runtime_ID';
GO
-- Add the runtime user to the database
USE [WPSDB];
EXEC sp_grantdbaccess 'runtime_ID';
GO
-- Create the configuration role
USE [WPSDB];
CREATE ROLE [WP_BASE_CONFIG_USERS];
GRANT CREATE TABLE TO [WP_BASE_CONFIG_USERS];
GRANT ALTER, SELECT, INSERT, UPDATE, DELETE, REFERENCES ON SCHEMA::[release] TO [WP_BASE_CONFIG_USERS];
EXEC sp_addrolemember N'WP_BASE_CONFIG_USERS', N'config_ID';
GO
-- Create the runtime role
CREATE ROLE [WP_BASE_RUNTIME_USERS];
EXEC sp_addrolemember N'WP_BASE_RUNTIME_USERS', N'runtime_ID';
GO
-- Create the schema
USE [WPSDB];
GO
CREATE SCHEMA community;
GO
-- Add the configuration user to the database
USE [WPSDB];
EXEC sp_grantdbaccess 'config_ID';
GO
-- Add the runtime user to the database
USE [WPSDB];
EXEC sp_grantdbaccess 'runtime_ID';
GO
-- Create the configuration role
USE [WPSDB];
GRANT ALTER, SELECT, INSERT, UPDATE, DELETE, REFERENCES ON SCHEMA::[community] TO [WP_BASE_CONFIG_USERS];
EXEC sp_addrolemember N'WP_BASE_CONFIG_USERS', N'config_ID';
GO
-- Create the runtime role
EXEC sp_addrolemember N'WP_BASE_RUNTIME_USERS', N'runtime_ID';
GO
-- Create the schema
USE [WPSDB];
GO
CREATE SCHEMA customization;
GO
-- Add the configuration user to the database
USE [WPSDB];
```

```
EXEC sp_grantdbaccess 'config_ID';
GO
-- Add the runtime user to the database
USE [WPSDB];
EXEC sp_grantdbaccess 'runtime_ID';
GO
-- Create the configuration role
USE [WPSDB];
GRANT ALTER, SELECT, INSERT, UPDATE, DELETE, REFERENCES ON SCHEMA::[customization] TO [WP_BASE_CONFIG_USERS];
EXEC sp_addrolemember N'WP_BASE_CONFIG_USERS', N'config';
GO
-- Create the runtime role
EXEC sp_addrolemember N'WP_BASE_RUNTIME_USERS', N'runtime_ID';
GO
-- Create the schema
USE [WPSDB];
GO
CREATE SCHEMA jcr;
GO
-- Add the configuration user to the database
USE [WPSDB];
EXEC sp_grantdbaccess 'config_ID';
GO
-- Add the runtime user to the database
USE [WPSDB];
EXEC sp_grantdbaccess 'runtime_ID';
GO
-- Create the configuration role
USE [WPSDB];
CREATE ROLE [WP_JCR_CONFIG_USERS];
GRANT CREATE TABLE TO [WP_JCR_CONFIG_USERS];
GRANT CREATE VIEW TO [WP_JCR_CONFIG_USERS];
GRANT ALTER, SELECT, INSERT, UPDATE, DELETE, REFERENCES ON SCHEMA::[jcr] TO [WP_JCR_CONFIG_USERS];
EXEC sp_addrolemember N'WP_JCR_CONFIG_USERS', N'config_ID';
GO
-- Create the runtime role
CREATE ROLE [WP_JCR_RUNTIME_USERS];
EXEC sp_addrolemember N'WP_JCR_RUNTIME_USERS', N'runtime_ID';
GO
-- Create the schema
USE [WPSDB];
GO
CREATE SCHEMA feedback;
GO
-- Add the configuration user to the database
USE [WPSDB];
EXEC sp_grantdbaccess 'config_ID';
GO
-- Add the runtime user to the database
USE [WPSDB];
EXEC sp_grantdbaccess 'runtime_ID';
GO
-- Create the configuration role
USE [WPSDB];
CREATE ROLE [WP_PZN_CONFIG_USERS];
GRANT CREATE TABLE TO [WP_PZN_CONFIG_USERS];
GRANT ALTER, SELECT, INSERT, UPDATE, DELETE, REFERENCES ON SCHEMA::[feedback] TO [WP_PZN_CONFIG_USERS];
EXEC sp_addrolemember N'WP_PZN_CONFIG_USERS', N'config_ID';
GO
-- Create the runtime role
CREATE ROLE [WP_PZN_RUNTIME_USERS];
EXEC sp_addrolemember N'WP_PZN_RUNTIME_USERS', N'runtime_ID';
GO
-- Create the schema
USE [WPSDB];
GO
CREATE SCHEMA likeminds;
GO
-- Add the configuration user to the database
USE [WPSDB];
EXEC sp_grantdbaccess 'config_ID';
GO
-- Add the runtime user to the database
USE [WPSDB];
EXEC sp_grantdbaccess 'runtime_ID';
GO
-- Create the configuration role
USE [WPSDB];
GRANT CREATE PROCEDURE TO [WP_PZN_CONFIG_USERS];
GRANT ALTER, SELECT, INSERT, UPDATE, DELETE, REFERENCES ON SCHEMA::[likeminds] TO [WP_PZN_CONFIG_USERS];
EXEC sp_addrolemember N'WP_PZN_CONFIG_USERS', N'config_ID';
GO
-- Create the runtime role
EXEC sp_addrolemember N'WP_PZN_RUNTIME_USERS', N'runtime_ID';
GO
```

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database,

migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Database transfer: Download the script and grant permissions for the MS SQL database users

Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

About this task

Attention: All steps for all database environments are included without consideration for your environment. Use the Configuration Wizard to generate custom instructions for your environment.

Procedure

1. Copy the downloaded script to the database server and extract it.
2. Run the SQL statements that are listed in this script in your SQL processor. `db2 -tvf SetupDB2Database.sql`

   Example of the script that is generated by the Configuration Wizard.

```
USE [WPSDB];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[ACL] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PROT_RES] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[ACL_ENTRY] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[ROLE_INST] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[LNK_USER_ROLE] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[UNIQUE_NAME] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PROT_RES_DEP] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[CONTENT_MAPPING] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PAGE_INST] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PAGE_INST_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PAGE_INST_MAD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PAGE_INST_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_INST] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_INST_PREF] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[COMP_INST] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[COMP_INST_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[COMP_INST_MAD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_ACT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_ACT_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_ACT_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_ACT_MD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_PROP] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_PROP_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_PROP_ALIAS] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_PARM] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_PARM_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_WIRE] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_WIRE_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_WIRE_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_WIRE_MD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PAGE_ACT_ASC] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PAGE_ACT_ASC_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[AUTH_LEVEL] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[VAULT_RESOURCES] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[VAULT_DATA] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[CRED_SEGMENT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[CRED_SLOT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[CRED_SLOT_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[APP_DESC] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[APP_DESC_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[APP_DESC_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_DESC] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_DESC_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_DESC_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_DESC_PREF] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[WF_DEF] TO [WP_BASE_RUNTIME_USERS];
```

```
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[WF_DEF_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[WF_DATA] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[WF_TP_MAPPING] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[WF_TP_MAPPING_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[WEB_MOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[WEB_MOD_UATTR] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[WSRP_PROD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[WSRP_PROD_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[WSRP_PROD_UATTR] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[WSRP_PROD_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[WSRP_PROD_PORT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[SERV_DESC] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[SERV_DESC_MAD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[SERV_DESC_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[SERV_DESC_PREF] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[SERV_DESC_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PUB_REND_PARM] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PUB_REND_PARM_QN] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PUB_REND_PARM_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[SKIN_DESC] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[SKIN_DESC_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[SKIN_DESC_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[SKIN_DESC_MAD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[THEME_DESC] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[THEME_DESC_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[THEME_DESC_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[THEME_DESC_MAD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PAGE_LAYOUT_DESC] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PAGE_LAYOUT_DESC_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PAGE_LAYOUT_DESC_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PAGE_LAYOUT_DESC_MAD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[SERVER_INFO] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[SERVER_INFO_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[SERVER_INFO_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[USER_DESC] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[USER_DESC_STATE] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[URL_MAP_CTX] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[URL_MAP_CTX_LBLS] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[LNK_PAGE_PORT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[LNK_THEME_SKIN] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[MARKUP_DESC] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[MARKUP_DESC_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[CLIENT_DESC] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[CLIENT_DESC_CAPS] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[LANG_DESC] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[LANG_DESC_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[EVT_HANDLER] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[EVT_INTERFACE] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORTAL_URL] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORTAL_URL_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[VP_DESC] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[VP_DESC_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[ACTION_SET] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[ACTION_SET_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[ACTION_DESC] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[ACTION_DESC_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[RES_TYPE] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[RES_TYPE_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[LNK_RES_TYPE_RES] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[LNK_RES_TYPE_AS] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[LNK_RESTYPE] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PSE_DESC] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PSE_DESC_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PSE_CATEGORY] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PSE_CATEGORY_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PSE_CATEGORY_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PSE_SOURCE] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PSE_SOURCE_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PSE_SOURCE_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[WP_SEARCH_REG] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[SEARCH_REG] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[SEARCH_REG_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[CP_FILTER_INST] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[CP_FILTER_DATA] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[DEVICE_CLASS_DESC] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[LNK_PAGE_DC] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[DEVICE_CLASS_DESC_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[VANITY_URL] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[OUTBOUND_CONFIG] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[OUTBOUND_CONFIG_VAR] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[OUTBOUND_MAPPING] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[OUTBOUND_MAPPING_MD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[OUTBOUND_MAPPING_IP] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[OUTBOUND_POLICY] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[OUTBOUND_POLICY_AM] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[OUTBOUND_POLICY_FC] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[OUTBOUND_POLICY_HD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[OUTBOUND_POLICY_MT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[OUTBOUND_POLICY_MD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[OUTBOUND_COOKIE_RULE] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[OUTBOUND_COOKIE_RULE_CN] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[OUTBOUND_COOKIE_RULE_TR] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PCM_DIALOG] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PCM_DIALOG_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PCM_DIALOG_MD] TO [WP_BASE_RUNTIME_USERS];
```

```
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PCM_TRANSITION_ENDPOINT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PCM_TRANSITION_ENDPOINT_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PCM_TRANSITION_ENDPOINT_MD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PCM_TRANSITION] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PCM_TRANSITION_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PCM_TRANSITION_MD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PAGE_INST_DRAFT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PAGE_INST_DRAFT_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PAGE_INST_DRAFT_MAD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PAGE_INST_DRAFT_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_INST_DRAFT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_INST_DRAFT_PREF] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[COMP_INST_DRAFT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[COMP_INST_DRAFT_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[COMP_INST_DRAFT_MAD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[UNIQUE_NAME_DRAFT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PROT_RES_DRAFT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[ROLE_INST_DRAFT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[LNK_USER_ROLE_DRAFT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[CONTENT_MAPPING_DRAFT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_WIRE_DRAFT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_WIRE_DRAFT_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_WIRE_DRAFT_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PORT_WIRE_DRAFT_MD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PAGE_ACT_ASC_DRAFT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[PAGE_ACT_ASC_DRAFT_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[LNK_PAGE_PORT_DRAFT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[LNK_PAGE_DC_DRAFT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[VANITY_URL_DRAFT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[WPS_SCDTASK] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[WPS_SCDTREG] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[WPS_SCDLMGR] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[release].[WPS_SCDLMPR] TO [WP_BASE_RUNTIME_USERS];


USE [WPSDB];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[ACL] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PROT_RES] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[ACL_ENTRY] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[ROLE_INST] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[LNK_USER_ROLE] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[UNIQUE_NAME] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PROT_RES_DEP] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[CONTENT_MAPPING] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PAGE_INST] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PAGE_INST_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PAGE_INST_MAD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PAGE_INST_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PORT_INST] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PORT_INST_PREF] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[COMP_INST] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[COMP_INST_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[COMP_INST_MAD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PORT_ACT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PORT_ACT_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PORT_ACT_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PORT_ACT_MD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PORT_PROP] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PORT_PROP_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PORT_PROP_ALIAS] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PORT_PARM] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PORT_PARM_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PORT_WIRE] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PORT_WIRE_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PORT_WIRE_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PORT_WIRE_MD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PAGE_ACT_ASC] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[PAGE_ACT_ASC_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[AUTH_LEVEL] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[VAULT_RESOURCES] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[VAULT_DATA] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[CRED_SEGMENT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[CRED_SLOT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[CRED_SLOT_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[CP_RESOURCE_TAG] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[CP_RESOURCE_RATING] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[CONTACTS_MEMBER] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[CONTACTS_MEMBER_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[CONTACTS_GROUP] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[LNK_MEMBER_GROUP] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[CATA] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[CATA_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[CATE] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[CATE_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[CATA_ENTRY] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[CATA_ENTRY_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[customization].[CATA_ENTRY_DD] TO [WP_BASE_RUNTIME_USERS];


USE [WPSDB];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[ACL] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PROT_RES] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[ACL_ENTRY] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[ROLE_INST] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[LNK_USER_ROLE] TO [WP_BASE_RUNTIME_USERS];
```

```
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[UNIQUE_NAME] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PROT_RES_DEP] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[CONTENT_MAPPING] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PAGE_INST] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PAGE_INST_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PAGE_INST_MAD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PAGE_INST_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PORT_INST] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PORT_INST_PREF] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[COMP_INST] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[COMP_INST_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[COMP_INST_MAD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PORT_ACT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PORT_ACT_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PORT_ACT_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PORT_ACT_MD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PORT_PROP] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PORT_PROP_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PORT_PROP_ALIAS] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PORT_PARM] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PORT_PARM_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PORT_WIRE] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PORT_WIRE_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PORT_WIRE_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PORT_WIRE_MD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PAGE_ACT_ASC] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PAGE_ACT_ASC_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AUTH_LEVEL] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[APP_DESC] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[APP_DESC_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[APP_DESC_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PORT_DESC] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PORT_DESC_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PORT_DESC_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[PORT_DESC_PREF] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[WF_DEF] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[WF_DEF_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[WF_DATA] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[WF_TP_MAPPING] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[WF_TP_MAPPING_DD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AI_CATEGORY] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AI_CATEGORY_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AI_TFOLDER] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AI_TFOLDER_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AI_TEMPLATE] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AI_TEMPLATE_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AI_TEMPLATE_ZIP] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AI_TEMP_METADATA] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AI_APP] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AI_APP_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AI_APP_METADATA] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AI_RSTR_PT] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AI_BC] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AI_BC_METADATA] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AI_BC_VAR] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AI_PARM] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AI_PARM_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AI_WIRE] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AI_BC_WIRE] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AI_APP_WIRE] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AI_BC_META_WIRE] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[AI_PARM_BCMD_WIRE] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[CP_CUSTOMRESOURCE] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[CP_CATEGORY] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[LOC_DATA] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[LOC_DATA_LOD] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[HASHED_BLOB] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[CP_RESOURCE_UPDATED] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[CP_RESOURCE_TAG] TO [WP_BASE_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[community].[CP_RESOURCE_RATING] TO [WP_BASE_RUNTIME_USERS];


USE [WPSDB];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ACL] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[PROT_RES] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ACL_ENTRY] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ROLE_INST] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[LNK_USER_ROLE] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[UNIQUE_NAME] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[PROT_RES_DEP] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[CONTENT_MAPPING] TO [WP_JCR_RUNTIME_USERS];


USE [WPSDB];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[AGGREGATE_CONTENT] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[AGGREGATEKEY] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[AGGREGATES] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[AGGREGATESTATUS] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[BROWSERS] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[CALENDAR] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[CATEGORIES] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[CATEGORY_PATTERNS] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[CATEGORY_SETS] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[CATEGORYMAP] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[COOKIESSTATUS] TO [WP_PZN_RUNTIME_USERS];
```

```
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[DOMAIN] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[ENTITIES] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[ENTITYTRAVERSAL] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[HIT_FACTS] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[HITPARMS] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[HTTPVERSION] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[IMPORTHISTORY] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[JAVASCRIPTSTATUS] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[KEY] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[KEY_VALUE_COMBO] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[KEY_VALUE_PAIR] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[LINKAGE] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[LOG_FILE_STATUS] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[LOGCONTROL] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[LOGS] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[NETWORKS] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[PARMS] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[PLATFORMS] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[PROTOCOLS] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[REFERRER] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[REFERRERHOST] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[REFERRERURL] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[RESETSTATUS] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[RESOURCES] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[RESULT] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[RETURNCODES] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[SERVERNODES] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[SERVERS] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[SESSION_FACTS] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[SESSIONPARMS] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[SUBDOMAINS] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[TIMEOFDAY] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[TIMESPAN] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[USERAGENTS] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[USERS] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[VALUE] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[feedback].[WEB_NODES] TO [WP_PZN_RUNTIME_USERS];


USE [WPSDB];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[likeminds].[lps_admin_users] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[likeminds].[lps_cfg] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[likeminds].[lps_item_data] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[likeminds].[lps_mba_scored] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[likeminds].[lps_rtg_pool] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[likeminds].[lps_trx_mentor] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[likeminds].[lps_trx_pool] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[likeminds].[lps_trx_type] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[likeminds].[lps_user_data] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[likeminds].[lps_user_mentor] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[likeminds].[lps_user_rating] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[likeminds].[lps_user_trx] TO [WP_PZN_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[likeminds].[lps_visit_list] TO [WP_PZN_RUNTIME_USERS];

GRANT EXECUTE ON OBJECT::[likeminds].[sp_lpsdeleteitem] TO [WP_PZN_RUNTIME_USERS];
GRANT EXECUTE ON OBJECT::[likeminds].[sp_lpsdeletetype] TO [WP_PZN_RUNTIME_USERS];
GRANT EXECUTE ON OBJECT::[likeminds].[sp_lpsdeleteuser] TO [WP_PZN_RUNTIME_USERS];


USE [WPSDB];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[WCM_OBJECTS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[WCM_TIMERS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[JOBS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[OBJECTS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[TASKJOBMAPPING] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[TASKS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[WCM_USER_SHORTCUT] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[WCM_CONTROLLABLES] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[WCM_LIBRARIES] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[WCM_PROJECTS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[WCM_PROJLIBS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[WCM_SYNDICATORS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[WCM_SYNDLIBS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[WCM_VERSIONS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[WCM_VPORTALS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[WCM_SYND_FAILURE] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[WCM_SYND_SESSION] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[WCM_SYND_EXTRA_DATA] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[OPENJPA_SEQUENCE_TABLE] TO [WP_JCR_RUNTIME_USERS];



USE [WPSDB];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTADMINDOMAINS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTACCESSCODES] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTCOLLNAME] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTSYSCONTROL] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTATTRDEFS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTATTRGROUP] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTNLSLANGUAGES] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTMAXKEYWORD] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTNLSKEYWORDS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTITEMTYPEDEFS] TO [WP_JCR_RUNTIME_USERS];
```

```
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTITVIEWDEFS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTITVIEWID] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTCOMPDEFS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTCOMPATTRS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTCOMPATTRSFK] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTCOMPVIEWDEFS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTCOMPVIEWATTRS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTITEMS001001] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTITEMVER001001] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTLINKS001001] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTMIMETYPES] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTITEMSTODELETE] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTXDOOBJECTS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTRI001001] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTVIEWACCESS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMUT00600001] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRNODELOCKS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRWS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRNSURIS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRNSPREFIXES] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMUTSWIDE0] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMUTMWIDE0] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRWSNODES] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRLINKS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRDELTALINKS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRLINKREL] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRDELTALR] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRGLBLPROPS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRADDTLPROPS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRNODETYPES] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRPROPDEFS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRNODEDEFS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRRNODETYPES] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRSUPERTYPES] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRTIMESTAMPS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRIDS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRDUMMY] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRMAXSNSIDX] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRGENDDL] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRINDEXNAMES] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRINDEXES] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRTSPENDING] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRTSERRORS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRTSINDEXES] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRTSPATHINFO] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRTSPOSCOR] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRTSNEGCOR] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRAPPL] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRNODEREGISTER] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRTSFULLCRAWLTOPICS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRTSSUBSCRIPTIONMANAGER] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRTSSEEDLISTPENDING] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRTSINCSUBSMSGS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCREDITIONINFO] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCREDITIONS] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRREMOVEHLP] TO [WP_JCR_RUNTIME_USERS];
GRANT SELECT, INSERT, UPDATE, DELETE ON OBJECT::[jcr].[ICMSTJCRIMPORTREF] TO [WP_JCR_RUNTIME_USERS];
```

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Database transfer: Create Oracle database

Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

About this task

Attention: All steps for all database environments are included without consideration for your environment. Use the Configuration Wizard to generate custom instructions for your environment.

Procedure

1. See the Oracle product documentation for instructions on creating databases.
2. All databases must be created using Unicode Database and National character sets such as UTF8, AL32UTF8, or AL16UTF16.
3. It is recommended that all databases to be used with WebSphere Portal Express are configured in Dedicated Server Mode.
4. If you are using Oracle 11g databases, you must configure database transfer and runtime with only the `ojdbc6.jar`.
5. If you are using Oracle 12c databases, you must configure database transfer and runtime with the `ojdbc7.jar` and `xdb6.jar`.
6. You must set the buffer pools allocated to the Oracle database in order for WebSphere Portal Express to communicate with the Java Content Repository database. Refer to the Oracle product documentation for information on how to set the buffer pools. Use these recommended buffer pool values as a guide for setting your values:

   ```
   db_block_size = 8192 bytes
   db_cache_size = 1 gigabyte
   db_files = 1024 files
   log_buffer = 65536 bytes
   open_cursors = 1500 cursors
   pga_aggregate_target = 200 megabytes
   pre_page_sga = true
   processes = 300 processes
   shared_pool_size = 200 megabytes
   ```
7. If you are using Java Content Repository, the open_cursors value might need to be increased based on the table count in the Java Content Repository schema.
8. Raise the number of parallel servers as appropriate. For example, if you have more than 875 parallel servers, you should set the parallel_max_servers to 1200.
9. The Oracle parameter CURSOR_SHARING allows similar SQL Statements to be shared when possible, which prevents parsing and establishing a new execution plan. The execution plan is used by Oracle to gather the data that is needed to satisfy a request. There are two options for CURSOR_SHARING. WebSphere Portal Express supports both options. Regardless of the option that is selected, portlet applications should not be affected. Contact your database administrator for further assistance on these options.

   FORCE
   When you select this option, Oracle uses the same execution plan for all SQLs that are similar in value even if the values are different. When you use this option, the execution plan may not provide optimum performance. For example, similar SQLs with different values may behave differently when executed running the same plan.

   EXACT
   When you select this option, Oracle only shares the same execution plan for SQLs that are identical and use the same values. This option removes the risk of a SQL statement being executed when optimum performance conditions do not exist.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Database transfer: Download the script and setup your Oracle database

Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

About this task

Attention: All steps for all database environments are included without consideration for your environment. Use the Configuration Wizard to generate custom instructions for your environment.

Procedure

1. Copy the downloaded script to the database server and extract it.
2. Run the SQL statements that are listed in this script in your SQL processor.

   Example of the script that the Configuration Wizard generates

```
CREATE USER runtime_ID
 IDENTIFIED BY runtime_pwd
 DEFAULT TABLESPACE USERS
 TEMPORARY TABLESPACE TEMP;



CREATE USER config_ID
 IDENTIFIED BY config_pwd
 DEFAULT TABLESPACE USERS
 TEMPORARY TABLESPACE TEMP;
CREATE USER release
 IDENTIFIED BY config_pwd
 DEFAULT TABLESPACE USERS
 TEMPORARY TABLESPACE TEMP;

GRANT UNLIMITED TABLESPACE TO release;

CREATE ROLE WP_BASE_CONFIG_USERS NOT IDENTIFIED;
GRANT ALTER ANY TABLE, CREATE ANY TABLE, DROP ANY TABLE,
 CREATE ANY INDEX, DROP ANY INDEX,
 INSERT ANY TABLE, UPDATE ANY TABLE, SELECT ANY TABLE, DELETE ANY TABLE,
 CREATE SESSION
 TO WP_BASE_CONFIG_USERS;


GRANT SELECT ON DBA_PENDING_TRANSACTIONS TO WP_BASE_CONFIG_USERS;
GRANT WP_BASE_CONFIG_USERS TO config_ID;

CREATE ROLE WP_BASE_RUNTIME_USERS NOT IDENTIFIED;
GRANT CREATE SESSION TO WP_BASE_RUNTIME_USERS;
GRANT SELECT ON DBA_PENDING_TRANSACTIONS TO WP_BASE_RUNTIME_USERS;
GRANT WP_BASE_RUNTIME_USERS TO runtime_ID;



CREATE USER community
 IDENTIFIED BY config_pwd
 DEFAULT TABLESPACE USERS
 TEMPORARY TABLESPACE TEMP;
```

```
GRANT UNLIMITED TABLESPACE TO community;

GRANT ALTER ANY TABLE, CREATE ANY TABLE, DROP ANY TABLE,
 CREATE ANY INDEX, DROP ANY INDEX,
 INSERT ANY TABLE, UPDATE ANY TABLE, SELECT ANY TABLE, DELETE ANY TABLE,
 CREATE SESSION
 TO WP_BASE_CONFIG_USERS;


GRANT SELECT ON DBA_PENDING_TRANSACTIONS TO WP_BASE_CONFIG_USERS;
GRANT WP_BASE_CONFIG_USERS TO config_ID;

GRANT CREATE SESSION TO WP_BASE_RUNTIME_USERS;
GRANT SELECT ON DBA_PENDING_TRANSACTIONS TO WP_BASE_RUNTIME_USERS;
GRANT WP_BASE_RUNTIME_USERS TO runtime_ID;




CREATE USER customization
 IDENTIFIED BY config_pwd
 DEFAULT TABLESPACE USERS
 TEMPORARY TABLESPACE TEMP;

GRANT UNLIMITED TABLESPACE TO customization;

GRANT ALTER ANY TABLE, CREATE ANY TABLE, DROP ANY TABLE,
 CREATE ANY INDEX, DROP ANY INDEX,
 INSERT ANY TABLE, UPDATE ANY TABLE, SELECT ANY TABLE, DELETE ANY TABLE,
 CREATE SESSION
 TO WP_BASE_CONFIG_USERS;


GRANT SELECT ON DBA_PENDING_TRANSACTIONS TO WP_BASE_CONFIG_USERS;
GRANT WP_BASE_CONFIG_USERS TO config_ID;

GRANT CREATE SESSION TO WP_BASE_RUNTIME_USERS;
GRANT SELECT ON DBA_PENDING_TRANSACTIONS TO WP_BASE_RUNTIME_USERS;
GRANT WP_BASE_RUNTIME_USERS TO runtime_ID;




CREATE USER jcr
 IDENTIFIED BY config_pwd
 DEFAULT TABLESPACE USERS
 TEMPORARY TABLESPACE TEMP;

GRANT UNLIMITED TABLESPACE TO jcr;

CREATE ROLE WP_JCR_CONFIG_USERS NOT IDENTIFIED;
GRANT ALTER ANY TABLE, CREATE ANY TABLE, DROP ANY TABLE,
 CREATE ANY INDEX, DROP ANY INDEX,
 INSERT ANY TABLE, UPDATE ANY TABLE, SELECT ANY TABLE, DELETE ANY TABLE,
 CREATE SESSION
 TO WP_JCR_CONFIG_USERS;

GRANT CREATE TABLESPACE, DROP TABLESPACE,
 CREATE ANY SEQUENCE, DROP ANY SEQUENCE,
 CREATE ANY TRIGGER, DROP ANY TRIGGER,
 CREATE ANY TYPE, DROP ANY TYPE, EXECUTE ANY TYPE,
 CREATE ANY VIEW, DROP ANY VIEW
 TO WP_JCR_CONFIG_USERS;
GRANT SELECT ON DBA_IND_COLUMNS to WP_JCR_CONFIG_USERS;
GRANT SELECT ON DBA_INDEXES to WP_JCR_CONFIG_USERS;


GRANT SELECT ON DBA_PENDING_TRANSACTIONS TO WP_JCR_CONFIG_USERS;
GRANT WP_JCR_CONFIG_USERS TO config_ID;

CREATE ROLE WP_JCR_RUNTIME_USERS NOT IDENTIFIED;
GRANT CREATE SESSION TO WP_JCR_RUNTIME_USERS;
GRANT SELECT ON DBA_PENDING_TRANSACTIONS TO WP_JCR_RUNTIME_USERS;
GRANT WP_JCR_RUNTIME_USERS TO runtime_ID;




CREATE USER feedback
 IDENTIFIED BY config_pwd
 DEFAULT TABLESPACE USERS
 TEMPORARY TABLESPACE TEMP;

GRANT UNLIMITED TABLESPACE TO feedback;

CREATE ROLE WP_PZN_CONFIG_USERS NOT IDENTIFIED;
GRANT ALTER ANY TABLE, CREATE ANY TABLE, DROP ANY TABLE,
 CREATE ANY INDEX, DROP ANY INDEX,
 INSERT ANY TABLE, UPDATE ANY TABLE, SELECT ANY TABLE, DELETE ANY TABLE,
 CREATE SESSION
 TO WP_PZN_CONFIG_USERS;
```

```
GRANT CREATE ANY SEQUENCE, DROP ANY SEQUENCE TO WP_PZN_CONFIG_USERS;

GRANT SELECT ON DBA_PENDING_TRANSACTIONS TO WP_PZN_CONFIG_USERS;
GRANT WP_PZN_CONFIG_USERS TO config_ID;

CREATE ROLE WP_PZN_RUNTIME_USERS NOT IDENTIFIED;
GRANT CREATE SESSION TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON DBA_PENDING_TRANSACTIONS TO WP_PZN_RUNTIME_USERS;
GRANT WP_PZN_RUNTIME_USERS TO runtime_ID;
```

```
CREATE USER likeminds
 IDENTIFIED BY config_pwd
 DEFAULT TABLESPACE USERS
 TEMPORARY TABLESPACE TEMP;

GRANT UNLIMITED TABLESPACE TO likeminds;

GRANT ALTER ANY TABLE, CREATE ANY TABLE, DROP ANY TABLE,
 CREATE ANY INDEX, DROP ANY INDEX,
 INSERT ANY TABLE, UPDATE ANY TABLE, SELECT ANY TABLE, DELETE ANY TABLE,
 CREATE SESSION
 TO WP_PZN_CONFIG_USERS;


GRANT SELECT ON DBA_PENDING_TRANSACTIONS TO WP_PZN_CONFIG_USERS;
GRANT WP_PZN_CONFIG_USERS TO config_ID;

GRANT CREATE SESSION TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON DBA_PENDING_TRANSACTIONS TO WP_PZN_RUNTIME_USERS;
GRANT WP_PZN_RUNTIME_USERS TO runtime_ID;
CREATE TABLESPACE ICMLFQ32 DATAFILE 'oracle/product/11.2.0/dbhome_1/wpsdb/data/wpsdb_ICMLFQ32_01.dbf' SIZE 300M REUSE AUTOEXTEND ON NEXT
CREATE TABLESPACE ICMLNF32 DATAFILE 'oracle/product/11.2.0/dbhome_1/wpsdb/data/wpsdb_ICMLNF32_01.dbf' SIZE 25M REUSE AUTOEXTEND ON NEXT 1
CREATE TABLESPACE ICMVFQ04 DATAFILE 'oracle/product/11.2.0/dbhome_1/wpsdb/data/wpsdb_ICMVFQ04_01.dbf' SIZE 25M REUSE AUTOEXTEND ON NEXT 1
CREATE TABLESPACE ICMSFQ04 DATAFILE 'oracle/product/11.2.0/dbhome_1/wpsdb/data/wpsdb_ICMSFQ04_01.dbf' SIZE 150M REUSE AUTOEXTEND ON NEXT
CREATE TABLESPACE ICMLSNDX DATAFILE 'oracle/product/11.2.0/dbhome_1/wpsdb/index/wpsdb_ICMLSNDX_01.dbf' SIZE 10M REUSE AUTOEXTEND ON NEXT
```

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Database transfer: Download the script and grant permissions for the Oracle database users

Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

About this task

Attention: All steps for all database environments are included without consideration for your environment. Use the Configuration Wizard to generate custom instructions for your environment.

Procedure

1. Copy the downloaded script to the database server and extract it.
2. Run the SQL statements that are listed in this script in your SQL processor.

Example of the script that is generated by the Configuration Wizard.

```
CREATE ROLE WP_BASE_RUNTIME_USERS NOT IDENTIFIED;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.ACL TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PROT_RES TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.ACL_ENTRY TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.ROLE_INST TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.LNK_USER_ROLE TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.UNIQUE_NAME TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PROT_RES_DEP TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.CONTENT_MAPPING TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PAGE_INST TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PAGE_INST_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PAGE_INST_MAD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PAGE_INST_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_INST TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_INST_PREF TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.COMP_INST TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.COMP_INST_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.COMP_INST_MAD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_ACT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_ACT_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_ACT_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_ACT_MD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_PROP TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_PROP_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_PROP_ALIAS TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_PARM TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_PARM_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_WIRE TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_WIRE_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_WIRE_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_WIRE_MD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PAGE_ACT_ASC TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PAGE_ACT_ASC_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.AUTH_LEVEL TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.VAULT_RESOURCES TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.VAULT_DATA TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.CRED_SEGMENT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.CRED_SLOT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.CRED_SLOT_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.APP_DESC TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.APP_DESC_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.APP_DESC_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_DESC TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_DESC_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_DESC_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_DESC_PREF TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.WF_DEF TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.WF_DEF_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.WF_DATA TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.WF_TP_MAPPING TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.WF_TP_MAPPING_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.WEB_MOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.WEB_MOD_UATTR TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.WSRP_PROD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.WSRP_PROD_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.WSRP_PROD_UATTR TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.WSRP_PROD_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.WSRP_PROD_PORT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.SERV_DESC TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.SERV_DESC_MAD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.SERV_DESC_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.SERV_DESC_PREF TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.SERV_DESC_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PUB_REND_PARM TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PUB_REND_PARM_QN TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PUB_REND_PARM_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.SKIN_DESC TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.SKIN_DESC_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.SKIN_DESC_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.SKIN_DESC_MAD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.THEME_DESC TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.THEME_DESC_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.THEME_DESC_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.THEME_DESC_MAD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PAGE_LAYOUT_DESC TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PAGE_LAYOUT_DESC_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PAGE_LAYOUT_DESC_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PAGE_LAYOUT_DESC_MAD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.SERVER_INFO TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.SERVER_INFO_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.SERVER_INFO_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.USER_DESC TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.USER_DESC_STATE TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.URL_MAP_CTX TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.URL_MAP_CTX_LBLS TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.LNK_PAGE_PORT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.LNK_THEME_SKIN TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.MARKUP_DESC TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.MARKUP_DESC_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.CLIENT_DESC TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.CLIENT_DESC_CAPS TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.LANG_DESC TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.LANG_DESC_LOD TO WP_BASE_RUNTIME_USERS;
```

```
GRANT SELECT, INSERT, UPDATE, DELETE ON release.EVT_HANDLER TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.EVT_INTERFACE TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORTAL_URL TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORTAL_URL_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.VP_DESC TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.VP_DESC_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.ACTION_SET TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.ACTION_SET_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.ACTION_DESC TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.ACTION_DESC_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.RES_TYPE TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.RES_TYPE_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.LNK_RES_TYPE_RES TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.LNK_RES_TYPE_AS TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.LNK_RESTYPE TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PSE_DESC TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PSE_DESC_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PSE_CATEGORY TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PSE_CATEGORY_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PSE_CATEGORY_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PSE_SOURCE TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PSE_SOURCE_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PSE_SOURCE_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.WP_SEARCH_REG TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.SEARCH_REG TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.SEARCH_REG_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.CP_FILTER_INST TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.CP_FILTER_DATA TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.DEVICE_CLASS_DESC TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.LNK_PAGE_DC TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.DEVICE_CLASS_DESC_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.VANITY_URL TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.OUTBOUND_CONFIG TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.OUTBOUND_CONFIG_VAR TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.OUTBOUND_MAPPING TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.OUTBOUND_MAPPING_MD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.OUTBOUND_MAPPING_IP TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.OUTBOUND_POLICY TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.OUTBOUND_POLICY_AM TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.OUTBOUND_POLICY_FC TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.OUTBOUND_POLICY_HD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.OUTBOUND_POLICY_MT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.OUTBOUND_POLICY_MD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.OUTBOUND_COOKIE_RULE TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.OUTBOUND_COOKIE_RULE_CN TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.OUTBOUND_COOKIE_RULE_TR TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PCM_DIALOG TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PCM_DIALOG_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PCM_DIALOG_MD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PCM_TRANSITION_ENDPOINT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PCM_TRANSITION_ENDPOINT_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PCM_TRANSITION_ENDPOINT_MD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PCM_TRANSITION TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PCM_TRANSITION_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PCM_TRANSITION_MD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PAGE_INST_DRAFT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PAGE_INST_DRAFT_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PAGE_INST_DRAFT_MAD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PAGE_INST_DRAFT_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_INST_DRAFT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_INST_DRAFT_PREF TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.COMP_INST_DRAFT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.COMP_INST_DRAFT_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.COMP_INST_DRAFT_MAD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.UNIQUE_NAME_DRAFT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PROT_RES_DRAFT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.ROLE_INST_DRAFT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.LNK_USER_ROLE_DRAFT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.CONTENT_MAPPING_DRAFT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_WIRE_DRAFT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_WIRE_DRAFT_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_WIRE_DRAFT_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PORT_WIRE_DRAFT_MD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PAGE_ACT_ASC_DRAFT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.PAGE_ACT_ASC_DRAFT_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.LNK_PAGE_PORT_DRAFT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.LNK_PAGE_DC_DRAFT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.VANITY_URL_DRAFT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.WPS_SCDTASK TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.WPS_SCDTREG TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.WPS_SCDLMGR TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON release.WPS_SCDLMPR TO WP_BASE_RUNTIME_USERS;


CREATE ROLE WP_BASE_RUNTIME_USERS NOT IDENTIFIED;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.ACL TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PROT_RES TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.ACL_ENTRY TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.ROLE_INST TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.LNK_USER_ROLE TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.UNIQUE_NAME TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PROT_RES_DEP TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.CONTENT_MAPPING TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PAGE_INST TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PAGE_INST_LOD TO WP_BASE_RUNTIME_USERS;
```

```
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PAGE_INST_MAD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PAGE_INST_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PORT_INST TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PORT_INST_PREF TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.COMP_INST TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.COMP_INST_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.COMP_INST_MAD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PORT_ACT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PORT_ACT_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PORT_ACT_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PORT_ACT_MD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PORT_PROP TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PORT_PROP_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PORT_PROP_ALIAS TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PORT_PARM TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PORT_PARM_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PORT_WIRE TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PORT_WIRE_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PORT_WIRE_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PORT_WIRE_MD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PAGE_ACT_ASC TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.PAGE_ACT_ASC_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.AUTH_LEVEL TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.VAULT_RESOURCES TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.VAULT_DATA TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.CRED_SEGMENT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.CRED_SLOT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.CRED_SLOT_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.CP_RESOURCE_TAG TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.CP_RESOURCE_RATING TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.CONTACTS_MEMBER TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.CONTACTS_MEMBER_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.CONTACTS_GROUP TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.LNK_MEMBER_GROUP TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.CATA TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.CATA_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.CATE TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.CATE_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.CATA_ENTRY TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.CATA_ENTRY_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON customization.CATA_ENTRY_DD TO WP_BASE_RUNTIME_USERS;


CREATE ROLE WP_BASE_RUNTIME_USERS NOT IDENTIFIED;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.ACL TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PROT_RES TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.ACL_ENTRY TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.ROLE_INST TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.LNK_USER_ROLE TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.UNIQUE_NAME TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PROT_RES_DEP TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.CONTENT_MAPPING TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PAGE_INST TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PAGE_INST_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PAGE_INST_MAD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PAGE_INST_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PORT_INST TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PORT_INST_PREF TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.COMP_INST TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.COMP_INST_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.COMP_INST_MAD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PORT_ACT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PORT_ACT_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PORT_ACT_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PORT_ACT_MD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PORT_PROP TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PORT_PROP_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PORT_PROP_ALIAS TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PORT_PARM TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PORT_PARM_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PORT_WIRE TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PORT_WIRE_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PORT_WIRE_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PORT_WIRE_MD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PAGE_ACT_ASC TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PAGE_ACT_ASC_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AUTH_LEVEL TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.APP_DESC TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.APP_DESC_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.APP_DESC_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PORT_DESC TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PORT_DESC_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PORT_DESC_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.PORT_DESC_PREF TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.WF_DEF TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.WF_DEF_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.WF_DATA TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.WF_TP_MAPPING TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.WF_TP_MAPPING_DD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AI_CATEGORY TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AI_CATEGORY_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AI_TFOLDER TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AI_TFOLDER_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AI_TEMPLATE TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AI_TEMPLATE_LOD TO WP_BASE_RUNTIME_USERS;
```

```
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AI_TEMPLATE_ZIP TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AI_TEMP_METADATA TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AI_APP TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AI_APP_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AI_APP_METADATA TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AI_RSTR_PT TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AI_BC TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AI_BC_METADATA TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AI_BC_VAR TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AI_PARM TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AI_PARM_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AI_WIRE TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AI_BC_WIRE TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AI_APP_WIRE TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AI_BC_META_WIRE TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.AI_PARM_BCMD_WIRE TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.CP_CUSTOMRESOURCE TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.CP_CATEGORY TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.LOC_DATA TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.LOC_DATA_LOD TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.HASHED_BLOB TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.CP_RESOURCE_UPDATED TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.CP_RESOURCE_TAG TO WP_BASE_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON community.CP_RESOURCE_RATING TO WP_BASE_RUNTIME_USERS;


GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ACL TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.PROT_RES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ACL_ENTRY TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ROLE_INST TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.LNK_USER_ROLE TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.UNIQUE_NAME TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.PROT_RES_DEP TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.CONTENT_MAPPING TO WP_JCR_RUNTIME_USERS;


CREATE ROLE WP_PZN_RUNTIME_USERS NOT IDENTIFIED;
GRANT
  CREATE SESSION
TO
  WP_PZN_RUNTIME_USERS;
GRANT
  SELECT ON DBA_PENDING_TRANSACTIONS
TO
  WP_PZN_RUNTIME_USERS;
CREATE ROLE WP_PZN_RUNTIME_USERS NOT IDENTIFIED;
GRANT CREATE SESSION TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON DBA_PENDING_TRANSACTIONS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.AGGREGATEKEY TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.AGGREGATES TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.AGGREGATESTATUS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.AGGREGATE_CONTENT TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.BROWSERS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.CALENDAR TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.CATEGORIES TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.CATEGORYMAP TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.CATEGORY_PATTERNS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.CATEGORY_SETS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.COOKIESSTATUS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.DOMAIN TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.ENTITIES TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.ENTITYTRAVERSAL TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.HITPARMS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.HIT_FACTS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.HTTPVERSION TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.IMPORTHISTORY TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.JAVASCRIPTSTATUS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.KEY TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.KEY_VALUE_COMBO TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.KEY_VALUE_PAIR TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.LINKAGE TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.LOGCONTROL TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.LOGS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.LOG_FILE_STATUS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.NETWORKS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.PARMS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.PLATFORMS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.PROTOCOLS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.REFERRER TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.REFERRERHOST TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.REFERRERURL TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.RESETSTATUS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.RESOURCES TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.RESULT TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.RETURNCODES TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.SERVERNODES TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.SERVERS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.SESSIONPARMS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.SESSION_FACTS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.SUBDOMAINS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.TIMEOFDAY TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.TIMESPAN TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.USERAGENTS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.USERS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.VALUE TO WP_PZN_RUNTIME_USERS;
```

```
GRANT SELECT, INSERT, UPDATE, DELETE ON feedback.WEB_NODES TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.AGGREGATESTATUS_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.AGGREGATES_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.BROWSERS_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.CALENDAR_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.CATEGORIES_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.CATEGORY_PATTERNS_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.CATEGORY_SETS_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.COOKIESSTATUS_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.DOMAIN_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.ENTITIES_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.ENTITYTRAVERSAL_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.HIT_FACTS_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.HTTPVERSION_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.IMPORTHISTORY_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.JAVASCRIPTSTATUS_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.KEY_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.KEY_VALUE_PAIR_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.LINKAGE_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.LOGS_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.LOG_FILE_STATUS_RES_ID_1 TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.LOG_FILE_STATUS_RES_ID_2 TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.NETWORKS_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.PARMS_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.PLATFORMS_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.PROTOCOLS_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.REFERRERHOST_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.REFERRERURL_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.REFERRER_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.RESETSTATUS_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.RESOURCES_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.RESULT_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.RETURNCODES_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.SERVERS_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.SESSION_FACTS_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.SUBDOMAINS_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.TIMEOFDAY_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.TIMESPAN_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.USERAGENTS_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.USERS_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.VALUE_ID TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON feedback.WEB_NODES_ID TO WP_PZN_RUNTIME_USERS;
CREATE ROLE WP_PZN_RUNTIME_USERS NOT IDENTIFIED;
GRANT CREATE SESSION TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON DBA_PENDING_TRANSACTIONS TO WP_PZN_RUNTIME_USERS;


CREATE ROLE WP_PZN_RUNTIME_USERS NOT IDENTIFIED;
GRANT
  CREATE SESSION
TO
  WP_PZN_RUNTIME_USERS;
GRANT
  SELECT ON DBA_PENDING_TRANSACTIONS
TO
  WP_PZN_RUNTIME_USERS;
CREATE ROLE WP_PZN_RUNTIME_USERS NOT IDENTIFIED;
GRANT CREATE SESSION TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON DBA_PENDING_TRANSACTIONS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_ADMIN_USERS TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_CFG TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_ITEM_DATA TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_MBA_SCORED TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_RTG_POOL TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_TRX_MENTOR TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_TRX_POOL TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_TRX_TYPE TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_USER_DATA TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_USER_MENTOR TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_USER_RATING TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_USER_TRX TO WP_PZN_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON likeminds.LPS_VISIT_LIST TO WP_PZN_RUNTIME_USERS;
CREATE ROLE WP_PZN_RUNTIME_USERS NOT IDENTIFIED;
GRANT CREATE SESSION TO WP_PZN_RUNTIME_USERS;
GRANT SELECT ON DBA_PENDING_TRANSACTIONS TO WP_PZN_RUNTIME_USERS;


GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_OBJECTS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_TIMERS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.JOBS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.OBJECTS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.TASKJOBMAPPING TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.TASKS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_USER_SHORTCUT TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_CONTROLLABLES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_LIBRARIES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_PROJECTS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_PROJLIBS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_SYNDICATORS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_SYNDLIBS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_VERSIONS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_VPORTALS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_SYND_FAILURE TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_SYND_SESSION TO WP_JCR_RUNTIME_USERS;
```

```
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.WCM_SYND_EXTRA_DATA TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.OPENJPA_SEQUENCE_TABLE TO WP_JCR_RUNTIME_USERS;



GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTADMINDOMAINS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTACCESSCODES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTCOLLNAME TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTSYSCONTROL TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTATTRDEFS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTATTRGROUP TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTNLSLANGUAGES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTMAXKEYWORD TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTNLSKEYWORDS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTITEMTYPEDEFS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTITVIEWDEFS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTITVIEWID TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTCOMPDEFS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTCOMPATTRS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTCOMPATTRSFK TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTCOMPVIEWDEFS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTCOMPVIEWATTRS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTITEMS001001 TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTITEMVER001001 TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTLINKS001001 TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTMIMETYPES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTITEMSTODELETE TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTXDOOBJECTS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTRI001001 TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTVIEWACCESS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMUT00600001 TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRNODELOCKS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRWS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRNSURIS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRNSPREFIXES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMUTSWIDE0 TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMUTMWIDE0 TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRWSNODES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRLINKS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRDELTALINKS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRLINKREL TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRDELTALR TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRGLBLPROPS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRADDTLPROPS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRNODETYPES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRPROPDEFS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRNODEDEFS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRRNODETYPES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRSUPERTYPES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRTIMESTAMPS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRIDS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRDUMMY TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRMAXSNSIDX TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRGENDDL TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRINDEXNAMES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRINDEXES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRTSPENDING TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRTSERRORS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRTSINDEXES TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRTSPATHINFO TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRTSPOSCOR TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRTSNEGCOR TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRAPPL TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRNODEREGISTER TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRTSFULLCRAWLTOPICS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRTSSUBSCRIPTIONMANAGER TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRTSSEEDLISTPENDING TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRTSINCSUBSMSGS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCREDITIONINFO TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCREDITIONS TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRREMOVEHLP TO WP_JCR_RUNTIME_USERS;
GRANT SELECT, INSERT, UPDATE, DELETE ON jcr.ICMSTJCRIMPORTREF TO WP_JCR_RUNTIME_USERS;
```

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Database transfer: Set up the data and index directory for JCR collation

Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

About this task

Attention: All steps for all database environments are included without consideration for your environment. Use the Configuration Wizard to generate custom instructions for your environment.

Create the data directory, `data`, and the index directory, `index`, on your database server

Procedure

1. Create the `data` directory in the following location:
   &*dbpath*/**jcr.DbName**/data/
2. Create the `index` directory in the following location:
   &*dbpath*/**jcr.DbName**/index/

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Database transfer: Set up JCR collation

Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

About this task

Attention: All steps for all database environments are included without consideration for your environment. Use the Configuration Wizard to generate custom instructions for your environment.

JCR collation is recommended when the language locales of your users do not natively collate correctly in the DB2 database and when language locale correct ordering is important.

Procedure

1. Stop the WebSphere Portal Express server.
2. Copy files from the WebSphere Portal Express server to a temporary directory on the DB2 server.
   - `${WpsInstallLocation}/jcr/wp.content.repository.install/lib/wp.content.repository.install.jar`
   - `${WasUserHome}/PortalServer/jcr/config/registerCollationUDFTemplate.sql`

Set up collation on the database where the JCR domain is located.

3. Change to this directory:

   db2_instance_owner_home`/sqllib/function`
4. Enter this command:

   ```
   db2home/sqllib/java/jdk/bin/jar -xvf ${wf.ejp.collationTempDir}/
   wp.content.repository.install.jar icm/CollationUDF.class
   ```
5. Change to the temporary directory where you copied the files in a previous step. For example, you can use this temporary directory on the DB2 server:

   `${wf.ejp.collationTempDir}`
6. Open the file `registerCollationUDFTemplate.sql`, and change the multiple `<`*`SCHEMA`*`>` references to the JCR schema; for example, JCR. The value set for `<`*`SCHEMA`*`>` should match the value set for the jcr.DbSchema property. You specify jcr.DbSchema in the configuration file `wkplc_dbdomain.properties` when you modify database properties.
7. Connect to the JCR database.

   ```
   db2 connect to ${jcr.DbName} user ${jcr.DBA.DbUser} using
   ${jcr.DBA.DbPassword}
   ```
8. Enter this command to run the script:

   ```
   db2 -tvf ${wf.ejp.collationTempDir}/registerCollationUDFTemplate.sql
   ```
9. Disconnect from the JCR database.
10. Restart the DB2 instance.

Verify that the UDF is registered properly.

11. Log in as `${jcr.DbUser}`.
12. Open a DB2 terminal window.
13. Connect to the database that contains JCR domain:

    ```
    db2 connect to ${jcr.DbName} user ${jcr.DbUser} using ${jcr.DbPassword}
    ```
14. When you have connected to the JCR database, verify that the UDF is registered properly. To verify the UDF registration, run this command:

    ```
    db2 values ${jcr.DbSchema}.sortkeyj('abc','en')
    ```
15. Disconnect from DB2 terminal window:

    ```
    db2 disconnect all
    db2 terminate
    ```
16. Start the WebSphere Portal Express server.

Update the collation configuration options.

17. Log in to WebSphere Integrated Solutions Console.
18. Go to **Resources** > **Resource Environment** > **Resource Environment Providers** > **JCR ConfigService PortalContent** > **Custom properties**.
19. Add or update the following properties as necessary:

Enable/Disable collation support for all DB2 platforms(LUW, Z, I), this is disabled (false) by default

Name: jcr.query.collation.db2.enabled

Value: true

Type: java.lang.String

Name: jcr.query.collation.en

Value: en

Type: java.lang.String

20. To apply your settings, stop WebSphere Portal Express, and then restart WebSphere Portal Express.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Related reference:

"JCR search service configuration parameters" on page 680
The following search service configuration parameters can be modified to enable and configure searching for content that is stored in the JCR database. These JCR search service configuration parameters can be modified by accessing the **JCR ConfigService PortalContent** resource environment provider.

Database transfer: Improve database response time for your database domains

Manual steps from the Configuration Wizard are included in IBM Knowledge Center for reference and for advanced users. The procedure includes variables and steps for different databases. When you use the wizard to configure your deployment, it replaces the variable with information that you provided in the wizard. It also shows only the steps that are specific to your environment. The instructions that the wizard generates are specific to your environment.

About this task

Attention: All steps for all database environments are included without consideration for your environment. Use the Configuration Wizard to generate custom instructions for your environment.

After transferring the database tables, run the `dbms_stats.gather_schema_stats` command to avoid slow database response.

Run the following commands for your JCR domain. You might also want to run the following commands on all of your database domains, if there are large amounts of data. To run the commands on domains other than JCR, you must replace the *jcr*`.DbName` and *jcr*`.DbUser` with the specific domain for which you plan to run the commands. For example, *release*`.DbName` or *feedback*`.DbName`.

Note: *jcr.DbUser* in the following command refers to the schema owner of the JCR database objects.

Procedure

Run the following commands. Start SQL*Plus and log in to the **jcr.DbName** database as the **jcr.DbUser** user SQL> execute dbms_stats.gather_schema_stats(ownname=> '**jcr.DbUser**', cascade=> TRUE); SQL > commit; Exit sqlplus

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Oracle: Creating JCR table spaces (Automatic Storage Management)

If you configured your database with Automatic Storage Management, you might need to perform additional manual instructions to create JCR table spaces when you use the Database Transfer option in the Configuration Wizard . If you select the option to let the wizard create your database schemas and assign permissions, you must perform the steps in this topic after you run the setup database script.

Procedure

1. Create table spaces using the following commands as examples:

   Note:

   - If you plan on customizing your table space names, you must ensure that the customized table space names are used during database transfer. Custom table spaces must exist prior to performing database transfer.
   - *ASM_disk_group_name* is the disk group name that you used to configure your Automatic Storage Management environment.
   - Ensure that the '.' is included in the variables when you substitute the values of your environment with these variables.

   create tablespace *ICMLFQ32* **datafile** '*+ASM_disk_group_name*' **size** *300M* **reuse** *autoextend on next 10M* **maxsize** *UNLIMITED* **extent management** local *autoallocate*;

   create tablespace *ICMLNF32* **datafile** '*+ASM_disk_group_name*' **size** *25M* **reuse** *autoextend on next 10M* **maxsize** *UNLIMITED* **extent management** local *autoallocate*;

   create tablespace *ICMVFQ04* **datafile** '*+ASM_disk_group_name*' **size** *25M* **reuse** *autoextend on next 10M* **maxsize** *UNLIMITED* **extent management** local *autoallocate*;

   create tablespace *ICMSFQ04* **datafile** '*+ASM_disk_group_name*' **size** *150M* **reuse** *autoextend on next 10M* **maxsize** *UNLIMITED* **extent management** local *autoallocate*;

create tablespace *ICMLSNDX* **datafile** '*+ASM_disk_group_name*' **size** *10M* **reuse** *autoextend on next 10M* **maxsize** *UNLIMITED* **extent management** local *autoallocate*;

a. Set the **size**, **autoextend**, and **maxsize** values according to your environment. For example, you may want to change the **maxsize** to a set value rather than *UNLIMITED*.
b. Consult your Database Administrator for specific guidance about creating tablespaces for your environment.
c. Refer to the Oracle command reference for more information about using the **create tablespaces** command.

2. Ensure the database is registered with the Oracle listener. Use the tnsnames.ora file to describe this database and recycle the listener.

Assigning custom table spaces

You cannot use the **Database Transfer** option in the configuration wizard to assign custom table spaces on your database server. You can perform manual steps to assign custom table spaces.

"DB2: Assigning custom table spaces"
The repository of WebSphere Portal Express consists of many tables and indices that are created in default table spaces. When using an existing set of table spaces for the objects of the repository, specify this when executing the database transfer to the target database system.

"Oracle: Assigning custom table spaces" on page 556
The repository of WebSphere Portal Express consists of many tables and indices that are created in default table spaces. When using an existing set of table spaces for the objects of the repository, specify this when executing the database transfer to the target database system.

"SQL Server: Assigning custom table spaces" on page 558
The repository of WebSphere Portal Express consists of many tables and indices that are created in default filegroups. When using an existing set of filegroups for the objects of the repository, specify this when executing the database transfer to the target management database system.

"DB2 for z/os: Assigning custom table spaces" on page 559
The repository of WebSphere Portal Express consists of many tables and indices that are created in default table spaces. When using an existing set of table spaces for the objects of the repository, specify this when executing the database transfer to the target database system.

DB2: Assigning custom table spaces

The repository of WebSphere Portal Express consists of many tables and indices that are created in default table spaces. When using an existing set of table spaces for the objects of the repository, specify this when executing the database transfer to the target database system.

Before you begin

Before you begin:

- The custom table spaces must exist prior to the execution of database transfer.
- To see which table spaces can be customized in each domain, refer to the *wp_profile_root*/PortalServer/config/tablespaces/dbdomain.space_mapping.properties file.
- The page size of table spaces used by WebSphere Portal must be 8192 bytes.
- For details on creating table spaces refer to the documentation for the database.

About this task

If custom table spaces are assigned, each must be assigned explicitly. The default table spaces can be used to contain database objects; however the name of the default table space must be specified in the corresponding mapping files. This applies to all database domains that are transferred in a single database transfer.

To configure custom table space assignments:

Procedure

1. Determine the names of your custom table spaces.
2. Open the mapping file *wp_profile_root* `/PortalServer/config/tablespaces/` *`dbdomain`*`.space_mapping.properties` that specifies the table space and index space property pairs for each database table:

   dbdomain.table_name.tablespace

   dbdomain.table_name.index_name.indexspace

   For the file name and each table space and index space property pair, *dbdomain* can be any one of the following values:

   - release
   - community
   - customization
   - jcr
   - feedback
   - likeminds

   Note: For jcr, you need to open an additional maping file: *wp_profile_root*`/PortalServer/jcr/config/jcr.space_mapping.properties`. This mapping file contains additional table space and index space property pairs for each jcr.table_name.tablespace database table.
3. Assign a table space to each entry in the mapping file. The table space name must be prepended by the keyword `IN` and a space. For example: `community.COMP_INST.tablespace=IN COMM8KSPACE`
   Repeat this step for each domain that you are transferring.
4. Save and close *`dbdomain`*`.space_mapping.properties`.
5. From a command prompt, specify the option `-DuseCustomTablespaceMapping=true` when starting the database transfer. For example,

   Windows: `ConfigEngine.bat database-transfer -DuseCustomTablespaceMapping=true`

   Linux: `./ConfigEngine.sh database-transfer -DuseCustomTablespaceMapping=true`

   IBM i: `ConfigEngine.sh database-transfer -DuseCustomTablespaceMapping=true`

Oracle: Assigning custom table spaces

The repository of WebSphere Portal Express consists of many tables and indices that are created in default table spaces. When using an existing set of table spaces for the objects of the repository, specify this when executing the database transfer to the target database system.

Before you begin

Before you begin:

- The custom table spaces must exist prior to the execution of database transfer.
- To see which table spaces can be customized in each domain, refer to the *wp_profile_root*/PortalServer/config/tablespaces/dbdomain.space_mapping.properties file.
- For details on creating table spaces refer to the documentation for the database.

About this task

If custom table spaces are assigned, each must be assigned explicitly. The default table spaces can be used to contain database objects; however the name of the default table space must be specified in the corresponding mapping files. This applies to all database domains that are transferred in a single database transfer.

To configure custom table space assignments:

Procedure

1. Determine the names of your custom table spaces.
2. Open the mapping file *wp_profile_root* /PortalServer/config/tablespaces/*dbdomain*.space_mapping.properties that specifies the table space and index space property pairs for each database table:

   dbdomain.table_name.tablespace

   dbdomain.table_name.index_name.indexspace

   For the file name and each table space and index space property pair, *dbdomain* can be any one of the following values:
   - release
   - community
   - customization
   - jcr
   - feedback
   - likeminds

   Note: For jcr, you need to open an additional maping file: *wp_profile_root*/PortalServer/jcr/config/jcr.space_mapping.properties. This mapping file contains additional table space and index space property pairs for each jcr.table_name.tablespace database table.
3. Assign a table space to each entry in the mapping file. The table space name must be prepended by the keyword TABLESPACE and a space. For example: community.COMP_INST.tablespace=TABLESPACE COMM8KSPACE
   Repeat this step for each domain that you are transferring.
4. Save and close *dbdomain*.space_mapping.properties.
5. From a command prompt, specify the option -DuseCustomTablespaceMapping=true when starting the database transfer. For example,

   Windows: ConfigEngine.bat database-transfer -DuseCustomTablespaceMapping=true

   Linux: ./ConfigEngine.sh database-transfer -DuseCustomTablespaceMapping=true

IBM i: `ConfigEngine.sh database-transfer -DuseCustomTablespaceMapping=true`

SQL Server: Assigning custom table spaces

The repository of WebSphere Portal Express consists of many tables and indices that are created in default filegroups. When using an existing set of filegroups for the objects of the repository, specify this when executing the database transfer to the target management database system.

Before you begin

Before you begin:

- The custom filegroups must exist prior to the execution of the database transfer.
- To see which table spaces can be customized in each domain, refer to the *wp_profile_root*`/PortalServer/config/tablespaces/dbdomain.space_mapping.properties` file.
- For details on creating filegroups refer to the documentation for the database.

About this task

If custom filegroups are assigned, each must be assigned explicitly. The default filegroups can be used to contain database objects; however the name of the default file group must be specified in the corresponding mapping files. This applies to all database domains that are transferred in a single database transfer.

To configure custom filegroups:

Procedure

1. Determine the names of your custom filegroups.
2. Open the mapping file *wp_profile_root* `/PortalServer/config/tablespaces` that specifies the table space and index space for each property pairs for each database table:

   dbdomain.table_name.tablespace

   dbdomain.table_name.indexspace.indexspace

   For the file name and each table space and index space property pair, *dbdomain* can be any one of the following values:

   - release
   - community
   - customization
   - jcr
   - feedback
   - likeminds

   Note: For jcr, you need to open an additional maping file: *wp_profile_root*`/PortalServer/jcr/config/jcr.space_mapping.properties`. This mapping file contains additional table space and index space property pairs for each jcr.table_name.tablespace database table.
3. Assign a filespace to each entry in the mapping file. The filegroup name must be prepended by the keyword `ON` and a space. For example:
   `community.COMP_INST.tablespace=ON COMM8KSPACE`
   Repeat this step for each domain that you are transferring.
4. Save and close *`dbdomain`*`.space_mapping.properties`.

5. From a command prompt, specify the option `-DuseCustomTablespaceMapping=true` when starting the database transfer. For example,

   Windows: `ConfigEngine.bat database-transfer -DuseCustomTablespaceMapping=true`

   Linux: `./ConfigEngine.sh database-transfer -DuseCustomTablespaceMapping=true`

   IBM i: `ConfigEngine.sh database-transfer -DuseCustomTablespaceMapping=true`

DB2 for z/os: Assigning custom table spaces

The repository of WebSphere Portal Express consists of many tables and indices that are created in default table spaces. When using an existing set of table spaces for the objects of the repository, specify this when executing the database transfer to the target database system.

Before you begin

Before you begin:

- The custom table spaces must exist prior to the execution of database transfer.
- To see which table spaces can be customized in each domain, refer to the *wp_profile_root*`/PortalServer/config/tablespaces/dbdomain.space_mapping.properties` file.
- For details on creating table spaces refer to the documentation for the database.
- When using IBM DB2 Universal Database for z/OS as a data store, WebSphere Portal Express requires that its indexes are not padded. Therefore, you must set the DSNZPARM parameters to **RETVLCFK**=*NO* or **PADIX**=*NO*, or both.

About this task

If custom table spaces are assigned, each must be assigned explicitly. The default table spaces can be used to contain database objects; however the name of the default table space must be specified in the corresponding mapping files. This applies to all database domains that are transferred in a single database transfer.

To configure custom table space assignments:

Procedure

1. Determine the names of your custom table spaces.
2. Open the mapping file *wp_profile_root* `/PortalServer/config/tablespaces/`*`dbdomain`*`.space_mapping.properties` that specifies the table space and index space property pairs for each database table:

   dbdomain.table_name.tablespace

   dbdomain.table_name.index_name.indexspace

   For the file name and each table space and index space property pair, *dbdomain* can be any one of the following values:

   - release
   - community
   - customization
   - jcr
   - feedback

- likeminds

  Note: For jcr, you need to open an additional maping file: *wp_profile_root*/PortalServer/jcr/config/jcr.space_mapping.properties. This mapping file contains additional table space and index space property pairs for each jcr.table_name.tablespace database table.

3. Assign a table space to each .tablespace entry in the mapping file. Assignments to .indexspace entries are ignored. The table space name must be qualified by the database name and prepended by the keyword IN and a space. For example: community.COMP_INST.tablespace=IN COMM8KSPACE
   Repeat this step for each domain that you are transferring.
4. Save and close *dbdomain*.space_mapping.properties.
5. From a command prompt, specify the option -DuseCustomTablespaceMapping=true when starting the database transfer. For example,

   Windows: ConfigEngine.bat database-transfer -DuseCustomTablespaceMapping=true

   Linux: ./ConfigEngine.sh database-transfer -DuseCustomTablespaceMapping=true

   IBM i: ConfigEngine.sh database-transfer -DuseCustomTablespaceMapping=true

DB2: Enabling support for high availability recovery and rollforward recovery

Optional: To prevent data loss on DB2, modify the JCR schema to support High Availability Disaster Recovery and rollforward recovery.

Before you begin

Before you run the configuration task in this topic, use the **Database Transfer** option in the Configuration Wizard to transfer your data from Apache Derby to DB2

About this task

Procedure

1. Stop the portal server. Depending on your environment, you might have multiple portal servers to stop.
2. Back up the JCR database. This task permanently changes the JCR schema.
3. Run a configuration task from the *wp_profile_root*\ConfigEngine directory to convert columns in ICMUTMWIDE0 table from NOT LOGGED BLOG TO LOGGED BLOG.

   Note: For clustered environments, run this task only from one node.

   - Linux : ./ConfigEngine.sh reconfigure-jcr-for-hadr -DTransferDomainList=jcr -DWasPassword=*password*
   - Windows: ConfigEngine.bat reconfigure-jcr-for-hadr -DTransferDomainList=jcr -DWasPassword=*password*
4. Start the portal server.

Related tasks:

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

Connecting to existing database domains

View the steps to share a database domain between two separate WebSphere Portal Express instances (*instance1* and *instance2*), where *instance2* is connected to an *instance1* database domain.

Before you begin

A WebSphere Portal Express instance can be either a single server or WebSphere Portal Express cluster.

Notes:

- Release data cannot be shared between separate WebSphere Portal Express instances. Also, if Web Content Manager is installed, the JCR domain cannot be shared.
- You can share only the WebSphere Portal Express Version 8.5 data.
- The entire set of database properties that are specified in the file *wp_profile_root*`/ConfigEngine/properties/wkplc_dbomain.properties` must be valid for the current configuration of all database domains. If you plan to connect or transfer a single database domain, you must modify only the database properties for that domain before you run the **`connect-database`** or **`database-transfer`** tasks.

Note: The **`connect-database`** configuration task does not preserve customizations to the data sources for the WebSphere Portal Express databases. If you previously tuned your data sources for the WebSphere Portal Express databases, make a note of the settings, run **`connect-database`**, and reapply the tuning after you run the configuration task.

About this task

Both WebSphere Portal Express instances must be installed and operational.

Procedure

1. Verify that both WebSphere Portal Express instances are operational.
2. Ensure that the database client software is installed on *instance2* with the same settings as *instance1* and that you can connect to the remote database.
3. The *instance2* data sources must be made to point to the database used by *instance1*. The `wkplc_dbtype.properties` and `wkplc_dbdomain.properties` files must be updated to specify what remote database is used. This information must match the database information that is used for the *instance1* installation.

   Remember: Do not use the same data source names for the domains that are used in the previous version of the product. Use a different data source so that the **`connect-database`** command drops the old data source and create the new one with the new connection information.
4. Reconfigure *instance2* to use one or more remote database domains of *instance1* by running the following command from the directory *wp_profile_root*`/ConfigEngine`:

- Windows: `ConfigEngine.bat connect-database -DTransferDomainList=domain1,domain2,domain3 -DWasPassword=password`
- Linux: `./ConfigEngine.sh connect-database -DTransferDomainList=domain1,domain2,domain3 -DWasPassword=password`
- IBM i From the UserData directory:
  - `ConfigEngine.sh connect-database -DTransferDomainList=domain1,domain2,domain3 -DWasPassword=password`

5. Restart *instance2*.

User registry

User information is stored in your user registry. You can enable LDAP referrals, configure IBM WebSphere Portal Express to use dynamic groups, update your user registry, or delete your user registry configurations.

"Enable federated security" on page 563
You can use the Configuration Wizard to configure WebSphere Portal to use a federated LDAP for security. Use the following information to get familiar with the information you must provide in the wizard and the configuration procedure that it generates.

"Adding more attributes to VMM" on page 568
After you install IBM WebSphere Portal Express and configuring your LDAP user registries, you must adapt the attribute configuration to match the configured LDAP servers and your business needs. However, do not complete these steps if you configured only a database user registry or the default federated file-based repository for out-of-box installations.

"Enabling application groups" on page 577
You can define user groups within the database user registry with members (users or groups) contained in the federated LDAP user registry you configured with application groups. The benefit of application groups is that you can create groups that are only used in IBM WebSphere Portal Express.

"Advanced group configurations" on page 578
It is possible to use IBM WebSphere Portal Express ConfigEngine helper tasks to set up advanced Virtual Member Manager (VMM) group configurations. Specifically, it is possible to configure VMM to understand and use the "Group membership attribute" that many directories support.

"Adding realm support" on page 582
A realm is a group of users from one or more user registries that form a coherent group within IBM WebSphere Portal Express. Realms allow flexible user management with various configuration options. A realm must be mapped to a Virtual Portal to allow the defined users to log in to the Virtual Portal. When you configure realm support, complete these steps for each base entry that exists in your LDAP and database user registry to create multiple realm support.

"Updating your user registry" on page 585
After you deploy IBM WebSphere Portal Express, you can adjustment your federated user repository configurations. You can update these configurations to achieve the correct user registry configuration.

"Deleting your user registry configurations" on page 602
After you deploy IBM WebSphere Portal Express, you might not require some of the LDAP entity types, realms, realm base entries, or repositories that you created. You can delete these configurations to achieve the correct user registry configuration.

Enable federated security

You can use the Configuration Wizard to configure WebSphere Portal to use a federated LDAP for security. Use the following information to get familiar with the information you must provide in the wizard and the configuration procedure that it generates.

Configuration Wizard

The primary Configuration Wizard options are based on your target configuration topology, such as a stand-alone server or a cluster. The federated security option is included with both **Set Up a Stand-alone Server** and **Set Up a Cluster**. For the stand-alone server topology, run the federated security option after database transfer. For the cluster topology, run the federated security option after you create the cluster, but before you add more nodes.

Validation

For this configuration option, the wizard can connect to your LDAP directory and validate the information that you enter in the wizard. By default, validation is enabled. On the Security Settings panel, you can choose to turn validation on or off by selecting **Yes** or **No** to the **Validate LDAP user registry entries** option. Select **No** if you know that your parameters are correct and that your LDAP server is unavailable at the time of creating your instructions.

Two types of validation are performed when you select to validate settings including field syntax and LDAP connection validations. The syntax validation, for example, checks that you entered a valid port number in the range of 1 - 65535. The connection validation, for example, checks that a connection can be made to your LDAP server.

Enabling validation is recommended because it can prevent a possible failure by validating your entries in the wizard before you run the configuration. The LDAP settings including the **Repository ID**, **Host name**, **Port**, **Bind DN**, **Bind password**, **Base DN**, **Administrator group DN**, **Administrator DN**, and **Administrator password** are all validated before the wizard creates your instructions to run the configuration. Review the following worksheet section to see which fields are required based on your selections in the wizard.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Worksheet

When you set up the federated security, you answer questions about your wanted configuration. Some fields apply to all federated security configurations. Some fields are required based on your environment. The remaining fields are advanced and do not apply to most configurations.

Minimal required fields

The following table lists the fields that are unique to the LDAP configuration. You might be prompted for additional information about system or user IDs and passwords that you defined during the portal installation process.

Attention: The Enable Federated Security option modifies the `wimconfig.xml` file. Make a backup copy of this file before you run any of the configuration tasks.
wp_profile_root/config/cells/*CellName*/wim/config/wimconfig.xml

Table 70. Minimal required fields

Field Label	Property	Your Value
LDAP Repository ID	**federated.ldap.id**	
LDAP host name	**federated.ldap.host**	
LDAP port	**federated.ldap.port**	
Bind DN **Restriction:** The following parameters must be unique to your environment: • **PortalAdminId**: this parameter is the user ID that you enter in the **Administrator user ID** field during the installation • **Bind DN** • **Administrator DN from LDAP**	**federated.ldap.bindDN**	
Bind password	**federated.ldap.bindPassword**	

Table 71. Optional fields

Field Label	Property	Your Value
Base DN **Note:** This field is optional. However, it is recommended that you enter a Base DN that matches your LDAP settings. If you are using a Domino LDAP, and you do not have a Base DN defined, then you can leave this field blank.	**federated.ldap.baseDN**	

Use an administrator from your LDAP

If you select to use an administrator from your LDAP server, then you must provide additional information about the LDAP group and ID.

Table 72. Use an administrator and administrator group in your LDAP

Field Label	Property	Your Value
Administrator group DN from LDAP	**newAdminGroupId**	

Table 72. Use an administrator and administrator group in your LDAP (continued)

Field Label	Property	Your Value
Administrator DN from LDAP **Restriction:** The following parameters must be unique to your environment: • **PortalAdminId**: this parameter is the user ID that you enter in the **Administrator user ID** field during the installation • **Bind DN** • **Administrator DN from LDAP**	**newAdminId**	
Administrator password from LDAP	**newAdminPw**	
Default parent for group	**groupParent**	
Default parent for PersonAccount	**personAccountParent**	

Advanced fields

Click **Advanced** on the Customize Values page to see the advanced properties. Default values are provided for advanced fields that are required.

Table 73. Advanced files

Field Label	Property	Your Value
LDAP group objectclasses	**federated.ldap.et.group.objectClass**	
LDAP group objectclasses for creating groups	**federated.ldap.et.group.objectClassForCreate**	
LDAP group search bases	**federated.ldap.et.group.searchBases**	
LDAP PersonAccount objectclasses	**federated.ldap.et.personaccount.objectClasses**	
LDAP PersonAccount objectclasses for creating users	**federated.ldap.et.personaccount.objectClassesForCreate**	
LDAP search bases for the PersonAccount	**federated.ldap.gm.personaccount.searchBases**	
Group dummy member	**federated.ldap.gm.dummyMember**	
Group member attribute	**federated.ldap.gm.groupMemberName**	
Group object class	**federated.ldap.gm.objectClas**	
GM member attribute scope	**federated.ldap.gm.scope**	
Membership attribute name	**federated.ldap.gc.name**	
GC member attribute scope	**federated.ldap.gc.scope**	
Certificate filter	**federated.ldap.certificateFilter**	
Certificate map mode	**federated.ldap.certificatMapMode**	
Group RDN attribute	**groupRdnProperties**	
PersonAccount RDN attribute	**personAccoutnRdnProperties**	
Application server SSL configuration	**federated.ldap.sslConfiguration**	

Nested or dynamic group support

If you need nested group support, then the wizard provides default values for some of the advanced fields. The default values are based on your LDAP server selection. You must click **Advanced** to see the fields if you want to verify the defaults. Nested or dynamic group support fields include **Group member**

attribute, **Membership attribute name**, **LDAP group objectclasses**, and **GC member attribute scope**.

Enabling federated security

After you answer questions and provide information about your LDAP, the wizard generates a custom configuration procedure.

About this task

Depending on your environment, the wizard generates a configuration process. The following steps reflect all possible steps in the configuration process. The steps do not represent a literal configuration. The steps are provided as a reference.

If you click **View Step Command**, you can see the task and properties that are associated with each step in the wizard.

Procedure

1. Manual Step: Retrieve the SSL certificate from the SSL port.

   Condition
   Select to configure SSL enabled LDAP.

   ConfigEngine task
   None

2. CF07 Create a backup of the WebSphere Portal Express profile before modifying cell security.

   Note: The backup is created in `/opt/IBM/WebSphere/AppServer/profiles/cw_profile/`.

   Condition
   None

   ConfigEngine task
   None

3. Validate your LDAP server settings.

   Condition
   None

   ConfigEngine task
   validate-federated-ldap

4. Add an LDAP user registry to the default federated repository.

   Condition
   None

   ConfigEngine task
   wp-create-ldap

   recycle-dmgr-if-cluster

5. Register the WebSphere Application Server scheduler tasks.

   Condition
   None

   ConfigEngine task
   stop-portal-server

   start-portal-server

reregister-scheduler-tasks

6. Replace the file-based WebSphere Portal and WebSphere Application Server users and groups with users and groups from your LDAP server.

   Condition
   Select to use an administrator and administrator group that is stored in your LDAP.

   ConfigEngine task
   wp-change-portal-admin-user

   wp-change-was-admin-user

7. Update the user registry where new users and groups are stored.

   Condition
   None

   ConfigEngine task
   wp-set-entitytypes

8. Recycle the servers after a security change.

   Condition
   None

   ConfigEngine task
   recyle-servers-after-security-change

9. Update the search administration user.

   Condition
   Select to use an administrator and administrator group that is stored in your LDAP.

   ConfigEngine task
   start-portal-server

   action-fixup-after-security-change-portal-wp.search.webscanner

10. After you change the security model, the servers need to be restarted. Restart the portal server.

    Condition
    None

    ConfigEngine task
    recycle-servers-after-security-change

    start-portal-server

11. Verify that all defined attributes are available in the configured LDAP user registry.

    Condition
    None

    ConfigEngine task
    wp-validate-federated-ldap-attribute-config

12. Manual Step: Update the appropriate `MemberFixerModule.properties` file with the values for your LDAP users.

    Condition
    Select to use an administrator and administrator group that is stored in your LDAP.

    ConfigEngine task
    None

13. Run the member fixer tool.

    Condition
    Select to use an administrator and administrator group that is stored in your LDAP.

    ConfigEngine task
    `run-wcm-admin-task-member-fixer`

14. Restart the WebSphere Portal Server.

    Condition
    None

    ConfigEngine task
    `stop-portal-server`

    `start-portal-server`

15. Manual Step: Map attributes to ensure proper communication between WebSphere Portal and the LDAP server.

    Condition
    None

    ConfigEngine task
    None

What to do next

If you are setting up a stand-alone server environment, you can now explore your site.

You can also use the **Create an Additional Cluster Node** option to add more nodes to your cluster. Your security updates to the primary node are applied to the nodes that you add to your cluster.

Adding more attributes to VMM

After you install IBM WebSphere Portal Express and configuring your LDAP user registries, you must adapt the attribute configuration to match the configured LDAP servers and your business needs. However, do not complete these steps if you configured only a database user registry or the default federated file-based repository for out-of-box installations.

About this task

After installation, IBM WebSphere Portal Express has a predefined set of attributes for users and groups. Your LDAP server might have a different set of predefined user and group attributes. To ensure communication between WebSphere Portal Express and your LDAP server, you can configure extra attributes. Flag the attributes as required or unsupported on a per repository basis or for all configure repositories.

LDAP servers can handle only attributes that are explicitly defined in their schema. The LDAP schema is different from the WebSphere Portal Express schema. The two schemas are required to match for communication between WebSphere Portal Express and the LDAP server. The task to add the LDAP user registry does some basic attribute configurations that depend on the LDAP server. You might still have to adapt the WebSphere Portal Express configuration to match the LDAP schema.

If an attribute is in WebSphere Portal Express but not in the LDAP server, complete one of the following tasks to resolve this mismatch:

- Flag the attribute as unsupported for the LDAP server
- Introduce an attribute mapping that maps the WebSphere Portal Express attribute to an attribute defined in the LDAP schema

Use the following tasks to adapt the attribute configuration to match the configured LDAP servers and your business needs:

1. "Querying the defined attributes"
   After you install IBM WebSphere Portal Express and configure your LDAP user registries, query the defined attributes. The task defines a list of attributes that are flagged as unsupported or mapped to a different LDAP attribute.
2. "Adding attributes"
   The VMM is configured with a default attribute schema that might not be compatible with your LDAP server. Add attributes to extend the VMM attribute schema and then map them between IBM WebSphere Portal Express and your user registry.
3. "Mapping attributes" on page 573
   Map the attributes between IBM WebSphere Portal Express and your LDAP user registries.
4. "Removing attributes" on page 575
   The Virtual Member Manager (VMM) has a limitation. There is no task to update an attribute.

Related reference:

"People Finder" on page 928
The **People Finder** portlet provides both quick search and advanced search options for locating people and information about people. Once found, a person is visible to other users as a person link that indicates online presence and displays a menu of instant messaging and other options.

Querying the defined attributes

After you install IBM WebSphere Portal Express and configure your LDAP user registries, query the defined attributes. The task defines a list of attributes that are flagged as unsupported or mapped to a different LDAP attribute.

Procedure

1. Open a command prompt.
2. Change to the *wp_profile_root*`/ConfigEngine` directory.
3. Run the following task:
   - Linux : `./ConfigEngine.sh wp-query-attribute-config -DWasPassword=`*`password`*
   - IBM i: `ConfigEngine.sh wp-query-attribute-config -DWasPassword=`*`password`*
   - Windows: `ConfigEngine.bat wp-query-attribute-config -DWasPassword=`*`password`*

   Note: This task does not validate the existence of attributes in the LDAP schema.

Adding attributes

The VMM is configured with a default attribute schema that might not be compatible with your LDAP server. Add attributes to extend the VMM attribute schema and then map them between IBM WebSphere Portal Express and your user registry.

About this task

You can add multiple attributes at one time by creating an XML file that includes the properties and attributes for each property. The XML file is referred to as a *deployment file*. Create the XML file before you start this procedure.

The following sample is an XML deployment file that is used to add three properties.

```
<wplc-add-property>
 <resource propertyName="attribute_name_1" dataType="Int" entityTypes="Group" multiValued="true" />
 <resource propertyName="attribute_name_2" dataType="String" entityTypes="PersonAccount" multiValue
 <resource propertyName="attribute_name_3" dataType="Base64Binary" entityTypes="Group,PersonAccount
</wplc-add-property>
```

The resource tag includes attributes that are specific for the property: **propertyName**, **dataType**, **entityType**, and **multiValued**.

Procedure

1. This task requires server connections.
   - In a stand-alone environment, ensure that the WebSphere_Portal server is running.
   - In a clustered environment, stop all application servers on the system. Ensure that the WebSphere_Portal server is stopped. Then, start the node agent and deployment manager servers.
2. Install the enterprise archive (`.ear`) file on WebSphere Application Server:
   a. Open a command line.
   b. Change to the *wp_profile_root*`/ConfigEngine` directory.
   c. Run the following task:

Table 74. Stand-alone and cluster tasks to install the enterprise archive (EAR) file. Use the **wp-la-install-ear** task to install the EAR.

Environment	Task
Stand-alone environment	• Linux : `./ConfigEngine.sh wp-la-install-ear -DWasPassword=`*password* • IBM i: `ConfigEngine.sh wp-la-install-ear -DWasPassword=`*password* • Windows: `ConfigEngine.bat wp-la-install-ear -DWasPassword=`*password*

Table 74. Stand-alone and cluster tasks to install the enterprise archive (EAR) file (continued). Use the **wp-la-install-ear** task to install the EAR.

Environment	Task
Clustered environment	• Linux : `./ConfigEngine.sh wp-la-install-ear -DWasPassword=`*`dmgr_password`* `-DServerName=`*`dmgr_server_name`* `-DNodeName=`*`node_name`* • IBM i: `ConfigEngine.sh wp-la-install-ear -DWasPassword=`*`dmgr_password`* `-DServerName=`*`dmgr_server_name`* `-DNodeName=`*`node_name`* • Windows: `ConfigEngine.bat wp-la-install-ear -DWasPassword=`*`dmgr_password`* `-DServerName=`*`dmgr_server_name`* `-DNodeName=`*`node_name`* Where the default value for *`dmgr_server_name`* is `dmgr`. You can find the *`dmgr_server_name`* value in the WebSphere Integrated Solutions Console. Go to **System administrator** > **Deployment Manager** > **Configuration tab** > **General Properties** > **Name**. Where *`node_name`* is the name of the node where the deployment manager is located. You can find the *`node_name`* value in the WebSphere Integrated Solutions Console. Go to **System administrator** > **Deployment Manager** > **Runtime tab** > **General Properties** > **Node Name**.

3. Stop and restart the appropriate servers to propagate the changes. For instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.
4. Use a text editor to open the `wkplc.properties` file in the *wp_profile_root*`/ConfigEngine/properties` directory.
5. Find the *VMM Property Extension Properties* heading. You can either add a single attribute or multiple attributes.
   - To add a single attribute, update the following properties with information about the property to add.

     `la.providerURL`

     Description

     The remote endpoint where your portal server or deployment manager installation is available. Check the value for hostname:port. The port points to the BOOTSTRAP_ADDRESS port of either the WebSphere_Portal server or the deployment manager. The deployment manager is used in a clustered environment.

     `la.propertyName`

Description
The name of the property that you are adding.

la.entityTypes

Description
This value is a list of entity types that the new property is applicable to. If you need to enter multiple values, use a comma to separate each value, for example "value1,value2".

Valid values
Group

PersonAccount

la.dataType

Description

Defines the type of data that is stored in the attribute that is being created. If this attribute is mapped to LDAP, this data type must match the corresponding attribute type in LDAP. Consult your LDAP administrator if you are unsure of the data types in LDAP. If this attribute is stored in the VMM property extension database, the data type must match the corresponding attribute type as defined in VMM's database.

While it is possible to add attributes of different types to VMM, the Registration/Edit My Profile Portlet is only capable of working with attributes of type String and Int. If you need UI support for other types, you would need your own custom form or portlet that can process those types. Portal does not have a UI that reads or updates group attributes. The one exception is the UI that is used to create a group.

Valid values
String

Int

DateTime

Base64Binary

IdentifierType

Boolean

Long

Double

Short

la.multiValued

Description
Defines if the property can contain multiple values or not.

- To add multiple attributes, update the following properties:

la.providerURL

Description
The remote endpoint where your portal server or deployment manager installation is available. Check the

value for hostname:port. The port points to the BOOTSTRAP_ADDRESS port of either the WebSphere_Portal server or the deployment manager. The deployment manager is used in a clustered environment.

la.deployfile

Description

Use this property when you want to create multiple properties by using a single ConfigEngine operation. Specify the path and name of the XML file that contains the properties that you want to add. You can specify a path that is relative to the ConfigEngine directory or the fully qualified file system path. If you specify a value for this property, do not specify a value for la.propertyName, la.dataType, or la.Multivalued.

The following is a sample of an XML deploy file that is used to add three properties.

```
<wplc-add-property>
   <resource propertyName="attribute_name_1" dataType="Int" entityType
   <resource propertyName="attribute_name_2" dataType="String" entityT
   <resource propertyName="attribute_name_3" dataType="Base64Binary" e
</wplc-add-property>
```

The resource tag includes attributes that are specific for the property: propertyName, dataType, entityType, and multiValued.

6. Save your changes to the `wkplc.properties` file.
7. Run the **wp-add-property** task to add a property that maps to an attribute in your user registry. If you are defining a new property to store in the property extension database, run **wp-add-la-property**.

   Note: This task calls an EJB that must authenticate against WebSphere Application Server. Depending on the configuration in the `sas.client.props` file, you might receive a prompt that asks for your user ID and password. Enter the WebSphere Application Server user ID and password.

   - Linux : `./ConfigEngine.sh wp-add-property -DWasPassword=`*password*
   - IBM i: `ConfigEngine.sh wp-add-property -DWasPassword=`*password*
   - Windows: `ConfigEngine.bat wp-add-property -DWasPassword=`*password*
8. Stop and restart the appropriate servers to propagate the changes.

Mapping attributes

Map the attributes between IBM WebSphere Portal Express and your LDAP user registries.

Procedure

1. Use a text editor to open the `wkplc.properties` file in the *wp_profile_root*`/ConfigEngine/properties` directory.
2. Enter values for the following set of parameters to identify your LDAP server:

   The following parameters are found in the VMM Federated repository properties heading:

   Note: Go to the properties file for specific information about the parameters.

   federated.ldap.id

federated.ldap.host

federated.ldap.port

federated.ldap.sslEnabled

federated.ldap.bindDN

federated.ldap.bindPassword

federated.ldap.baseDN

Note: Make sure you use the same values that you used to configure your LDAP server.

3. Run the following task to check that all defined attributes that are available in the configured LDAP user registry:
   - Linux : `./ConfigEngine.sh wp-validate-federated-ldap-attribute-config -DWasPassword=`*`password`*
   - IBM i: `ConfigEngine.sh wp-validate-federated-ldap-attribute-config -DWasPassword=`*`password`*
   - Windows: `ConfigEngine.bat wp-validate-federated-ldap-attribute-config -DWasPassword=`*`password`*
4. Open the `ConfigTrace.log` file, in the *wp_profile_root*`\log` directory. Review the following output for the **PersonAccount** and **Group** entity type:

   The following attributes are defined in WebSphere Portal Express but not in the LDAP server

   This list contains all attributes that are defined in WebSphere Portal Express but not available in the LDAP. Flag attributes that you do not plan to use in WebSphere Portal Express as unsupported. Map the attributes that you plan to use to the attributes that exist in the LDAP; you must also map the `uid`, `cn`, `firstName`, `sn`, `preferredLanguage`, and `ibm-primaryEmail` attributes if they are contained in the list.

   The following attributes are flagged as required in the LDAP server but not in WebSphere Portal Express

   This list contains all attributes that are defined as "must" in the LDAP server but not as required in WebSphere Portal Express. Flag these attributes as required within WebSphere Portal Express; go to the next step to flag an attribute as either unsupported or required.

   The following attributes have a different type in WebSphere Portal Express and in the LDAP server

   This list contains all attributes that WebSphere Portal Express might ignore because the data type within WebSphere Portal Express and within the LDAP server do not match.
5. Use a text editor to open the `wkplc.properties` file
6. Enter a value for the following set of parameters in the `wkplc.properties` file to correct any issues that are found in the configuration trace file:

   The following parameters are found in the VMM Federated repository properties heading:

   federated.ldap.attributes.nonSupported

   federated.ldap.attributes.nonSupported.delete

   federated.ldap.attributes.mapping.ldapName

   federated.ldap.attributes.mapping.portalName

   federated.ldap.attributes.mapping.entityTypes

The following values flag `certificate` and `members` as unsupported attributes and maps `ibm-primaryEmail` to `mail` and `ibm-jobTitle` to `title` for the `PersonAccount` **`entityTypes`**:

```
federated.ldap.attributes.nonSupported=certificate, members
federated.ldap.attributes.nonSupported.delete=

federated.ldap.attributes.mapping.ldapName=mail, title
federated.ldap.attributes.mapping.portalName=ibm-primaryEmail, ibm-jobTitle
federated.ldap.attributes.mapping.entityTypes=PersonAccount
```

If you want to map attributes for your groups instead of users, set the **`entityTypes`** to `Group`.

`federated.ldap.attributes.mapping.entityTypes=``Group`

7. Save your changes to the `wkplc.properties` file.
8. Run the following task to update the LDAP user registry configuration with the following items:

   A list of unsupported attributes

   The correct mapping between WebSphere Portal Express and the LDAP user registry

   - Linux : `./ConfigEngine.sh wp-update-federated-ldap-attribute-config -DWasPassword=`*`password`*
   - IBM i: `ConfigEngine.sh wp-update-federated-ldap-attribute-config -DWasPassword=`*`password`*
   - Windows: `ConfigEngine.bat wp-update-federated-ldap-attribute-config -DWasPassword=`*`password`*
9. Stop and restart the appropriate servers to propagate the changes. For specific instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.
10. Optional: Complete the following steps to flag an attribute as either unsupported or required for the entire WebSphere Portal Express environment instead of just for the specified LDAP:
    a. Enter a value for the following required parameters in the `wkplc.properties` file:

       Note: Go to the properties file for specific information about the parameters.

       `user.attributes.required`

       `user.attributes.nonsupported`
    b. Save your changes to the `wkplc.properties` file.
    c. Run the following task:
       - Linux : `./ConfigEngine.sh wp-update-attribute-config -DWasPassword=`*`password`*
       - IBM i: `ConfigEngine.sh wp-update-attribute-config -DWasPassword=`*`password`*
       - Windows: `ConfigEngine.bat wp-update-attribute-config -DWasPassword=`*`password`*
    d. Stop and restart all necessary servers to propagate your changes.

Removing attributes

The Virtual Member Manager (VMM) has a limitation. There is no task to update an attribute.

About this task

Remove an attribute for any of the following circumstances:

- You added an attribute to a property extension database that was spelled incorrectly
- You adapted an attribute to match your LDAP server that was spelled incorrectly
- Your migration added the attribute

Use caution when you do these steps.

Important: Do not remove attributes that are populated with user values because it can cause inconsistencies.

Cluster note: In an idle-standby environment, complete these steps on the deployment manager and then resynch the nodes.

Procedure

1. Before you configure security, use the IBM WebSphere Application Server **backupConfig** task to create and store a backup of the IBM WebSphere Portal Express configuration. Read backupConfig command for information.
2. Complete the following steps to remove an attribute that is stored in a property extension database:
   a. Open the tool that you use to edit your database.
   b. Verify that your attribute name is available in the LAPROP table.
   c. Delete the required attributes from the LAPROP table.
   d. Open the `wimxmlextension.xml` file in the *dmgr_profile_root*`/config/cells/`*cellname*`/wim/model` directory.
   e. Locate and delete the **propertySchema** definition for the attributes that you deleted from the LAPROP table. For example:

```
<wim:propertySchema nsURI="http://www.ibm.com/websphere/wim" dataType="String"
    multiValued="true" propertyName="attribute_name">
  <wim:applicableEntityTypeNames>PersonAccount</wim:applicableEntityTypeNames>
</wim:propertySchema>
```

   f. Save your changes to the `wimxmlextension.xml` file.
   g. Open the `wimconfig.xml` file in the *dmgr_profile_root*`/config/cells/`*cellname*`/wim/config` directory.
   h. Locate and delete the **attributes** or **propertiesNotSupported** definitions for the attributes that you deleted from the LAPROP table. For example:

```
<config:attributes name="attribute_name" propertyName="attribute_name">
<config:entityTypes> PersonAccount </config:entityTypes>
<config:entityTypes> Group </config:entityTypes>
</config:attributes>
```

   or

```
<config:propertiesNotSupported name="attribute_name">
```

   i. Save your changes to the `wimconfig.xml` file.
   j. Stop and restart all the deployment manager, node agents, and WebSphere_Portal server to propagate the changes.
3. Complete the following steps to remove an attribute that is not stored in a property extension database:
   a. Open the `wimxmlextension.xml` file.
   b. Locate and delete the **propertySchema** definition for the attributes you previously added. For example:

```
<wim:propertySchema nsURI="http://www.ibm.com/websphere/wim" dataType="String"
        multiValued="true" propertyName="attribute_name">
   <wim:applicableEntityTypeNames>PersonAccount</wim:applicableEntityTypeNames>
    </wim:propertySchema>
```

c. Save your changes to the `wimxmlextension.xml` file.

d. Open the `wimconfig.xml` file.

e. Locate and delete the stanza that corresponds to the custom attribute you deleted from the `wimextension.xml` file. For example:

```
<config:attributes name="attribute_name" propertyName="property_name">
            <config:entityTypes>PersonAccount</config:entityTypes>
 </config:attributes>
```

f. Save your changes to the `wimconfig.xml` file.

g. Stop and restart the WebSphere_Portal server.

Related tasks:

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

Enabling application groups

You can define user groups within the database user registry with members (users or groups) contained in the federated LDAP user registry you configured with application groups. The benefit of application groups is that you can create groups that are only used in IBM WebSphere Portal Express.

Before you begin

Before you enable application groups, from the Configuration Wizard run the Enable Federated Security configuration option to add all required federated LDAP user registries. Then, add all required database user registries. You must also set the `Group entity` type to the database user registry and the `Person entity` type to the LDAP user registry.

About this task

You can use application groups in the following scenarios:

Read-only LDAP
: If you have a read-only LDAP, you cannot change the group membership of users and groups. If you need to define access rights for certain users that are in different groups, you can create an application group for these users with the required access rights.

Special group setup for WebSphere Portal Express
: In this scenario, you need to set up a special group hierarchy that is used only by WebSphere Portal Express and not by other applications that access your LDAP server. This set up can help you apply special access control rules just for WebSphere Portal Express because the roles assigned to the application group also apply to all of its members.

Note: Application groups apply only to WebSphere Portal Express; it does not apply to external security managers. Also, application groups are not supported when you use the built-in file repository.

Perform the following steps to enable application groups:

Procedure

1. Run the following task to enable application groups:

Table 75. Task to enable application groups by operating system

Operating system	Task
IBM i	`ConfigEngine.sh wp-update-group-repository-relationship -DWasPassword=`*`password`* `-Drepository.id=`*`ldapid`* `-Drepository.forgroups=`*`dbid`* from the *wp_profile_root*`/ConfigEngine` directory
Linux	`./ConfigEngine.sh wp-update-group-repository-relationship -DWasPassword=`*`password`* `-Drepository.id=`*`ldapid`* `-Drepository.forgroups=`*`dbid`* from the *wp_profile_root*`/ConfigEngine` directory
Windows	`ConfigEngine.bat wp-update-group-repository-relationship -DWasPassword=`*`password`* `-Drepository.id=`*`ldapid`* `-Drepository.forgroups=`*`dbid`* from the *wp_profile_root*`\ConfigEngine` directory

   When you run the `wp-create-ldap` task, *ldapid* is the value that is specified in **`federated.ldap.id`** and when you run the `wp-create-db` task, the *dbid* is the value that is specified in **`federated.db.id`**

2. Stop and restart the WebSphere_Portal server.

Advanced group configurations

It is possible to use IBM WebSphere Portal Express ConfigEngine helper tasks to set up advanced Virtual Member Manager (VMM) group configurations. Specifically, it is possible to configure VMM to understand and use the "Group membership attribute" that many directories support.

Background: Group membership attribute

A group membership attribute is an LDAP directory feature. It allows an LDAP client to ask the LDAP directory for a list of groups that the user is a member of. It is as if the list is an attribute of the user object. It does not query the various groups to see whether a user is a member of any of them.

It is beyond the scope of this document to describe in detail all the various ways that different LDAP directory servers support a group membership attribute. Most LDAP directories have some variant of this support: Active Directory has the **`memberOf`** attribute. Sun/Oracle has **`nsroles`** or **`isMemberOf`** based on the version. IBM Directory Server has the **`ibm-allGroups`** attribute.

Many LDAP directories support complex group membership algorithms that include nesting of groups as members within other groups. They also support the use of dynamic group membership by querying against user attributes, instead of static lists of members. While VMM supports these features, in many cases it requires multiple operations between VMM and the directory to do so and thus becomes inefficient and a performance bottleneck. The use of the group

membership attribute shifts the burden of the calculations of these complex group structures to the directory itself, where it can be managed much more efficiently.

If your LDAP directory supports a group membership attribute and your use cases include group nesting or the use of dynamic groups, configure VMM to use the group membership attribute. Most LDAP directories implement this group membership attribute as a "real-time" attribute. It is calculated on demand rather than pre-calculated and stored as a persistent attribute of the user. There is likely extra processor cost on the LDAP directory for using this support. Usually this processor cost is worth the performance benefit that can be gained from offloading the work of resolving complex group relationships from VMM.

Configuring VMM to use the group membership attribute

To configure VMM to use the group membership attribute, two things must be specified:

- The name of the group membership attribute
- The scope of the attribute. This information tells VMM how complete the response is to a request for the group membership attribute value for a user.

The name is the attribute name in the LDAP directory implementation. For example, this attribute is **ibm-allGroups** for the IBM Directory Server. This attribute is set on the **federated.ldap.gc.name** property in the `wkplc.properties` file.

The scope is set on the **federated.ldap.gc.scope** property and is one of three possible values: `direct`, `nested`, or `all`. The setting for this property depends on how your LDAP directory implements the group membership attribute.

direct Direct means that the value returned holds only the list of static groups of which the user is a direct member. It does not attempt to account for group nesting or dynamic group memberships. The response here is functionally equivalent, for example, to a query of the form `(&(objectClass=groupOfNames)(member=<dn of the user>))`. In this case, there is little if any reason to prefer the group membership attribute over the traditional query.

Note: When the group membership attribute scope is `direct` or when you use the traditional query method, VMM must do extra work if it needs to resolve nested groups or dynamic groups.

- VMM tries to resolve dynamic groups if the dynamic group configuration information is set up within the VMM configuration files.
- VMM tries to resolve nested groups if the client application, which is WebSphere Portal Express, requests it to. By default, WebSphere Portal Express requests that nested groups are used. If your access control models do not use group nesting to inherit permissions, turn off the nested group function within portal. Read the documentation for the **enableNestedGroups** custom property within the **WP AccessControlDataManagementService** Resource Environment Provider.

Note: To avoid conflicts between how WebSphere Portal Express and WebSphere Application Server handle nested groups, globally turn off nested groups. Change the **com.ibm.ws.wim.registry.grouplevel** value to 1. Go to Disabling nested group searches for instructions.

nested Nested means that the response from the LDAP server to a request for the group membership attribute already includes any nested group

relationships, but not any dynamic group memberships. If the user is a member of group "A2" and "A2" is a member of group "A1", then the list of group memberships includes both A1 and A2. This information tells VMM that even if a client requests nested group information, the response already provides it. No further work needs to be done by VMM to satisfy the request.

all All means that the response from the LDAP server to a request for the group membership attribute already includes both nested groups and also dynamic groups, if any.

It is important to set the scope value to accurately reflect how your LDAP directory works to get correct and efficient operation. It is beyond the scope of this documentation to describe the unique characteristics of every directory. In some cases, the directory might require specific setup to fully support the advanced group features. For example, the IBM Directory Server must be set up with specific auxiliary object classes and special membership records to fully support nested groups and dynamic groups with the **ibm-allGroups** attribute. Consult your specific LDAP directory documentation or check with your LDAP administrator for specific details about your directory deployment.

ConfigEngine tasks for advanced group configuration

Set the **federated.ldap.gc.name** and **federated.ldap.gc.scope** properties before you run one of the following tasks:

- **wp-create-ldap**
- **wp-create-ldap-groupconfig**

  "Updating the group membership configuration"
  The initial federated repositories setup might not include the advanced set up for the group membership attribute. You can configure the group membership attribute after the initial setup. Specify the properties in the `wkplc.properties` file and then run the **wp-create-ldap-groupconfig** task manually.

  "Configuring WebSphere Portal Express to use dynamic groups" on page 581
  By default, IBM WebSphere Portal Express is enabled for static groups. However, the Virtual Member Manager (VMM) allows users to be members of either static or dynamic groups. Static groups have a persistent binding between a group and its members. Dynamic groups have a search query that is defined to retrieve the members of a group.

Updating the group membership configuration

The initial federated repositories setup might not include the advanced set up for the group membership attribute. You can configure the group membership attribute after the initial setup. Specify the properties in the `wkplc.properties` file and then run the **wp-create-ldap-groupconfig** task manually.

Before you begin

In a stand-alone server environment, you can complete the following task when the servers are either stopped or started. In a clustered environment, start the deployment manager and node agent. Then, verify that they are able to synchronize.

Procedure

1. Go to the *wp_profile_root*`/ConfigEngine/properties` directory.
2. Open the `wkplc.properties` file with a text editor.

3. Update the following parameters in the `wkplc.properties` file under the VMM LDAP group member config heading:

   Note: Go to the properties file for specific information about the parameters.

   `gc.ldap.id`

   `gc.name`

   `gc.updateGroupMembership`

   `gc.scope`
4. Open a command prompt.
5. Change to the *wp_profile_root*`/ConfigEngine` directory.
6. Run the following task to update the group membership for the LDAP user registry:
   - Linux : `./ConfigEngine.sh wp-create-ldap-groupconfig -DWasPassword=`*`password`*
   - IBM i: `ConfigEngine.sh wp-create-ldap-groupconfig -DWasPassword=`*`password`*
   - Windows: `ConfigEngine.bat wp-create-ldap-groupconfig -DWasPassword=`*`password`*
7. Stop and restart the appropriate servers to propagate the changes. For instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.

Configuring WebSphere Portal Express to use dynamic groups

By default, IBM WebSphere Portal Express is enabled for static groups. However, the Virtual Member Manager (VMM) allows users to be members of either static or dynamic groups. Static groups have a persistent binding between a group and its members. Dynamic groups have a search query that is defined to retrieve the members of a group.

About this task

Disclaimer: The following procedure is to activate dynamic group support natively in VMM. It is preferable, if possible, to use the group membership attribute support to bring in dynamic group information from your LDAP server. Using the group membership attribute support for dynamic group membership is only possible if:

- Your LDAP server supports a group membership attribute
- The group membership attribute includes dynamic group membership information

Otherwise, complete the following steps to manually configure WebSphere Portal Express to use dynamic groups.

Clustered environments: Complete the following steps on the Deployment Manager and then synchronize the nodes.

Procedure

1. Log in to the WebSphere Integrated Solutions Console.
2. Go to **Security** > **Global security**.
3. In **Available realm definitions**, select **Federated repositories** and click **Configure**.
4. In **Related Items**, click **Manage repositories**.

5. Select the appropriate repository from the list.
6. In **Additional Properties**, click **Group attribute definition** and then click **Dynamic member attributes**.
7. Click **New** and specify values for the **Name** and **Object class** fields as appropriate. For example:
   - **Name:** `memberurl`
   - **Object class:** `groupofurls`
8. Click **OK** and save the changes to the master configuration.
9. Stop and restart the appropriate servers to propagate the changes. For instructions, read "Starting and stopping servers, deployment managers, and node agents" on page 1216.

Adding realm support

A realm is a group of users from one or more user registries that form a coherent group within IBM WebSphere Portal Express. Realms allow flexible user management with various configuration options. A realm must be mapped to a Virtual Portal to allow the defined users to log in to the Virtual Portal. When you configure realm support, complete these steps for each base entry that exists in your LDAP and database user registry to create multiple realm support.

Before you begin

Before you configure realm support, add all LDAP user registries and database user registries to the federated repository. To create multiple realms, you must create all required base entries within your LDAP user registries and database user registries. All base entry names must be unique within the federated repository.

In a stand-alone server environment, you can complete this task when the servers are either stopped or started. In a clustered environment, start the deployment manager and node agent and verify that they are able to synchronize.

Procedure

1. Use the IBM WebSphere Application Server **`backupConfig`** task to create and store a backup of the IBM WebSphere Portal Express configuration. Read backupConfig command for information.
2. Use a text editor to open the `wkplc.properties` file in the *wp_profile_root*`/ConfigEngine/properties` directory.
3. Required: Enter a value for the following parameters in the VMM realm configuration section:

   Note: Review the properties file for specific information about the parameters.

   `realmName`

   `securityUse`

   `delimiter`

   `addBaseEntry`
4. Save your changes.
5. Open a command line and change to the *wp_profile_root*`/ConfigEngine` directory.
6. Run the following task to add a realm to the Virtual Member Manager configuration:

Important: To create multiple realms, ensure that your federated repository contains the correct unique base entries. Stop and restart the appropriate servers for your installation environment, and then update the `wkplc.properties` file with the base entry information and rerun the **wp-create-realm** task. Repeat these steps until all realms are created.

- Linux :`./ConfigEngine.sh wp-create-realm -DWasPassword=`*`password`*
- IBM i: `ConfigEngine.sh wp-create-realm -DWasPassword=`*`password`*
- Windows: `ConfigEngine.bat wp-create-realm -DWasPassword=`*`password`*

7. Stop and restart the appropriate servers to propagate the changes. For instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.
8. Required: Enter a value for the following parameters in the `wkplc.properties` file in the VMM realm configuration section:

   realmName

   realm.personAccountParent

   realm.groupParent

   realm.orgContainerParent
9. Run the following task to update the default parents per entity type and realm:
   - Linux :`./ConfigEngine.sh wp-modify-realm-defaultparents -DWasPassword=`*`password`*
   - IBM i: `ConfigEngine.sh wp-modify-realm-defaultparents -DWasPassword=`*`password`*
   - Windows: `ConfigEngine.bat wp-modify-realm-defaultparents -DWasPassword=`*`password`*
10. Stop and restart the appropriate servers to propagate the changes. You can rerun the **wp-modify-realm-defaultparents** task to create more entity types and realms.
11. Optional: Complete the following steps to add more base entries to the realm configuration:

    For example, you have two more base entries (base entry 1 and base entry 2) to add to the realm you created. You must update the `wkplc.properties` file with the information from base entry 1 and then run this task. Then, update the properties file with the information for base entry 2 and then run this task.

    a. Enter a value for the following parameters in the `wkplc.properties` file in the VMM realm configuration section:

       realmName

       addBaseEntry

    b. Run the following task to add more LDAP base entries to the realm configuration:
       - Linux :`./ConfigEngine.sh wp-add-realm-baseentry -DWasPassword=`*`password`*
       - IBM i: `ConfigEngine.sh wp-add-realm-baseentry -DWasPassword=`*`password`*
       - Windows: `ConfigEngine.bat wp-add-realm-baseentry -DWasPassword=`*`password`*

    c. Stop and restart all necessary servers to propagate your changes.
12. Optional: Complete the following steps to replace the WebSphere Application Server and WebSphere Portal Express administrator user ID:

Tip: Complete these steps if you changed the default realm.

a. Create a user in the **Manage Users and Groups** portlet to replace the current WebSphere Application Server administrative user.

b. Create a user in the **Manage Users and Groups** portlet to replace the current WebSphere Portal Express administrative user.

c. Create a group in the **Manage Users and Groups** portlet to replace the current group.

d. Run the following task to replace the old WebSphere Application Server administrative user ID and group ID with the new user and group:

- Linux :`./ConfigEngine.sh wp-change-was-admin-user -DWasUser=`*`adminid`* `-DWasPassword=`*`password`* `-DnewAdminId=`*`newadminid`* `-DnewAdminPw=`*`newpassword`* `-DnewAdminGroupId=`*`newadmingroupid`*
- IBM i: `ConfigEngine.sh wp-change-was-admin-user -DWasUser=`*`adminid`* `-DWasPassword=`*`password`* `-DnewAdminId=`*`newadminid`* `-DnewAdminPw=`*`newpassword`* `-DnewAdminGroupId=`*`newadmingroupid`*
- Windows: `ConfigEngine.bat wp-change-was-admin-user -DWasUser=`*`adminid`* `-DWasPassword=`*`password`* `-DnewAdminId=`*`newadminid`* `-DnewAdminPw=`*`newpassword`* `-DnewAdminGroupId=`*`newadmingroupid`*

e. Verify that the task completed successfully. Stop and restart all servers.

f. Run the following task to replace the old WebSphere Portal Express administrative user ID and group ID with the new user and group:

- Linux :`./ConfigEngine.sh wp-change-portal-admin-user -DWasPassword=`*`password`* `-DnewAdminId=`*`newadminid`* `-DnewAdminPw=`*`newpassword`* `-DnewAdminGroupId=`*`newadmingroupid`*
- IBM i: `ConfigEngine.sh wp-change-portal-admin-user -DWasPassword=`*`password`* `-DnewAdminId=`*`newadminid`* `-DnewAdminPw=`*`newpassword`* `-DnewAdminGroupId=`*`newadmingroupid`*
- Windows: `ConfigEngine.bat wp-change-portal-admin-user -DWasPassword=`*`password`* `-DnewAdminId=`*`newadminid`* `-DnewAdminPw=`*`newpassword`* `-DnewAdminGroupId=`*`newadmingroupid`*

Important: You must provide the full distinguished name (DN) for the **`newAdminId`** and **`newAdminGroupId`** parameters.

Additional parameter for stopped servers: This task verifies the user against a running server instance. If the server is stopped, add the **`-Dskip.ldap.validation`**`=true` parameter to the task to skip the validation.

g. Verify that the task completed successfully. Stop and restart all servers.

13. Complete the following steps to set the realm you created as the default realm:

Remember: Only users that are defined in base entries that exist in the default realm are able to log in to WebSphere Portal Express. If a user cannot log in to WebSphere Portal Express, check whether the base entry that contains the user exists in the default realm. You can run the **`wp-query-realm-baseentry`** task to see what base entries are part of the default realm. If the default realm is missing the base entry, run the **`wp-add-realm-baseentry`** task to add the base entry to the default realm.

a. Open the `wkplc.properties` file.

b. For **`defaultRealmName`**, type the **`realmName`** property value you want to use as the default realm.

c. Save your changes.

d. Run the following task to set this realm as the default realm:
   - Linux :./ConfigEngine.sh wp-default-realm -DWasPassword=*password*
   - IBM i: ConfigEngine.sh wp-default-realm -DWasPassword=*password*
   - Windows: ConfigEngine.bat wp-default-realm -DWasPassword=*password*

e. Stop and restart all necessary servers to propagate your changes.

14. Complete the following steps to query a realm for a list of its base entries:
    a. Open the wkplc.properties file.
    b. For **realmName**, type the name of the realm you want to query.
    c. Save your changes.
    d. Run the following task to list the base entries for a specific realm:
       - Linux :./ConfigEngine.sh wp-query-realm-baseentry -DWasPassword=*password*
       - IBM i: ConfigEngine.sh wp-query-realm-baseentry -DWasPassword=*password*
       - Windows: ConfigEngine.bat wp-query-realm-baseentry -DWasPassword=*password*

15. Optional: Complete the following steps to enable the full distinguished name login if the short names are not unique for the realm:

    Tip: Run this task if the administrator name is in conflict with another user name in the attached repository. This command allows the Administrator to log in using the fully distinguished name instead of the short name.
    a. Open the wkplc.properties file.
    b. Enter a value for **realmName** or leave blank to update the default realm.
    c. Save your changes.
    d. Run the following task to list the base entries for a specific realm:
       - Linux :./ConfigEngine.sh wp-modify-realm-enable-dn-login -DWasPassword=*password*
       - IBM i: ConfigEngine.sh wp-modify-realm-enable-dn-login -DWasPassword=*password*
       - Windows: ConfigEngine.bat wp-modify-realm-enable-dn-login -DWasPassword=*password*

       Note: You can run the **wp-modify-realm-disable-dn-login** task to disable the feature.
    e. Stop and restart all necessary servers to propagate your changes.

Updating your user registry

After you deploy IBM WebSphere Portal Express, you can adjustment your federated user repository configurations. You can update these configurations to achieve the correct user registry configuration.

About this task

Choose from the following tasks to update your user registry configurations:

"Updating the context pool configuration" on page 587
After you configure your LDAP user registry, you can adjust the number of context instances that the context pool concurrently maintains to improve performance.

Updating the context pool configuration

After you configure your LDAP user registry, you can adjust the number of context instances that the context pool concurrently maintains to improve performance.

Before you begin

In a stand-alone server environment, you can complete the following task when the servers are either stopped or started. In a clustered environment, start the deployment manager and node agent. Then, verify that they are able to synchronize.

Procedure

1. Go to the *wp_profile_root*/ConfigEngine/properties directory.
2. Open the wkplc.properties file with a text editor.
3. Update the following parameters under the VMM LDAP context pool heading:

   Note: Go to the properties file for specific information about the parameters.

   cp.ldap.id

   cp.maxPoolSize

   cp.initPoolSize

   cp.prefPoolSize

   cp.poolTimeout

   cp.poolWaitTime
4. Save your changes to the wkplc.properties file.
5. Open a command prompt.
6. Change to the *wp_profile_root*/ConfigEngine directory.
7. Run the following task to update the context pool configuration for the LDAP user registry:
   - Linux : `./ConfigEngine.sh wp-update-ldap-contextpool -DWasPassword=`*password*
   - IBM i: `ConfigEngine.sh wp-update-ldap-contextpool -DWasPassword=`*password*
   - Windows: `ConfigEngine.bat wp-update-ldap-contextpool -DWasPassword=`*password*
8. Stop and restart the appropriate servers to propagate the changes. For instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.

Creating the entity type

If an entity type exists within IBM WebSphere Portal Express but not within your LDAP user registry, create the entity type within your LDAP user registry. Then, add the relative distinguished name (RDN) to the entity type to map it between WebSphere Portal Express and your LDAP user registry.

Before you begin

In a stand-alone server environment, you can complete the following task when the servers are either stopped or started. In a clustered environment, start the deployment manager and node agent. Then, verify that they are able to synchronize.

Procedure

1. Open a command prompt.
2. Change to the *wp_profile_root*/ConfigEngine directory.
3. Optional: Run the following task to list the names and types of configured repositories:
   - Linux : ./ConfigEngine.sh wp-query-repository -DWasPassword=*password*
   - IBM i: ConfigEngine.sh wp-query-repository -DWasPassword=*password*
   - Windows: ConfigEngine.bat wp-query-repository -DWasPassword=*password*
4. Go to the *wp_profile_root*/ConfigEngine/properties directory.
5. Open the wkplc.properties file with a text editor.
6. Enter the following parameters under the VMM LDAP entity type configuration heading:

   Note: Go to the properties file for specific information about the parameters.

   et.ldap.id

   et.entityTypeName

   et.objectClass

   et.searchFilter

   et.objectClassesForCreate

   et.searchBases
7. Save your changes to the wkplc.properties file.
8. Run the following task to update a realm configuration:
   - Linux : ./ConfigEngine.sh wp-update-realm -DWasPassword=*password*
   - IBM i: ConfigEngine.sh wp-update-realm -DWasPassword=*password*
   - Windows: ConfigEngine.bat wp-update-realm -DWasPassword=*password*
9. Stop and restart the appropriate servers to propagate the changes. For instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.
10. Open the wkplc.properties file.
11. Enter a value for the following parameters under the VMM LDAP entity type configuration heading:

    et.ldap.id

    et.entityTypeName

    et.objectClass

    et.searchFilter

    et.objectClassesForCreate

    et.searchBases

    et.rdnName
12. Save your changes to the wkplc.properties file.
13. Run the following task to add an LDAP entity type with a relative distinguished name (DN):
    - Linux : ./ConfigEngine.sh wp-add-ldap-entitytype-rdn -DWasPassword=*password*
    - IBM i: ConfigEngine.sh wp-add-ldap-entitytype-rdn -DWasPassword=*password*
    - Windows: ConfigEngine.bat wp-add-ldap-entitytype-rdn -DWasPassword=*password*

14. Stop and restart the appropriate servers to propagate the changes.

Changing from a stand-alone repository to a federated repository

If you originally configured a stand-alone LDAP user registry but require a robust security configuration, you can change to the federated user repository.

Before you begin

In a stand-alone server environment, you can complete the following task when the servers are either stopped or started. In a clustered environment, start the deployment manager and node agent. Then, verify that they are able to synchronize.

About this task

Remember: Starting with IBM WebSphere Portal Express Version 8.5, the stand-alone LDAP repository is deprecated. Change to the federated LDAP user repository.

Use the `wp_security_federated.properties` helper file that is in the *wp_profile_root*`/ConfigEngine/config/helpers` directory. It ensures that the correct properties are entered. In the following instructions, where the step refers to the `wkplc.properties` file, use your `wp_security_federated.properties` helper file.

Procedure

1. Go to the *wp_profile_root*`/ConfigEngine/properties` directory.
2. Open the `wkplc.properties` file with a text editor.
3. Update the following parameters in the `wkplc.properties` file under the VMM Federated repository properties heading:

   Note: Go to the properties file for specific information about the parameters.

   `federated.primaryAdminId`

   `federated.realm`

   `federated.serverId`

   `federated.serverPassword`
4. Open a command prompt.
5. Change to the *wp_profile_root*`/ConfigEngine` directory.
6. Run the following task to change the configuration to use a federated repository:
   - Linux : `./ConfigEngine.sh wp-modify-federated-security -DWasPassword=`*`password`*` -Dskip.ldap.validation=true`
   - IBM i: `ConfigEngine.sh wp-modify-federated-security -DWasPassword=`*`password`*` -Dskip.ldap.validation=true`
   - Windows: `ConfigEngine.bat wp-modify-federated-security -DWasPassword=`*`password`*` -Dskip.ldap.validation=true`
7. Stop and restart the appropriate servers to propagate the changes. For instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.
8. Log in to WebSphere Portal Express as an administrator.
   a. Click **Administration**. Then, click **Virtual Portals** > **Manage Virtual Portals**.
   b. Edit each Virtual Portal using the pencil icon.
   c. Set **User realm** as blank.

d. Click **OK**.
e. Edit each Virtual Portal using the pencil icon.
f. Set **User realm** to match the realm ID that you set for **federated.realm**.
g. Click **OK**.

Restoring the VMM setup with a federated file repository

If your business changes or your user registry configuration is inoperable, run the **wp-restore-default-repository-configuration** task to restore the default VMM setup with a federated file repository. Then, reconfigure your user registry. The task deletes all existing repositories, creates a realm, and configures a file repository in VMM. The task also creates a user and group, which is set to the IBM WebSphere Portal Express administrator.

Procedure

1. Before you configure security, use the IBM WebSphere Application Server **backupConfig** task to create and store a backup of the WebSphere Portal Express configuration. Read backupConfig command for information.
2. Go to the *wp_profile_root*/ConfigEngine/properties directory.
3. Open the wkplc.properties file with a text editor.
4. Required: Enter a value for the following parameters under the VMM Federated repository properties heading:

   Note: Go to the properties file for specific information about the parameters.

   federated.primaryAdminId

   federated.realm

   federated.serverId

   federated.serverPassword
5. Open a command prompt.
6. Change to the *wp_profile_root*/ConfigEngine directory.
7. Run the following task to restore the default repository:
   - Linux : ./ConfigEngine.sh wp-restore-default-repository-configuration -DWasPassword=*password*
   - IBM i: ConfigEngine.sh wp-restore-default-repository-configuration -DWasPassword=*password*
   - Windows: ConfigEngine.bat wp-restore-default-repository-configuration -DWasPassword=*password*
8. Stop and restart the appropriate servers to propagate the changes. For instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.
9. Run the following task to restore the default repository group member:
   - Linux : ./ConfigEngine.sh wp-restore-default-repository-add-group-member -DgroupUniqueName=*value* -DmemberUniqueName=*value* -DWasPassword=*password*
   - IBM i: ConfigEngine.sh wp-restore-default-repository-add-group-member -DgroupUniqueName=*value* -DmemberUniqueName=*value* -DWasPassword=*password*
   - Windows: ConfigEngine.bat wp-restore-default-repository-add-group-member -DgroupUniqueName=*value* -DmemberUniqueName=*value* -DWasPassword=*password*
10. Stop and restart the appropriate servers to propagate the changes.

Updating an entity type

After you add your user registry, you can update a single entity type with the value of the default parent. For example, if you delete a repository, you must update the entity type if it points to the deleted repository.

Before you begin

In a stand-alone server environment, you can complete the following task when the servers are either stopped or started. In a clustered environment, start the deployment manager and node agent. Then, verify that they are able to synchronize.

Procedure

1. Go to the *wp_profile_root*/ConfigEngine/properties directory.
2. Open the wkplc.properties file with a text editor.
3. Enter a value for the following parameters under the VMM supported entity types configuration heading:

   Note: Go to the properties file for specific information about the parameters.

   entityTypeName

   defaultParent

   rdnProperties
4. Save your changes to the wkplc.properties file.
5. Open a command prompt.
6. Change to the *wp_profile_root*/ConfigEngine directory.
7. Run the following task to delete the old entity types before you add the RDN attribute as the only entry in the RDN list:
   - Linux : ./ConfigEngine.sh wp-set-entitytype -DWasPassword=*password*
   - IBM i: ConfigEngine.sh wp-set-entitytype -DWasPassword=*password*
   - Windows: ConfigEngine.bat wp-set-entitytype -DWasPassword=*password*
8. Run the following task to add an entity type to the existing list:
   - Linux : ./ConfigEngine.sh wp-update-entitytype -DWasPassword=*password*
   - IBM i: ConfigEngine.sh wp-update-entitytype -DWasPassword=*password*
   - Windows: ConfigEngine.bat wp-update-entitytype -DWasPassword=*password*
9. Stop and restart the appropriate servers to propagate the changes. For instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.

Updating the base entry

After you create your base entries, you can update the distinguished name (DN) in the repository that uniquely identifies the base entry name. This task applies only to federated repository configurations. This task does not update the base DN entry if you use a stand-alone repository.

Before you begin

In a stand-alone server environment, you can complete the following task when the servers are either stopped or started. In a clustered environment, start the deployment manager and node agent. Then, verify that they are able to synchronize.

Procedure

1. Go to the *wp_profile_root*/ConfigEngine/properties directory.
2. Open the wkplc.properties file with a text editor.
3. Enter a value for the following parameters under the VMM repository base entry configuration heading:

   Note: Go to the properties file for specific information about the parameters.

   id

   baseDN

   nameInRepository
4. Save your changes to the wkplc.properties file.
5. Open a command prompt.
6. Change to the *wp_profile_root*/ConfigEngine directory.
7. Run the following task to delete the old entity types before you add the RDN attribute as the only entry in the RDN list:
   - Linux : ./ConfigEngine.sh wp-update-base-entry -DWasPassword=*password*
   - IBM i: ConfigEngine.sh wp-update-base-entry -DWasPassword=*password*
   - Windows: ConfigEngine.bat wp-update-base-entry -DWasPassword=*password*
8. Stop and restart the appropriate servers to propagate the changes. For instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.

Updating the database user registry

After you create and use the database user registry, you can update the database user ID, password, and where the data is stored. This task does not change the DN structure that is stored in the database repository.

Before you begin

If you plan to change the database where data is stored, populate the new database with all necessary VMM tables and create the data sources in WebSphere Application Server. Then, run this task. Read the following information for information about setting up a VMM database. After you populate the new database, restart the WebSphere_Portal server.

- Linux Windows: Setting up an entry mapping repository, a property extension repository, or a custom registry database repository using wsadmin commands
- IBM i: Setting up an entry mapping repository, a property extension repository, or a custom registry database repository using wsadmin commands

If you change the database administrator password, complete the steps in "Changing database passwords that are used by WebSphere Portal Express" on page 1677 before you run this task.

About this task

Note: Use the wp_add_DB.properties helper file in the *wp_profile_root*/ConfigEngine/config/helpers directory to ensure that the correct properties are entered. In the following instructions, where the step refers to the wkplc.properties file, use your wp_add_DB.properties helper file.

Procedure

1. Prepare your servers.

- In a stand-alone server environment, you can complete the following task when the servers are either stopped or started.
- In a clustered environment, start the deployment manager and node agent. Then, verify that they are able to synchronize.

2. Go to the *wp_profile_root*/ConfigEngine/properties directory.
3. Open the wkplc.properties file with a text editor.
4. Enter the following parameters in the wkplc.properties file under Federated DB repository heading:

   Note: Go to the properties file for specific information about the parameters.

   federated.db.DataSourceName

   federated.db.DbType

   federated.db.DbUrl

   federated.db.id

   federated.db.DbUser

   federated.db.DbPassword
5. Save your changes to the wkplc.properties file.
6. Open a command prompt.
7. Change to the *wp_profile_root*/ConfigEngine directory.
8. Run the following task to delete the required repository:
   - Linux : ./ConfigEngine.sh wp-update-db -DWasPassword=*password*
   - IBM i: ConfigEngine.sh wp-update-db -DWasPassword=*password*
   - Windows: ConfigEngine.bat wp-update-db -DWasPassword=*password*
9. Stop and restart the appropriate servers to propagate the changes. For instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.

Related tasks:

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

Updating a group member

After you create your LDAP user registry, you might find that your group member is not correct. You can update the group member in your LDAP user registry configuration.

Before you begin

In a stand-alone server environment, you can complete the following task when the servers are either stopped or started. In a clustered environment, start the deployment manager and node agent. Then, verify that they are able to synchronize.

Procedure

1. Open a command prompt.
2. Change to the *wp_profile_root*/ConfigEngine directory.
3. Optional: Run the following task to list the names and types of configured repositories:

- Linux : `./ConfigEngine.sh wp-query-repository -DWasPassword=password`
- IBM i: `ConfigEngine.sh wp-query-repository -DWasPassword=password`
- Windows: `ConfigEngine.bat wp-query-repository -DWasPassword=password`

4. Go to the *wp_profile_root*`/ConfigEngine/properties` directory.
5. Open the `wkplc.properties` file with a text editor.
6. Enter a value for the following parameters under the VMM LDAP group member attribute configuration heading:

   Note: Go to the properties file for specific information about the parameters.

   gm.ldap.id

   gm.groupMemberName

   gm.objectClass

   gm.scope

   gm.dummyMember
7. Save your changes to the `wkplc.properties` file.
8. Run the following task to update the group member information for your LDAP user registry or to create the group member if it does not exist:
   - Linux : `./ConfigEngine.sh wp-update-ldap-groupmember -DWasPassword=password`
   - IBM i: `ConfigEngine.sh wp-update-ldap-groupmember -DWasPassword=password`
   - Windows: `ConfigEngine.bat wp-update-ldap-groupmember -DWasPassword=password`
9. Stop and restart the appropriate servers to propagate the changes. For instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.

Updating the federated LDAP user registry

After you create and use the LDAP user registry in the default federated repository, you might find that your LDAP user registry is not working correctly. You can update the LDAP user registry and make the necessary changes. For example, you can change your LDAP Bind password.

Before you begin

In a stand-alone server environment, you can complete the following task when the servers are either stopped or started. In a clustered environment, start the deployment manager and node agent. Then, verify that they are able to synchronize.

About this task

Note: The update federated LDAP user registry task does not modify the following attributes:

Administrative users

Entity types

LDAP entity types

LDAP group membership attributes

LDAP group configuration

LDAP context pool

There are separate tasks to update these attributes.

Note: Use the `wp_security_federated.properties` helper file, that is in the *wp_profile_root*`/ConfigEngine/config/helpers` directory to ensure that the correct properties are entered. In the following instructions, where the step refers to the `wkplc.properties` file, use your `wp_security_federated.properties` helper file.

Procedure

1. Go to the *wp_profile_root*`/ConfigEngine/properties` directory.
2. Open the `wkplc.properties` file with a text editor.
3. Enter the following parameters in the `wkplc.properties` file under Federated LDAP repository heading:

   Note: Go to the properties file for specific information about the parameters.

   federated.ldap.id

   federated.ldap.host

   federated.ldap.baseDN

   federated.ldap.ldapServerType

   federated.ldap.port

   federated.ldap.bindDN

   federated.ldap.bindPassword
4. Save your changes to the `wkplc.properties` file.
5. Open a command prompt.
6. Change to the *wp_profile_root*`/ConfigEngine` directory.
7. Run the following task to validate your LDAP server settings:
   - Linux : `./ConfigEngine.sh validate-federated-ldap -DWasPassword=`*password*
   - IBM i: `ConfigEngine.sh validate-federated-ldap -DWasPassword=`*password*
   - Windows: `ConfigEngine.bat validate-federated-ldap -DWasPassword=`*password*

   Note: In an environment that is configured with an LDAP with SSL, you are prompted to add a signer to the truststore. The prompt is `Add signer to the truststore now?`. If you do, press **y** and then **Enter**.
8. Run the following task to update the LDAP user registry in the default federated repository:
   - Linux : `./ConfigEngine.sh wp-update-federated-ldap -DWasPassword=`*password*
   - IBM i: `ConfigEngine.sh wp-update-federated-ldap -DWasPassword=`*password*
   - Windows: `ConfigEngine.bat wp-update-federated-ldap -DWasPassword=`*password*
9. Stop and restart the appropriate servers to propagate the changes. For instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.

Updating the realm configuration

After you create and use the realm in the default federated repository, you might find that your realm configuration is not working correctly. You can update the realm configurations and make the necessary changes.

Procedure

1. Go to the *wp_profile_root*/ConfigEngine/properties directory.
2. Open the wkplc.properties file with a text editor.
3. Enter a value for the following parameters under the VMM realm configuration heading:

   Note: Go to the properties file for specific information about the parameters.

   realmName

   securityUse

   delimiter
4. Save your changes to the wkplc.properties file.
5. Open a command prompt.
6. Change to the *wp_profile_root*/ConfigEngine directory.
7. Run the following task to delete the old entity types before you add the RDN attribute as the only entry in the RDN list:
   - Linux : ./ConfigEngine.sh wp-update-realm -DWasPassword=*password*
   - IBM i: ConfigEngine.sh wp-update-realm -DWasPassword=*password*
   - Windows: ConfigEngine.bat wp-update-realm -DWasPassword=*password*
8. Stop and restart the appropriate servers to propagate the changes. For instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.

Configuring a property extension database

A property extension database stores attributes that the LDAP directory does not or cannot store, but that you want to include in your portal user registry. This situation often occurs when you are using an LDAP directory that does not allow schema extensions for new attributes to support portal applications. When you configure a property extension database, you effectively extend the user registry to make new attributes available as part of your portal user profile. However, it is preferable to store all user attributes in the main user registry. Complete this task only if you cannot add attributes to your LDAP directory.

Before you begin

Configure portal security with your main user registry before you configure your property extension database. If you complete these steps and then configure your user registry, your property extension database configuration does not work.

The Virtual Member Manager (VMM) has a limitation that includes no tasks to update attributes. To change an attribute, you must first remove the attribute then add it again. For this reason, ensure that you spell all attributes correctly and use caution when you add attributes to your property extension database.

The VMM database schema has a limit of 36 characters on the repository ID column. For this reason, you must use a repository ID that is 36 characters or less.

Procedure

1. This task requires server connections.
   - In a stand-alone environment, ensure the WebSphere_Portal server is running.

- In a clustered environment, stop all application servers on the system. Ensure that the WebSphere_Portal server is stopped. Then, start the node agent and deployment manager servers.

2. Complete the following steps to install the enterprise archive (`.ear`) file on WebSphere Application Server:
   a. Open a command prompt.
   b. Change to the *wp_profile_root*`/ConfigEngine` directory.
   c. Run the following task:

Table 76. Stand-alone and cluster tasks to install the enterprise archive file.. The **wp-la-install-ear** command.

Environment	Task
Stand-alone environment	• Linux : `./ConfigEngine.sh wp-la-install-ear -DWasPassword=`*`password`* • IBM i: `ConfigEngine.sh wp-la-install-ear -DWasPassword=`*`password`* • Windows: `ConfigEngine.bat wp-la-install-ear -DWasPassword=`*`password`*
Clustered environment	• Linux : `./ConfigEngine.sh wp-la-install-ear -DWasPassword=`*`dmgr_password`*` -DServerName=`*`dmgr_server_name`*` -DNodeName=`*`node_name`* • IBM i: `ConfigEngine.sh wp-la-install-ear -DWasPassword=`*`dmgr_password`*` -DServerName=`*`dmgr_server_name`*` -DNodeName=`*`node_name`* • Windows: `ConfigEngine.bat wp-la-install-ear -DWasPassword=`*`dmgr_password`*` -DServerName=`*`dmgr_server_name`*` -DNodeName=`*`node_name`* Where the default value for *`dmgr_server_name`* is `dmgr`. You can find the *`dmgr_server_name`* value in the WebSphere Integrated Solutions Console. Go to **System administrator** > **Deployment Manager** > **Configuration tab** > **General Properties** > **Name**. Where *`node_name`* is the name of the node where the deployment manager is located. You can find the *`node_name`* value in the WebSphere Integrated Solutions Console. Go to **System administrator** > **Deployment Manager** > **Runtime tab** > **General Properties** > **Node Name**.

3. Stop and restart the appropriate servers to propagate the changes. For specific instructions, read "Starting and stopping servers, deployment managers, and node agents" on page 1216.

4. Set up a new database. Create a user with the appropriate database privileges for accessing the database. The database user needs to have permissions to create tables in the database schema and permission to access the data in the database tables.

   Instructions for setting up databases: Refer to the appropriate documentation for the type of database you want to set up.

   Consulting your database administrator: A database administrator typically completes the task of setting up a new database. However, the following steps are provided for your reference in the event you create a stand-alone database for testing or demonstration purposes. Consult your database administrator if you plan to create a database for a production environment.

Table 77. Steps for creating a database to use as a database user registry.. This table describes the steps by database type to create a database for your database user registry.

Database	Steps
DB2	Complete the following steps to create a DB2 database: 1. Install DB2. 2. Enter the following commands to tune your database: `db2 "CREATE DB dbname using codeset UTF-8 territory us PAGESIZE 8192"` `db2 "UPDATE DB CFG FOR dbname USING applheapsz 4096"` `db2 "UPDATE DB CFG FOR dbname USING app_ctl_heap_sz 1024"` `db2 "UPDATE DB CFG FOR dbname USING stmtheap 32768"` `db2 "UPDATE DB CFG FOR dbname USING dbheap 2400"` `db2 "UPDATE DB CFG FOR dbname USING locklist 1000"` `db2 "UPDATE DB CFG FOR dbname USING logfilsiz 4000"` `db2 "UPDATE DB CFG FOR dbname USING logprimary 12"` `db2 "UPDATE DB CFG FOR dbname USING logsecond 20"` `db2 "UPDATE DB CFG FOR dbname USING logbufsz 32"` `db2 "UPDATE DB CFG FOR dbname USING avg_appls 5"` `db2 "UPDATE DB CFG FOR dbname USING locktimeout 30"` `db2 "UPDATE DB CFG FOR dbname using AUTO_MAINT off"`
Oracle	Complete the following steps to create an Oracle database: 1. Install Oracle with Unicode database and National character sets such as UTF8, AL32UTF8, or AL16UTF16. 2. Configure the database in Dedicated Server Mode. 3. Enter the initial buffer pool sizes or set them according to your business needs: **db_block_size** = 8192 **db_cache_size** = 300M **db_files** = 1024 **log_buffer** = 65536 **open_cursors** = 1500**open_cursors** = 1500 **pga_aggregate_target** = 200M **pre_page_sga** = true **processes** = 300 **shared_pool_size** = 200M
SQL Server	Complete the following steps to create an SQL Server database: 1. Create an SQL Server database with a name of your choice. 2. Create a SQL Server database user with the same permissions as your Portal database users for this new database. **Tip:** Do not use `userid=sa`. This ID is a special user ID that has restrictions. Instead, create a new user. Then, create a new schema with the user as owner and assign the user permissions to create database tables. **Note:** Install SQL Server with the appropriate portal database collation so that your **tempdb collation** setting matches the collation that you use for the property extension database. The **tempdb collation** is inherited from the master database, which you set when you install SQL Server.

5. Complete the following steps to create the IBM DB2 for i database:

Instructions for setting up databases: Refer to the appropriate documentation for the type of database you want to set up.

Consulting your database administrator: A database administrator typically completes the task of setting up a new database. However, the following steps are provided for your reference in the event you create a stand-alone database for testing or demonstration purposes. Consult your database administrator before you proceed with the following steps if you plan to create a database for a production environment.

a. Log in to a remote IBM i session.
b. Enter the `strsql` command to start the interactive sql session.
c. Enter the `create schema` *`database_name`* command, where *`database_name`* is the name you want to use for the database.

6. Complete the following steps to define the **`DbDriver`** and **`DbLibrary`** parameter values:
   a. Go to the *wp_profile_root*`/ConfigEngine/properties` directory.
   b. Open the `wkplc_dbtype.properties` file with a text editor.
   c. Enter a value for the following parameters in the appropriate database type properties heading:

      `db_type.DbDriver`

      `db_type.DbLibrary`

   d. Save your changes.
7. Specify values for the data source parameters in the `wp_add_LA.properties` file:
   a. Go to the *wp_profile_root*`/ConfigEngine/config/helpers` directory.
   b. Open the `wp_add_LA.properties` file with a text editor.
   c. Specify values for the following parameters:

      Note: Go to the properties file for specific information about the parameters.

      `la.JdbcProviderName`

      `la.DbType`

      `la.DbUrl`

      `la.DbName`

      `la.DataSourceName`

      `la.DbUser`

      `la.DbPassword`

      Adding parameters for a clustered environment: Add the following parameters to the `wp_add_LA.properties` if you are setting up the property extension database on a clustered environment:

      `la.schemaLocation=`*`WAS_install_location_on_DMGR`*`/AppServer/etc/wim/setup`

      `la.laPropXML=`*`WAS_install_location_on_DMGR`*`/AppServer/etc/wim/setup/wimlaproperties.xml`

      Where *`WAS_install_location_on_DMGR`* is the local path on your deployment manager node.

   d. Save your changes.
8. Change the value for the **`com.ibm.SOAP.requestTimeout`** parameter:

a. Go to the *wp_profile_root*\properties directory.
b. Open the soap.client.props with a text editor.
c. Locate the **com.ibm.SOAP.requestTimeout** parameter.
d. Ensure that the value is greater than 1000.
e. Save your changes.

9. Complete the following step in a clustered environment:
   a. Open the wkplc.properties file.
   b. Set the property value for **federated.db.DbType** if you are using a database user registry or if the cell is migrated from a previous version.
   c. Save your changes.
   d. Open a command prompt.
   e. Change to the *wp_profile_root*/ConfigEngine directory.
   f. Run the following task to create the local Deployment Manager WebSphere variable that is used to access the database JAR files:
      - Linux : `./ConfigEngine.sh wp-prep-vmm-db-secured-environment -DWasPassword=`*`password`* `-DDbDomain=la -D`*`db_type`*`.DmgrDbLibrary=`*`local path of the database jars on the Deployment Manager`* `-DDmgrNodeName=`*`dmgr_node_name`*
      - IBM i: `ConfigEngine.sh wp-prep-vmm-db-secured-environment -DWasPassword=`*`password`* `-DDbDomain=la -D`*`db_type`*`.DmgrDbLibrary=`*`local path of the database jars on the Deployment Manager`* `-DDmgrNodeName=`*`dmgr_node_name`*
      - Windows: `ConfigEngine.bat wp-prep-vmm-db-secured-environment -DWasPassword=`*`password`* `-DDbDomain=la -D`*`db_type`*`.DmgrDbLibrary=`*`local path of the database jars on the Deployment Manager`* `-DDmgrNodeName=`*`dmgr_node_name`*

      Note: Set the *db_type* in *db_type***.DmgrDbLibrary** to the type of database you are using, for example db2. The *local full path of the database jars on the Deployment Manager* is one of the following options:

      DB2 Type 2 driver: db2java.zip

      DB2 Type 4 driver: db2jcc4.jar:db2jcc_license_cu.jar

      DB2 for z/OS Type 2 driver: db2java.zip

      DB2 for z/OS Type 4 driver: db2jcc4.jar:db2jcc_license_cisuz.jar

      Oracle: ojdbc14.jar

      SQL Server JDBC driver that is provided by Microsoft: sqljdbc4.jar
   g. Run the following task on each node to create the variable that is used to access the VMM database JAR files. Include each node name as a comma-separated list in the command:
      - Linux : `./ConfigEngine.sh wp-node-prep-vmm-db-secured-environment -DWasPassword=`*`password`* `-DDbDomain=federated.db -DVmmNodeName=`*`node_name`*`,`*`node_name`*`,`*`node_name`* `-D`*`db_type`*`.NodeDbLibrary=`*`local full path of the database jars`*
      - IBM i: `ConfigEngine.sh wp-node-prep-vmm-db-secured-environment -DWasPassword=`*`password`* `-DDbDomain=federated.db -DVmmNodeName=`*`node_name`*`,`*`node_name`*`,`*`node_name`* `-D`*`db_type`*`.NodeDbLibrary=`*`local full path of the database jars`*
      - Windows: `ConfigEngine.bat wp-node-prep-vmm-db-secured-environment -DWasPassword=`*`password`* `-DDbDomain=federated.db`

```
-DVmmNodeName=node_name,node_name,node_name
-Ddb_type.NodeDbLibrary=local full path of the database jars
```

Note: VmmNodeName is a list of one or more WebSphere Portal Express nodes names in the cell which share database driver paths. Set the *db_type* in *db_type***.NodeDbLibrary** to the type of database you are using, for example db2.

h. Stop and restart all necessary servers to propagate your changes.

10. Run the following task to add a property extension repository to the federated LDAP repository:
    - Linux : `./ConfigEngine.sh wp-configure-la-complete -DWasPassword=password -DparentProperties=full path to wp_profile_root/ConfigEngine/config/helpers/wp_add_LA.properties`
    - IBM i: `ConfigEngine.sh wp-configure-la-complete -DWasPassword=password -DparentProperties=full path to wp_profile_root/ConfigEngine/config/helpers/wp_add_LA.properties`
    - Windows: `ConfigEngine.bat wp-configure-la-complete -DWasPassword=password -DparentProperties=full path to wp_profile_root/ConfigEngine/config/helpers/wp_add_LA.properties`
11. Stop and restart the appropriate servers to propagate the changes.
12. Specify values for the following parameters in `wp_add_LA.properties` file:

    Note: Go to the properties file for specific information about the parameters.

    la.providerURL

    la.propertyName

    la.entityTypes

    la.dataType

    la.multiValued

    Values for the dataType parameter: Available data types that are defined in `com.ibm.websphere.wim.SchemaConstants`:

    - `String`
    - `Int`
    - `Date`
    - `AnySimpleType`
    - `AnyURI`
    - `Boolean`
    - `Long`
    - `Double`
    - `Short`

    Note: A complete overview of valid **dataType** values can be found in the Configuring a property extension repository in a federated repository configuration file. All constant values of DATA_TYPE_* fields are valid input for **la.dataType**. Only the `String` data type is valid for displaying attributes in the Profile Management portlet. These attributes can be added to the Profile Management portlet through the configuration mode interface.

13. Complete the following steps to add the attribute to the property extension database:

    a. Run the following task:

Note: If the path name contains blank space, you must enclose the path in quotation marks.

- Linux : `./ConfigEngine.sh wp-add-la-property -DWasPassword=`*`password`* `-DparentProperties=`*`full path to wp_profile_root`*`/ConfigEngine/config/helpers/wp_add_LA.properties`
- IBM i: `ConfigEngine.sh wp-add-la-property -DWasPassword=`*`password`* `-DparentProperties=`*`full path to wp_profile_root`*`/ConfigEngine/config/helpers/wp_add_LA.properties`
- Windows: `ConfigEngine.bat wp-add-la-property -DWasPassword=`*`password`* `-DparentProperties=`*`full path to wp_profile_root`*`\ConfigEngine\config\helpers\wp_add_LA.properties`

b. Run the following task to add the attributes to Web Content Manager if you use user profiling or Category selection trees:

Note: If the path name contains blank space, you must enclose the path in quotation marks.

- Linux : `./ConfigEngine.sh add-wcm-la-attributes -DWasPassword=`*`password`* `-DparentProperties=`*`full path to wp_profile_root`*`/ConfigEngine/config/helpers/wp_add_LA.properties`
- IBM i: `ConfigEngine.sh add-wcm-la-attributes -DWasPassword=`*`password`* `-DparentProperties=`*`full path to wp_profile_root`*`/ConfigEngine/config/helpers/wp_add_LA.properties`
- Windows: `ConfigEngine.bat add-wcm-la-attributes -DWasPassword=`*`password`* `-DparentProperties=`*`full path to wp_profile_root`*`\ConfigEngine\config\helpers\wp_add_LA.properties`

Receiving an authentication prompt: This task makes an EJB call to WebSphere Application Server, which requires authentication. You might receive an authentication prompt. Enter the appropriate WebSphere Application Server user ID and password.

14. Stop and restart the appropriate servers to propagate the changes.

Deleting your user registry configurations

After you deploy IBM WebSphere Portal Express, you might not require some of the LDAP entity types, realms, realm base entries, or repositories that you created. You can delete these configurations to achieve the correct user registry configuration.

About this task

Choose from the following tasks to delete your user registry configurations:

"Deleting the LDAP entity type" on page 603
If you changed your LDAP user registry and no longer require an entity type, you can delete it.

"Deleting the base entry" on page 603
If you changed your LDAP user registry and no longer require a base entry, you can delete it.

"Deleting the LDAP group member" on page 605
If you changed your LDAP user registry and no longer require the group member, you can delete it from the LDAP user registry.

"Deleting the realm" on page 605
If you changed your IBM WebSphere Portal Express and no longer require a realm, you can delete it from your user registry.

"Deleting the realm base entry" on page 606
If you changed your realm and no longer require the base entry for that realm, you can delete it.

"Deleting the repository" on page 607
If you no longer require the use of a repository within your federated repository, you can delete it from your configuration.

Deleting the LDAP entity type

If you changed your LDAP user registry and no longer require an entity type, you can delete it.

Before you begin

In a stand-alone server environment, you can complete the following task when the servers are either stopped or started. In a clustered environment, start the deployment manager and node agent. Then, verify that they are able to synchronize.

Procedure

1. Go to the *wp_profile_root*/ConfigEngine/properties directory.
2. Open the wkplc.properties file with a text editor.
3. Enter the following parameters under the VMM LDAP entity type configuration heading:

   Note: Go to the properties file for specific information about the parameters.

   et.ldap.id

   et.entityTypeName

   et.objectClass

   et.searchFilter

   et.objectClassesForCreate

   et.searchBases
4. Save your changes to the wkplc.properties file.
5. Open a command prompt.
6. Change to the *wp_profile_root*/ConfigEngine directory.
7. Optional: Run the following task to delete an LDAP entity type:
   - Linux : ./ConfigEngine.sh wp-delete-ldap-entitytype -DWasPassword=*password*
   - IBM i: ConfigEngine.sh wp-delete-ldap-entitytype -DWasPassword=*password*
   - Windows: ConfigEngine.bat wp-delete-ldap-entitytype -DWasPassword=*password*
8. Stop and restart the appropriate servers to propagate the changes. For instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.

Deleting the base entry

If you changed your LDAP user registry and no longer require a base entry, you can delete it.

Before you begin

In a stand-alone server environment, you can complete the following task when the servers are either stopped or started. In a clustered environment, start the deployment manager and node agent. Then, verify that they are able to synchronize.

About this task

Important: If you delete a base entry from the default realm, then users who belong to that base entry can no longer log in to IBM WebSphere Portal Express. Therefore, the default realm must contain all the base entries for all active realms.

Procedure

1. Open a command prompt.
2. Change to the *wp_profile_root*/ConfigEngine directory.
3. Optional: Run the following task to list the base entries for a specific realm:
   - Linux : ./ConfigEngine.sh wp-query-realm-baseentry -DrealmName=*name_of_realm* -DWasPassword=*password*
   - IBM i: ConfigEngine.sh wp-query-realm-baseentry -DrealmName=*name_of_realm* -DWasPassword=*password*
   - Windows: ConfigEngine.bat wp-query-realm-baseentry -DrealmName=*name_of_realm* -DWasPassword=*password*
4. Go to the *wp_profile_root*/ConfigEngine/properties directory.
5. Open the wkplc.properties file with a text editor.
6. Enter a value for the following parameters under the VMM realm configuration heading:

   Note: Go to the properties file for specific information about the parameters.

   realmName

   addBaseEntry
7. Save your changes to the wkplc.properties file.
8. Optional: Run the following task to delete a base entry from a realm configuration:
   - Linux : ./ConfigEngine.sh wp-delete-realm-baseentry -DWasPassword=*password*
   - IBM i: ConfigEngine.sh wp-delete-realm-baseentry -DWasPassword=*password*
   - Windows: ConfigEngine.bat wp-delete-realm-baseentry -DWasPassword=*password*
9. Stop and restart the appropriate servers to propagate the changes. For instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.
10. Open the wkplc.properties file.
11. Enter a value for the following parameters under the VMM repository base entry configuration heading:

    id

    baseDN

    nameInRepository
12. Save your changes.

13. Optional: Run the following task to delete a base entry from a repository:
    - Linux : `./ConfigEngine.sh wp-delete-base-entry -DWasPassword=`*`password`*
    - IBM i: `ConfigEngine.sh wp-delete-base-entry -DWasPassword=`*`password`*
    - Windows: `ConfigEngine.bat wp-delete-base-entry -DWasPassword=`*`password`*
14. Stop and restart the appropriate servers to propagate the changes.

Deleting the LDAP group member

If you changed your LDAP user registry and no longer require the group member, you can delete it from the LDAP user registry.

Before you begin

In a stand-alone server environment, you can complete the following task when the servers are either stopped or started. In a clustered environment, start the deployment manager and node agent. Then, verify that they are able to synchronize.

Procedure

1. Go to the *wp_profile_root*`/ConfigEngine/properties` directory.
2. Open the `wkplc.properties` file with a text editor.
3. Enter a value for the following parameters under the VMM LDAP group member attribute configuration heading:

   Note: Go to the properties file for specific information about the parameters.

   `gm.ldap.id`

   `gm.groupMemberName`
4. Save your changes to the `wkplc.properties` file.
5. Open a command prompt.
6. Change to the *wp_profile_root*`/ConfigEngine` directory.
7. Run the following task to delete the group member information for your LDAP user registry:
   - Linux : `./ConfigEngine.sh wp-delete-ldap-groupmember -DWasPassword=`*`password`*
   - IBM i: `ConfigEngine.sh wp-delete-ldap-groupmember -DWasPassword=`*`password`*
   - Windows: `ConfigEngine.bat wp-delete-ldap-groupmember -DWasPassword=`*`password`*
8. Stop and restart the appropriate servers to propagate the changes. For instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.

Deleting the realm

If you changed your IBM WebSphere Portal Express and no longer require a realm, you can delete it from your user registry.

Before you begin

In a stand-alone server environment, you can complete the following task when the servers are either stopped or started. In a clustered environment, start the deployment manager and node agent. Then, verify that they are able to synchronize.

Procedure

1. Open a command prompt.
2. Change to the *wp_profile_root*/ConfigEngine directory.
3. Optional: Run the following task to list the names and types of configured repositories:
   - Linux : ./ConfigEngine.sh wp-delete-realm -DdeleteRealmName=*name_of_realm* -DWasPassword=*password*
   - IBM i: ConfigEngine.sh wp-delete-realm -DdeleteRealmName=*name_of_realm* -DWasPassword=*password*
   - Windows: ConfigEngine.bat wp-delete-realm -DdeleteRealmName=*name_of_realm* -DWasPassword=*password*
4. Stop and restart the appropriate servers to propagate the changes. For instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.

Deleting the realm base entry

If you changed your realm and no longer require the base entry for that realm, you can delete it.

Before you begin

In a stand-alone server environment, you can complete the following task when the servers are either stopped or started. In a clustered environment, start the deployment manager and node agent. Then, verify that they are able to synchronize.

About this task

Important: If you delete a base entry from the default realm, then users who belong to that base entry can no longer log in to IBM WebSphere Portal Express. Therefore, the default realm must contain all the base entries for all active realms.

Procedure

1. Open a command prompt.
2. Change to the *wp_profile_root*/ConfigEngine directory.
3. Optional: Run the following task to list the base entries for a specific realm:
   - Linux : ./ConfigEngine.sh wp-query-realm-baseentry -DrealmName=*name_of_realm* -DWasPassword=*password*
   - IBM i: ConfigEngine.sh wp-query-realm-baseentry -DrealmName=*name_of_realm* -DWasPassword=*password*
   - Windows: ConfigEngine.bat wp-query-realm-baseentry -DrealmName=*name_of_realm* -DWasPassword=*password*
4. Go to the *wp_profile_root*/ConfigEngine/properties directory.
5. Open the wkplc.properties file with a text editor.
6. Enter a value for the following parameters under the VMM realm configuration heading:

   Note: Go to the properties file for specific information about the parameters.

   realmName

   addBaseEntry
7. Save your changes to the wkplc.properties file.

8. Optional: Run the following task to delete a base entry from a realm configuration:
   - Linux : `./ConfigEngine.sh wp-delete-realm-baseentry -DWasPassword=`*`password`*
   - IBM i: `ConfigEngine.sh wp-delete-realm-baseentry -DWasPassword=`*`password`*
   - Windows: `ConfigEngine.bat wp-delete-realm-baseentry -DWasPassword=`*`password`*
9. Stop and restart the appropriate servers to propagate the changes. For instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.

Deleting the repository

If you no longer require the use of a repository within your federated repository, you can delete it from your configuration.

Before you begin

In a stand-alone server environment, you can complete the following task when the servers are either stopped or started. In a clustered environment, start the deployment manager and node agent. Then, verify that they are able to synchronize.

Procedure

1. Open a command prompt.
2. Change to the *wp_profile_root*`/ConfigEngine` directory.
3. Optional: Run the following task to list the names and types of configured repositories:
   - Linux : `./ConfigEngine.sh wp-query-repository -DWasPassword=`*`password`*
   - IBM i: `ConfigEngine.sh wp-query-repository -DWasPassword=`*`password`*
   - Windows: `ConfigEngine.bat wp-query-repository -DWasPassword=`*`password`*
4. Go to the *wp_profile_root*`/ConfigEngine/properties` directory.
5. Open the `wkplc.properties` file with a text editor.
6. Enter the following parameters in the `wkplc.properties` file under VMM Delete federated repository heading:

   Note: Go to the properties file for specific information about the parameters.

   `federated.delete.baseentry`

   `federated.delete.id`
7. Save your changes to the `wkplc.properties` file.
8. Run the following task to delete the required repository:
   - Linux : `./ConfigEngine.sh wp-delete-repository -DWasPassword=`*`password`*
   - IBM i: `ConfigEngine.sh wp-delete-repository -DWasPassword=`*`password`*
   - Windows: `ConfigEngine.bat wp-delete-repository -DWasPassword=`*`password`*
9. Stop and restart the appropriate servers to propagate the changes. For instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.
10. If the deleted repository contains the base entry where new users and groups are stored, run the update entity types task. Use this task to select a new base entry to store the new users and groups.

Search

You use Portal Search to search for text that is displayed in websites that are created by IBM Web Content Manager.

1. "Searching pages" on page 609
   By selecting a search type in Manage Pages, you can quickly locate and work with pages, labels, or URLs.
2. "Searching for tagged content" on page 610
   Users can search the tag space by using the browser search box.
3. "Planning and preparing for Portal Search" on page 611
   Learn about some planning considerations and first steps that you need to apply before working with Portal Search.
4. "Indexing web content" on page 616
   To search for web content, your content must first be indexed by the WebSphere Portal search engine. When the content is indexed, you can run searches by using the search center or by using a search component. If you search for documents in the WebSphere Portal search center, be aware that you see search results for published documents only. Unpublished pending changes in a project are not included in the results.
5. "Configuring Web Content Manager search options" on page 618
   You can edit the following search options to manage how the search service works with Web Content Manager search forms
6. Optional: "Configuring Search Center to search for web content" on page 620
   You can use the Search Center to search for web content by adding a web content search collection to the Search Center.
7. "Language and region support in Portal Search" on page 620
   Depending on your portal environment and your users, you might want to make multilingual content available and searchable in your portal.
8. "Crawling web content with search seedlists" on page 622
   Portal Search supports the use of seedlists to make crawling websites and their metadata more efficient and to provide content owners fine-grained control over how content and metadata are crawled. You can configure the portal to use seedlist support when crawling content generated with IBM Web Content Manager.
9. "Searching your local portal " on page 627
   View information on setting up your local portal for your users to search.
10. "Configuring your portal site for search by internet search engines" on page 636
    You can enable your portal site for search by using search services such as Google or Yahoo! Search on your portal site by external search services works for public portal pages only, that is, for pages that users can access without a user ID and password.
11. "Enabling anonymous users to search public pages of your portal" on page 641
    You can enable anonymous users (sometimes also called unauthenticated users) to search public pages of your portal by using the portal Search Center portlet. Search by anonymous users works only on public pages of your portal, as the users are not logged in to your portal.
12. "Configuring your custom portal themes to include the search box" on page 644
    Enable your portal users to use the Portal Search box and Search portlet in your own custom theme.

13. "Redirecting search requests from a custom search form to the Search Center" on page 645
    If you plan to develop a custom search form, you might want to redirect search requests issued by the search form to the Search Center.
14. "Remote search service" on page 645
    You can configure the search portlets for local operation, or you can configure them for remote search service. Depending on your configuration, remote search service might have performance benefits by offloading and balancing system load.
15. "Configuring search in a cluster" on page 662
    IBM WebSphere Portal Express provides two distinct search capabilities. You can use both types of search capabilities in a clustered environment.
16. "Configuring search in a portal farm" on page 664
    IBM WebSphere Portal Express provides two distinct search capabilities. You can use both types of search capabilities in a portal farm environment.
17. "Configuring search collections for a virtual portal" on page 666
    Configuring JCR search collections for a virtual portal might require additional administration, depending on how you set up the virtual portal.
18. "" on page 667
    Use Portal Search to facilitate indexing content sources and searching for information. You can administer search services, search collections, and search scopes, as well as enhance the search experience of your portal site with the portal search portlets.

Searching pages

By selecting a search type in Manage Pages, you can quickly locate and work with pages, labels, or URLs.

About this task

Follow these instructions to search for an item.

Procedure

1. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
2. Select the search type from the **Search by** drop-down menu.
3. Enter the search parameters in the **Search** field.
4. Click **Search**.

   Note: Searches for users or user groups can be case-sensitive, depending on the attributes that are used.

   Search Types:

   The following is a list of some search types that can be selected from the Search by: drop-down menu. The search types available depend on the resource type used.

   - **Title starts with**: Select this option to search on the beginning of a string in the title. This option is the default setting, and the input is expected in string format.
   - **Title contains**: Select this option to search on a string in the title. The input is expected in string format.
   - **Name starts with**: Select this option to search on the beginning of a string in the name. The input is expected in string format.

- **Name contains**: Select this option to search on a string in the name. The input is expected in string format.
- **Keyword starts with**: Select this option to search on the beginning of a keyword. The input is expected in string format.
- **Keyword contains**: Select this option to search on a keyword. The input is expected in string format.
- **Description starts with**: Select this option to search on the beginning of a string in the description. The input is expected in string format.
- **Description contains**: Select this option to search on a string in description. The input is expected in string format.
- **Unique Name contains**: Select this option to search on a string in the unique name. The input is expected in string format.
- **Markup supported**: Select this option to search the beginning of a string in the markup type. This option returns a list of pages that support that markup. The input is expected in string format.
- **Label**: Select this option to search on a URL context label.
- **Attributes**: Select this option to search on a user or group attribute.
- **Last modified**: Select this option to search by items that have been modified on or since a specific date. The input is expected in YYYY MM DD format.
- **All available**: Select this option to return a listing of all items. No input is required.

Searching for tagged content

Users can search the tag space by using the browser search box.

This works only for browsers that support OpenSearch. For details refer to http://www.opensearch.org/Home. OpenSearch is a collection of technologies that allow publishing of search results in a format suitable for syndication and aggregation. It is a way for Web sites and search engines to publish search results in a standardized and accessible format.

To be able to search for tagged content from non-portal pages, visit the portal site, navigate to where your browser allows you to add or remove search engines and add the search engine named WebSphere Portal Express Tagging Search Service. For details about how to do this refer to the documentation for your browser. After you added the search service to your browser, you can search the tag space of the Portal via this search service.

The tagging search service searches only the portal site at which the service was added. For example, if you use Firefox to access two different portal servers `server1` and `server2`, and you then add the WebSphere Portal Express Tagging Search Service at server1. If you then navigate to server2 and try to search for a tag, the search service searches server1 for the specified tag, not server2.

Advantages are as follows:

- Users can search for tagged content from pages that do not have a tag cloud, for example from Web pages external to the portal.
- When a user searches for a tag, the portal redirects the user to the All view of the tag cloud. The search results are shown in the tag results list. The user can now work the results.
- The tag result list shows **all** resources that have the searched tag.

- The search has type-ahead support. The following features make tag search easier:
  - As the user types text, one or more possible matches for the text are found and immediately presented to the user. This immediate feedback allows the user to stop short of typing the entire word or phrase they were looking for. The user can also choose a closely related tag from the presented list.
  - It starts after the user has typed the third character of the searched tag.
  - As the user types more characters, the type-ahead support filters and narrows the search progressively.
  - The suggested tags are sorted by their relevance: tags that have been applied more often will be suggested before tags that have not been assigned often.
  - Suggestions and the search results are provided according to the normalization configuration: if no normalization is configured, the suggestions and the search is case sensitive.

By default the searches are case sensitive. If you want to perform a case insensitive search, an administrator needs to enable tag normalization in the CP Configuration Service. For details refer to the topic about the CP Configuration Service for tagging and rating

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Related reference:

"Querying for the OpenSearch description document" on page 3190
You can query for the OpenSearch description document.

"CP Configuration Service for tagging and rating" on page 312
The CP Configuration Service provides the properties for tagging and rating.

"Type-ahead feature for the deprecated tag widget" on page 1306
The tag widget from earlier IBM WebSphere Portal Express versions provided a type-ahead feature. With portal V 8.5, that tag widget is deprecated. The type-ahead feature makes it easier for users to find suitable tags. Type-ahead supports users when they work with tags. For example, when users apply tags using the tag widget, or when they search for tags, for example by using the open search functionality, type-ahead provides users suggestions for tags that other users have applied already before. Type-ahead can also help reduce the number of variants of tags.

Related information:


Open Search - http://www.opensearch.org/Home

Planning and preparing for Portal Search

Learn about some planning considerations and first steps that you need to apply before working with Portal Search.

Make sure you read all of the following sections and also the topic about Hints and tips for using Portal Search.

For an administrative overview of the Portal Search portlets, refer to the topic about Portlets for working with Portal Search.

Portal Search is preconfigured with a search service, a portal site search collection, and scopes. You can add, modify, and remove search services, collections, and scopes.

To enhance your possibilities of working with search, you can install additional portlets related to Portal Search and apply special configurations. For information about how to do this refer to the subtopics of Administering Portal Search.

To prevent errors during crawls and ensure easy viewing of documents for users you need to enable Document Conversion Services. For more information see *Enabling Document Conversion Services*.

"Security considerations"
Learn about the security aspects that you need to consider with regards to content search in IBM WebSphere Portal Express Version 8.5.

Related tasks:

"Using the WebSphere Integrated Solutions Console to administer Portal Search" on page 694
You can administer Portal Search by using the WebSphere Integrated Solutions Console and using resource providers in XML format.

Chapter 6, "Document Conversion Services," on page 727
Document Conversion Services are used when you work with the Common Mail Portlet, IBM Web Content Manager authoring and previewing, and search.

"Administering Portal Search" on page 671
You can administer and configure many details for Portal Search.

Related reference:

"Hints and tips for using Portal Search" on page 714
View some useful tips for using Portal Search.

"Portlets for working with Search" on page 668
Get an overview of the portal search portlets.

Security considerations

Learn about the security aspects that you need to consider with regards to content search in IBM WebSphere Portal Express Version 8.5.

"Search on secured Portal Search collections and other secured content sources" on page 613
Portal search collections are portal resources. You can therefore secure them through Portal Access Control. Consequently, users can only access and search collections to which they have access permission.

"Searching on secured portal sites and pages and content management items" on page 613
For search on a secured local portal site, Portal Search provides a pre-configured search collection. You can also set up your own search collection for searching your portal site.

"Encrypting sensitive data" on page 614
When you create a new content source using the Manage Search portlet, some secured content sources require that you enter sensitive data. For example, this can be the user ID and password of the crawler user ID required for accessing the secured content source. When you later export the configuration of the search collection, you might want to protect the sensitive data. You can encrypt such sensitive data so that it is not stored as plain text on the hard drive. If you do not encrypt such data, the data is not included in the export.

"Configuring web server security" on page 615
For security reasons, the portal search seedlist by default redirects to HTTPs. Therefore, you need to configure your web server for HTTPs.

Search on secured Portal Search collections and other secured content sources:

Portal search collections are portal resources. You can therefore secure them through Portal Access Control. Consequently, users can only access and search collections to which they have access permission.

About this task

The Search Center portlet checks for access permissions of the searching users. It searches only in those search collections to which the users have access permission. For example, in the Search Center users see only the scopes containing search collections to which they have access.

Apply special care when giving users access through the search portlets to sensitive information such as web pages that have been collected from a secured website. Ideally, give only those users access permission for a search collection who have also access to the corresponding website that is being indexed and searched. Otherwise users might see documents listed, but cannot actually access them by clicking the respective link or URL. A user might see part of the information via the document summary, even if the user cannot open the document.

Searching on secured portal sites and pages and content management items:

For search on a secured local portal site, Portal Search provides a pre-configured search collection. You can also set up your own search collection for searching your portal site.

About this task

If you want to set up your own search collection for searching your portal site, you must apply special security considerations. Create a dedicated crawler user ID. Then, give that user ID access permission to the portlets and portal pages that you want to be available for search by users. For detailed information, refer to *Customizing your search collection for secured portal pages*.

Notes:

1. IBM Web Content Manager documents can be searched by portal users, as they are secured by Portal Access Control. Users can search documents to which they have the required access permissions on sites where search is enabled. For details about content security, refer to *User roles and access*. For information about configuring Portal Search on Web Content Manager, refer to *Indexing web content*.
2. Set the preferred language of the portal site crawler user ID to match the language of the portal site search collection that it crawls. If you set the language after you started a crawl, you must reset the portal site collection. Refer to *Creating or resetting the portal site collection*.

Related concepts:

"User roles and access" on page 1560
Different users will have different access to items and functions in your system depending on the role they are assigned. Roles can be assigned at the library level, and also assigned on individual items.

"Indexing web content" on page 616
To search for web content, your content must first be indexed by the WebSphere Portal search engine. When the content is indexed, you can run searches by using the search center or by using a search component. If you search for documents in the WebSphere Portal search center, be aware that you see search results for published documents only. Unpublished pending changes in a project are not included in the results.

Related tasks:

"Customizing your search collection for secured portal pages" on page 635
Set up your own customized search collection for searching a secured portal site.

"Resetting the default search collection" on page 711
Under certain circumstances, you might want to change the configuration of the portal site search collection. In this case, you must re-create the collection, as search collections cannot be modified.

Encrypting sensitive data:

When you create a new content source using the Manage Search portlet, some secured content sources require that you enter sensitive data. For example, this can be the user ID and password of the crawler user ID required for accessing the secured content source. When you later export the configuration of the search collection, you might want to protect the sensitive data. You can encrypt such sensitive data so that it is not stored as plain text on the hard drive. If you do not encrypt such data, the data is not included in the export.

About this task

For example, consider the case of content sources in the form of secured portal sites or HTTP sites that require a user ID and password. This sensitive data is stored on the portal server hard drive in plain text unless you choose to encrypt it. In order to ensure that such sensitive data is encrypted, perform the following procedure after portal installation:

Procedure

1. Copy the file `searchsecret.xml` to a temporary directory *temp* . The original file is located in the following directory:
   - For Linux: *PortalServer_root*`/search/wp.search.admin/bin`
   - For IBM i: *PortalServer_root*`/search/wp.search.admin/bin`
   - For Windows: *PortalServer_root*`\search\wp.search.admin\bin`
2.
3. Replace the string *CHANGE TO YOUR SECRET KEY* with a random string of your choice.
4. Run the updated file `searchsecret.xml` by using the XML configuration interface command:

   ```
   xmlaccess.sh|bat -in searchsecret.xml -out results.xml
                    -user wpsadmin -pwd wpsadmin
                    -url http://local_host:local_port/wps/config
   ```

   a. Specify the file name using the `-in` option.
   b. Specify a result file using the `-out` option.
   c. Check the result file to make sure that the XML request was executed successfully.

The script creates a slot called `search.secret` in the credential vault. Portal search uses this slot to encrypt the passwords configured for crawlers. If this slot does not exist, the password is saved as clear text on the portal server hard drive. The file `xmlaccess.sh|bat` is located in the directory *PortalServer_root*`/bin`. Example of the full command syntax for running the script `searchsecret.xml`:

```
PortalServer_root/bin/xmlaccess.sh|bat -in temp/searchsearchsecret.xml
     -user wpsadmin -password wpsadmin -url http://localhost:10039/wps/config
```

For more details about how to use the XML configuration interface, refer to the topics about the XML configuration interface, especially about Working with the XML configuration interface.

5. Delete the copied file `searchsecret.xml` that contains your encryption key.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Configuring web server security:

For security reasons, the portal search seedlist by default redirects to HTTPs. Therefore, you need to configure your web server for HTTPs.

About this task

For information about how to do this, go to the *Installing* section of the WebSphere Portal Express documentation. Select the appropriate operating system for your installation, and then the subsection for a stand-alone or clustered environment. Then, select the topic about *Preparing a remote Web server*.

What to do next

If you do not want to provide HTTPs security for your portal search seedlist, you can configure portal search to use HTTP only. To do this, run the configuration task `action-modify-servlet-transport-guarantee-none-wp.search.servlets/seedlist/servletEAR`. Use the following syntax, depending on your environment:

IBM i
```
ConfigEngine.sh action-modify-servlet-transport-guarantee-none-
wp.search.servlets/seedlist/servletEAR -DPortalAdminPwd=password
-DWasPassword=password
```

Linux
```
./ConfigEngine.sh action-modify-servlet-transport-guarantee-none-
wp.search.servlets/seedlist/servletEAR -DPortalAdminPwd=password
-DWasPassword=password
```

Windows
```
ConfigEngine.bat action-modify-servlet-transport-guarantee-none-
wp.search.servlets/seedlist/servletEAR -DPortalAdminPwd=password
-DWasPassword=password
```

For your updates to take effect, restart your portal server.

Indexing web content

To search for web content, your content must first be indexed by the WebSphere Portal search engine. When the content is indexed, you can run searches by using the search center or by using a search component. If you search for documents in the WebSphere Portal search center, be aware that you see search results for published documents only. Unpublished pending changes in a project are not included in the results.

Creating a content source for a site area

The WebSphere Portal search engine defines content sources that index your web content. All the child site areas and content items of the selected site area is included in the index. Related content sources are grouped in a search collection.

1. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
2. Select or create a new collection. The default search collection that is named **WebContentCollection** is provided by default.
3. Click **New Content Source**.
4. Select **WCM site** as the content source type.
5. Enter a name in the **Content Source Name** field.
6. Enter the following URL in the **Collect documents linked from this URL** field:

   For a stand-alone server:

```
http://hostname:port_number/wps/seedlist/myserver?SeedlistId=library/sitearea1/childsitearea2
&Source=com.ibm.workplace.wcm.plugins.seedlist.retriever.WCMRetrieverFactory&Action=GetDocuments
```

   You need to replace **hostname**, **port_number**, **library**, and **site area** with values appropriate for your site. If your library name or site area names contain spaces, you need to replace the spaces with a "+" symbol. For example, the path `library one/site area one` would instead be defined as `library+one/site+area+one`

   For a cluster:

   In this case you to use the host and port of the HTTP server:

```
http://httpserver:port_number/wps/seedlist/myserver?SeedlistId=library/sitearea1/childsitearea2
&Source=com.ibm.workplace.wcm.plugins.seedlist.retriever.WCMRetrieverFactory&Action=GetDocuments
```

   You need to replace **httpserver**, **port_number**, **library**, and **site area** with values appropriate for your site. If your library name or site area names contain spaces, you need to replace the spaces with a "+" symbol. For example, the path `library one/site area one` would instead be defined as `library+one/site+area+one`

   For a virtual portal configured to use the URL Context as its access point:

```
http://httpserver:port_number/wps/seedlist/myserver/virtualPortalContext?SeedlistId=library/sitearea1/childsitearea2
&Source=com.ibm.workplace.wcm.plugins.seedlist.retriever.WCMRetrieverFactory&Action=GetDocuments
```

   You need to replace **httpserver**, **port_number**, **virtualPortalContext**, **library**, and **site area** with values appropriate for your site. If your library name or site area names contain spaces, you need to replace the spaces with a "+" symbol. For example, the path `library one/site area one` would instead be defined as `library+one/site+area+one`

   For a virtual portal configured to use a different hostname as its access point:

```
http://vphostname:port_number/wps/seedlist/myserver/?SeedlistId=library/sitearea1/childsitearea2
&Source=com.ibm.workplace.wcm.plugins.seedlist.retriever.WCMRetrieverFactory&Action=GetDocuments
```

You need to replace **vphostname**, **port_number**, **library**, and **site area** with values appropriate for your site. If your library name or site area names contain spaces, you need to replace the spaces with a "+" symbol. For example, the path `library one/site area one` would instead be defined as `library+one/site+area+one`

Note: The seedlist ID can be any of the following:

- library
- library/site area
- library/site area/sub-site area/...
- the JCRID of a site area

7. If the content to be indexed is secured, go to the **Security** tab. Then, enter the user name and password of the user that is used to access the secured site. You must then click **Create** on the search tab itself.
8. If your site uses remote actions, you need to filter these actions out of your search index. Go to the **Filter** tab:
   a. Type a name in the **Rule Name** field
   b. Select **Apply rule while Collecting documents**
   c. Select the rule type of **Exclude**
   d. Select the rule basis of **URL text**
   e. Type `*&wcmAuthoringAction=*` in the **URL text** field
   f. Click **Create** in the Filter tab
9. Click **Create**.

If you have multiple parent site areas and want your searches to run across all site areas, create a content source for each of them in the same collection. If you do not want your searches to run across all parent site areas, create a separate collection for each parent site area or group of related parent site areas.

Searching web content in a virtual portal

Search services and search collections are separate for individual virtual portals and are not shared between individual virtual portals. You set up an individual search service and separate search collections for each virtual portal. These collections can be used to crawl and search the same set of documents.

If you are using a website that is shared across virtual portals, then to search that website in a virtual portal environment you must:

1. Create new search collection for the virtual portal. You can create a new content source by copying the URL from your original search collection.
2. Create new search component, or copy an existing search component, and configure it to use the new virtual portal search collection that is created in step 1.
3. Create new search form, by using an HTML component, which is configured to use the search component that is created in step 2.
4. Create new content item to display the HTML component that is created in step 3.

You must do these steps for each virtual portal in your system.

"Indexing web content in a multilingual environment" on page 618
Learn about the best practices for indexing web content if you are working with a multilingual Web Content Manager site.

Indexing web content in a multilingual environment

Learn about the best practices for indexing web content if you are working with a multilingual Web Content Manager site.

For example, if you work in French and Italian languages in your Web Content Manager site, consider the following recommendations:

1. Organize your web content in language-specific libraries. Create one library per language, for example, one library for French and one library for Italian. You can specify the language when you create the library.
2. Create language-specific search collections for the libraries you created. For example, create a search collection for your French library and a separate search collection for your Italian library. You can specify the language when you create the collection.
3. For each search collection, create one content source for each content source type. For example, for the French library search collection, create one content source for French site content and one for French Portal site content. Similarly, create content sources for your Italian library search collection as well.
   - To create a content source for site content, in the **Content source type** field, specify **WCM site**. In the **Collect documents linked from this URL** field, enter the URL that refers to the language-specific library that you want to use. Ensure that the language of your library matches the language of the search collection.
   - To create a content source for each of your supported portal site languages. In the **Content source type** field, specify **Portal site**. Add the content source to the search collection that you already created for the web content of the same language.

For more information about searching multilingual sites, go to Crawling a multilingual portal site in the related links.

Related tasks:

"Crawling a multilingual portal site" on page 631
View the steps to set up search on a multilingual portal for users with different language preferences.

Configuring Web Content Manager search options

You can edit the following search options to manage how the search service works with Web Content Manager search forms

About this task

You can edit the following search options to manage how the search service works with Web Content Manager search forms

Procedure

1. Open WebSphere Integrated Solutions Console to configure the **WCM SearchService** properties. Go to **Resources** > **Resource Environment** > **Resource Environment Providers**.
2. Select **WCM SearchService, Custom** properties.
3. Specify values for the search parameters.

   SearchService.DateFormatString
   This parameter is used to set the date format when dates are entered in

search forms and for displaying search results. Enter a supported Java date format string. If this property is not set, then the default format is *MMM dd yyyy HH:mm:ss z*.

SearchService.RecrawlInterval
This parameter is the recrawl interval in hours.

SearchService.BrokenLinksExpirationAge
This parameter is the default broken links expiration age in days.

SearchService.MetaFields
This parameter is used to specify extra elements to crawl when searching for metadata.

The format for this parameter value is: `elementName,key1`

To specify more than one metadata field maps, use the following format:

```
elementName1,key1;elementName2,key2;elementName3,key3
```

For example, to crawl for metadata in a text element named metaText:

- `SearchService.MetaFields=metaText,meta`
- `elementName` is the name of element you would like to search for metadata. Any valid element with that name in a searchable site area or content item is crawled.
- `key` is the "key" that is specified in an element tag that is used as part of a search element design. In the previous example, the key of "meta" is used. To render the content of the metaText element in a search element design, you would use the following tag: `<Element context="autoFill" type="content" key="meta"/>`

Note:

- Only text elements and short text elements can be searched.
- Only site areas that are configured to be searchable are crawled.

SearchService.SearchSeed.ExcludeFileAttachments=false
Set this parameter to `true` to prevent file resource component attachments from being included in the search results.

If set to false, the files that are stored in file resource elements in content items can also be searched. Files that are stored in file resource elements in a site area can also be searched so long as a default content item is selected.

SearchService.SearchSeed.excludeExtensions
Set this parameter to a list of file name extensions that you want to exclude from search results. Separate each item in the list with a comma. Any file resource component attachments and file resource element attachments that have these extensions are excluded from search results. For example, `SearchService.SearchSeed.excludeExtensions=avi,mpeg,zip`.

4. Click **Apply** and then **OK**.
5. Restart the portal for the new settings to take effect.
6. Delete and create a new search collection for changes to take effect.

Configuring Search Center to search for web content

You can use the Search Center to search for web content by adding a web content search collection to the Search Center.

Procedure

1. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
2. Click **Search Scopes**.
3. Click **New Scope**.
4. Click **Select Locations** and select your web content search collection.
5. Complete the search scope and click **OK**.

Language and region support in Portal Search

Depending on your portal environment and your users, you might want to make multilingual content available and searchable in your portal.

"Language support and multilingual sites"
Internet search engines usually take care of language support in terms of language determination and language specific processing.

"Region support and multi-regional sites"
Global companies might choose to publish content, and offer products or services specifically for certain regions. While today a language can be detected, there is no technical solution available to detect a region.

Language support and multilingual sites

Internet search engines usually take care of language support in terms of language determination and language specific processing.

The language information need not be carried in the URLs itself, nor in language specific metadata in the HTML source of the respective page. In most cases it is good practice to encode the content of pages by using UTF-8 encoding.

If the site is a multilingual site, then enable the web crawlers to get to the content in the various languages through links. Enabling language switching or selection based on either the cookie of a returning customer or using JavaScript to switch from one language to the other, will lock out web crawlers. Therefore in such a case the crawler picks up the content in the default language only. You can also check the webmaster recommendations published by the internet search providers for further details and suggestions.

Region support and multi-regional sites

Global companies might choose to publish content, and offer products or services specifically for certain regions. While today a language can be detected, there is no technical solution available to detect a region.

In this context you need to distinguish between the various regions; in this context the term 'region' is sometimes used synonymous with 'country'.

As was described for language identification in the topic about language support by Portal Search, Internet search engines also find that location or region specific metadata is not reliable enough to use it for region detection. Consequently crawlers expect the URL to carry region specific information. The most obvious is that the top-level domain name carries the country code and thereby the region

information. For example, a domain name such as XYZ-Co.fr or XYZ-Co.co.jp strongly indicates that the company resides or operates either in France or Japan.

If top-level domain names are used, no further action is required to allow region identification of that website. However, for more generic domain names such as .com or .eu , additional steps are required for enablement. In such cases you need to encode the region information in the URL, and pair it with tools provided by Internet search engines, such as Google Webmaster Tools for Geographic Targeting.

"Enabling region identification in WebSphere Portal Express"
You can use geographic targeting tools to associate a URL pattern with a specific region. However not all URL patterns are typically supported.

Enabling region identification in WebSphere Portal Express:

You can use geographic targeting tools to associate a URL pattern with a specific region. However not all URL patterns are typically supported.

About this task

URL patterns that are not supported have the region or country information encoded as a string value in a parameter list, such as www.XYZ-Co.com/products/information/?article=ABC123?country=UK . The following options are supported:

- Subdomains with generic domains, such as de.XYZ-Co.com
- Subdirectories with generic domains, for example www.XYZ-Co.com/es/...

You can configure top-level domain names with country codes and subdomains with generic domain by using standard portal configuration steps. However, when you write the region information into the relative path of a Portal URL, you need to adapt the portal site structure to differentiate between pages and page groups for a specific country. If you use Web Content Manager to deliver the content into the portal, this has implications as to how the content is managed and organized in the library or libraries. In both cases, the portal capabilities of rendering user friendly URLs is important. You have two options to handle this situation. They are described in the following.

Procedure

- The first option is based on the assumption that the site is split up at a certain point to reflect the different regions or countries. A typical scenario can be either an online shop or a product overview specific to the countries in which a company operates. Example URLs might then look as follows:
  - For the US: http://www.xyz-co.com/home/products/us/
  - For France: http://www.xyz-co.com/home/products/fr/
  - For Italy: http://www.xyz-co.com/home/products/it/

  To achieve this, set the friendly name of the top-level page that serves as the entry page for a specific country, with the appropriate country code as the identifier that is then published by means of the URL. Note that you need to assign friendly names in the whole hierarchy of all pages that can be identified as a page for a certain region. The previous example would result in two main unique portal pages:
  - home
  - products This has three region specifics pages: us , fr , and it .
- The second option has the same first two levels in its page hierarchy as the first: home and products . The products page renders the content based on user

selection or preference. Then Web Content Manager delivers the content for the chosen country. The association of a web page with a set of content items in the Web Content Manager library is done by using site areas. The names of the site areas represent the country codes. Three portal "content" pages are added to the `products` page associated with the site areas. In the example, this adds three region-specific portal pages that are associated with the site areas `us` , `fr` , and `it` . For more information, about friendly URLs see the topics about friendly URLs.

Related concepts:

"About friendly URLs for web content" on page 2041
With friendly URLs for web content, you can construct URLs to content items that are clear and concise. Although you can construct friendly URLs that reference web content items, IBM Web Content Manager itself does not generate friendly URLs by default. However, to cause the web content viewer to generate friendly URLs, you can create a plug-in that implements a content URL generation filter.

Related tasks:

"Using friendly URLs" on page 1280
You can associate friendly URLs with portal pages and labels. You and your users can use these friendly URLs to access specific portal pages or labels by using a human readable path, which is easy to remember.

Crawling web content with search seedlists

Portal Search supports the use of seedlists to make crawling websites and their metadata more efficient and to provide content owners fine-grained control over how content and metadata are crawled. You can configure the portal to use seedlist support when crawling content generated with IBM Web Content Manager.

About this task

By default Portal Search is configured to use seedlist format 1.0 when indexing content for search collections. When used with web content, seedlist format 1.0 makes it possible to use the web content page type to render content that is found in the search results on the corresponding web content page. You can also include custom metadata fields from a web content item that appears in the search seedlist but not in the HTML source.

Search seedlist 1.0 can make access control information available in a way that makes pre-filtering of contents possible. Pre-filtering provides the fastest filtering approach because it takes place in the search index level. An extra advantage of pre-filtering is that remote secured content sources can be searched from the portal. The filtering mode is defined as part of the search service configuration parameters.

Note: Support for generic seedlist 1.0 crawling is only available with IBM OmniFind® Enterprise Edition Version 9.1 and later.

"The search seedlist 1.0 format"
Portal Search is configured to support the IBM Web Content Manager search seedlist 1.0 format by default.

The search seedlist 1.0 format

Portal Search is configured to support the IBM Web Content Manager search seedlist 1.0 format by default.

About this task

Search seedlist 1.0 provides several features:

- You can use the web content page type to render content that is found in the search results on the corresponding web content page.
- You can include custom metadata fields from a web content item that appear in the search seedlist but not in the HTML source.
- You can search within a specific library or site area, across all web content libraries, or across a list of libraries.
- You can run incremental crawling of libraries for faster seedlist processing. With incremental crawling, when a crawl requests new items, only items that have been added, changed, or deleted since the previous crawl are retrieved.

Important: The syntax of the seedlist URL has changed with seedlist format 1.0. Older search collections that are created by using seedlist format 0.9 cannot be reused or migrated to the new format. Be sure that you index all your content again after you update the Web Content Manager seedlist format from 0.9 to 1.0.

Search seedlist 1.0 can make access control information available in a way that makes pre-filtering of contents possible. Pre-filtering provides the fastest filtering approach because it takes place in the search index level. Another advantage of pre-filtering is that remote secured content sources can be searched from the portal. The filtering mode is defined as part of the search service configuration parameters.

Restriction: The seedlist must only be used for search. For example, do not use the seedlist as an alternative logging method. For the purposes of event logging, the Java messaging service (JMS) is used.

"Enabling support for search seedlist 1.0"
If you want to use Portal Search to crawl your web content and use features like web content pages, you must enable seedlist 1.0 support for the Portal Search crawler.

"Customizing metadata field search support" on page 624
With the search seedlist 1.0 support, custom metadata fields that are specified on content items are added to the search seedlist as metadata information, without requiring the metadata to appear in the HTML source for the content items.

"Seedlist 1.0 REST service API" on page 625
The IBM Web Content Manager API for retrieving application content through a seedlist is based on the REST architecture style. To obtain seedlist content, third-party crawlers or administrator applications need to construct and send only HTTP requests to the application servlet.

Enabling support for search seedlist 1.0:

If you want to use Portal Search to crawl your web content and use features like web content pages, you must enable seedlist 1.0 support for the Portal Search crawler.

Procedure

1. Log in to the portal as an administrator.
2. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
3. To create a new search collection,

a. Click **Search Collections**.
b. Create a new search collection for your web content. Be sure that the new search collection uses the portal search service that is edited in the previous steps.

4. Add the following custom properties to the `WP ConfigService` resource environment by using the WebSphere Integrated Solutions Console:
   a. wcm.config.seedlist.version=1.0
   b. wcm.config.seedlist.servletpath=/seedlist
   c. wcm.config.seedlist.metakeys=<metakey1>,<metakey2> This is an optional step and is only required if you want to specify your own metadata.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Customizing metadata field search support:

With the search seedlist 1.0 support, custom metadata fields that are specified on content items are added to the search seedlist as metadata information, without requiring the metadata to appear in the HTML source for the content items.

Procedure

1. Log in to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Click **WP ConfigService**.
4. Under **Additional Properties**, click **Custom Properties**.
5. Click **New**, and enter the property name `wcm.config.seedlist.metakeys` , and set the string value to a comma-delimited list of your own metadata (for example, *<metakey1, metakey2>*).

   Add the names of the text element from your content to include in the search results to the `wcm.config.seedlist.metakeys` property. If you want to add more than one text element, separate them with commas. The name of the text element on your content item included in the search seedlist must match the name that is configured for this configuration key.

   For example, set `wcm.config.seedlist.metakeys=language,region` in the WP ConfigService resource environment provider, and add a IBM Web Content Manager text component as an element with the name `language` to a content item or authoring template. In your content item, you can enter the value `german` into the text component for the language. After you save the content item, the search crawler will add the value `german` into a metadata field that is called `language` within the search seedlist. Then, you can filter the search results based on your metadata information.
6. Click **OK**, and save the changes to the master configuration.
7. Restart the portal.

Seedlist 1.0 REST service API:

The IBM Web Content Manager API for retrieving application content through a seedlist is based on the REST architecture style. To obtain seedlist content, third-party crawlers or administrator applications need to construct and send only HTTP requests to the application servlet.

All REST API requests are synchronous calls. The order of the parameters in the requests does not matter. The parameter names are case-sensitive and must be entered in the format described here. An HTTP error response (for example, status code 404) is generated in the following situations:

- An unknown or unsupported parameter is submitted as part of the request.
- Web Content Manager cannot resolve the site area path or ID.
- Web Content Manager cannot find any items.
- The search seedlist enterprise application (Seedlist_Servlet) is not running.

The request is a standard HTTP GET command. The URL is formed by combining the seedlist servlet host name, port number, and path, followed by a collection of input parameters that are separated by ampersand (&) characters. The input parameters are entered as name-value pairs.

For example:

```
http://host_name:port_number/wps/seedlist/myserver?SeedlistId=library_list&Source=com.ibm.workplace.wcm.plugins.seedlist.retriever.WCMRetrieverFactory&Action=action&Range=number_of_entries
```

library_list
One or more web content libraries, which are separated by commas. If no value is specified, all libraries are used.

action The action to run on the request. The following actions are available:

GetDocuments
Retrieves a list of content items with their associated information.

number_of_entries
For each seedlist page that is returned, this value specifies the number of entries in the list of content items. If no value is specified, 100 items are returned.

Examples

In these examples, replace the following variables with values that are appropriate for your environment:

- *host_name*
- *virtual_portal_host_name*
- *http_server*
- *port_number*
- *library*
- *site_area*
- *site_area_id*

For the **SeedlistId** parameter, you can specify the value in the following formats:

- No value
- A specific library (for example, *library1*)
- A specific site area (for example, *site_area1*)

- A list of libraries, which are separated by commas (for example, *library1*,*library2*,*library3*)
- The JCRID of a site area

Retrieve a maximum of 100 items from a stand-alone server by using the path to the site area

```
http://host_name:port_number/wps/seedlist/myserver?SeedlistId=library/site_area&Source=com.ibm.workplace.wcm.plugins.seedlist.retriever.WCMRetrieverFactory&Action=GetDocuments
```

Retrieve a maximum of 200 items from a stand-alone server by using the ID of the site area

```
http://host_name:port_number/wps/seedlist/myserver?SeedlistId=site_area_id&Source=com.ibm.workplace.wcm.plugins.seedlist.retriever.WCMRetrieverFactory&Action=GetDocuments&Range=200
```

Retrieve a maximum of 100 items from a specific library

```
http://host_name:port_number/wps/seedlist/myserver?SeedlistId=library&Source=com.ibm.workplace.wcm.plugins.seedlist.retriever.WCMRetrieverFactory&Action=GetDocuments
```

Retrieve a maximum of 100 items from all libraries

Note: To use all libraries, leave `SeedlistId` value empty.

```
http://host_name:port_number/wps/seedlist/myserver?SeedlistId=&Source=com.ibm.workplace.wcm.plugins.seedlist.retriever.WCMRetrieverFactory&Action=GetDocuments
```

Retrieve a maximum of 100 items from a specified list of libraries

```
http://host_name:port_number/wps/seedlist/myserver?SeedlistId=library1,library2&Source=com.ibm.workplace.wcm.plugins.seedlist.retriever.WCMRetrieverFactory&Action=GetDocuments
```

Retrieve a maximum of 100 items from a cluster

Note: When referencing a cluster, specify the request with the host name and port number of the HTTP server.

```
http://http_server:port_number/wps/seedlist/myserver?SeedlistId=library/site_area&Source=com.ibm.workplace.wcm.plugins.seedlist.retriever.WCMRetrieverFactory&Action=GetDocuments
```

Retrieve a maximum of 100 items from a virtual portal that is configured to use the URL context as the access point

```
http://http_server:port_number/wps/seedlist/myserver/virtual_portal_context?SeedlistId=library/site_area&Source=com.ibm.workplace.wcm.plugins.seedlist.retriever.WCMRetrieverFactory&Action=GetDocuments
```

Retrieve a maximum of 100 items from a virtual portal that is configured to use a different host name as the access point

```
http://virtual_portal_host_name:port_number/wps/seedlist/myserver?SeedlistId=library/site_area&Source=com.ibm.workplace.wcm.plugins.seedlist.retriever.WCMRetrieverFactory&Action=GetDocuments
```

Important: You can access the REST API for the Web Content Manager search seedlist 1.0 with a secured connection (HTTPS) or with an unsecured connection (HTTP). Depending on the method, ensure that you use the correct port. However, if you access this REST API with an unsecured connection, you are automatically redirected to a secured connection.

Parameter	Default Value	Description
SeedlistID	No default; must be specified.	Identifies the seedlist. This parameter can be specified in the following ways: • An empty value causes all libraries to be used. • A specific library (for example, *library1*) • A specific site area (for example, *site_area1*) • A list of libraries, which are separated by commas. For example, `library1,library2,library3` • The JCRID of a site area
Start	0	Defines the start number for currently returned section.
Range	100	Defines the number of returned entries for current section.

Parameter	Default Value	Description
Date	No default. If not specified, all applicable results are returned.	Indicates that entries (documents) that were updated after this date are retrieved. The date format (compliant to standard ISO 8601) is the following : *date*T*timezone*, where *date* is yyyy-MM-dd, *time* is HH:mm:ss, and *zone* is ±hhmm. This format includes time zone information, which is critical if the client and server are in different time zones. **Important:** Proper HTML URL encoding must be performed (for example, represent the plus symbol + as %2B).
Action	GetDocuments	Defines requested action to execute. • GetDocuments retrieve all underlying documents. • GetNumberOfDocuments returns the number of all underlying documents, typically for debug purposes. This value must be the same as the number of all documents that are returned from an appropriate GetDocuments request.
Format	ATOM	Defines the output format : ATOM / HTML/ XML.
Timestamp	No default.	Indicates the content provider timestamp from a previous crawling session. The timestamp represents for the content provider some snapshot of the content and allows the crawler to get only the content changes on the next crawling. This parameter is used for incremental crawling.

Searching your local portal

View information on setting up your local portal for your users to search.

About this task

The portal default search collection combines two content sources and their related crawlers:

- The Portal Content Source. This contains the local portal site, where users can search for portal pages and portlets.
- The Web Content Manager (WCM) Content Source, which users can search for web content.

"Configuring a crawler to search your local portal site" on page 628
Configure and run a search crawler on your local portal site to gather information and create a search collection that enables your users to search your portal site.

"Crawling a multilingual portal site" on page 631
View the steps to set up search on a multilingual portal for users with different language preferences.

"Configuring search on a secured portal site" on page 632
Crawling and searching secured portal sites might require some additional configuration.

Configuring a crawler to search your local portal site

Configure and run a search crawler on your local portal site to gather information and create a search collection that enables your users to search your portal site.

About this task

Portal Search provides a default portal site search collection that enables your users to search your portal site. Before your users can search the portal site collection, do the following tasks.

Procedure

1. Set the crawler user ID. Set a dedicated crawler user ID for crawling the portal site content source. Proceed as follows:
   a. Define the crawler user ID by using the Manage Users and Groups portlet. Proceed as follows:

      Note: It is of benefit to define a dedicated crawler user ID. The pre-configured default portal site search uses the default administrator user ID `wpsadmin` with the default password of that user ID for the crawler. If you changed the default administrator user ID during your portal installation, the crawler uses that default user ID. If you changed the password for the `wpsadmin` or other administrative user ID, or if you changed the default administrator user ID to an ID other than `wpsadmin`, or if you want to use a separate user ID, you must set the crawler user ID.
   b. Set the preferred language of the portal site crawler user ID to match the language of the portal site search collection that it crawls. If you do this task after you started a crawl on the portal site search collection, you must reset the portal site collection. For details, see the topic about Resetting the default search collection.
   c. Edit the portal site collection content source and enter the crawler user ID and its password. To do this task, proceed as follows:
      1) To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
      2) Click **Search Collections.**.
      3) Select **Default Search Collection** from the Search Collection list.
      4) In the Content Source Name list of the search collection, click the **Portal Content Source** search collection.
      5) Click the **Edit** icon next to the Portal Content Source collection name.
      6) Select the **Security** tab.
      7) Click the **Edit** icon next to the security realm that you want to modify.
      8) Type the crawler user ID and password into the appropriate fields.
      9) Click **Update**.
      10) Click **Save** to save your changes.
   d. Optional: For content sources of type **Web Site**, you can configure the crawler to follow external links from inside the portal. To do this task, modify the value in the field **Levels of links to follow** under the tab **General Parameters**. Set the level to a value higher than 1. In addition, you can configure filters for those external links from the Filters tab. The default filter suppresses all links that point back to portal pages. The default filter is displayed only after you save the configuration of the content source.
2. Start the initial crawl. Start the initial crawl on the portal site content source:

a. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.Then, click **Search Collections**.
b. In the search collection list, click **Default Search Collection**.
c. Click the **Start Crawler** icon next to the Portal content source name.

3. Configure regular crawls. If you want regular crawls on the portal site content source, do either of the following tasks:
   - Enable the default scheduler. To do this task, proceed as follows:
     a. Click the **View Content Source Schedulers** icon next to the collection name.
     b. In the Manage Schedulers page, click **Disabled**. This action changes the status of the scheduler to Enabled and displays a confirmation message.
   - Set up your own scheduler. To do this task, proceed as follows:
     a. Click the **Edit** icon for the content source.

        Note: You can have only one schedule at a time. Therefore, to create your own schedule, you first must delete the existing schedule.
     b. Select the **Schedulers** tab.
     c. Configure your own scheduler as needed. For more information, see *Manage Search portlet help*.
     d. Click **Save** to save your changes.

Example

For more information about how to work with content sources, see *Managing the content sources of a search collection* and *Manage Search portlet help*.

Notes:

1. The local portal site is visible through a service that requires SSL. Therefore, if your portal is configured with a web server and you configure the content source root URL through the web server, you must configure the web server for SSL.
2. By default, items in the result lists from portal site searches provide no summary information. If you want to have the summary information that is added, configure the portlet with the summary parameter enabled as follows: `PortalCollectionSummarizer=on`.
3. When you crawl a portal site, be aware that a Portal Search crawl can use extended memory and time, depending on your Portal Search environment and configuration. For details, see the topic about Hints and tips for Portal Search crawls.
4. Do not change the default value of 1 for the option **Levels of links to follow**. Changing this value initiates web crawling logic and might result in unexpected results. For example, crawler might trigger unwanted in some of the administration portlets.
5. Set the preferred language of the crawler user ID to match the language of the search collection that it crawls.
6. The portal site search collection is created when an administrator goes to the Manage Search portlet. However, you must start the crawl for users to be able to search the portal site. Depending on your portal configuration and environment and possible customization, you might need to reset the portal site

search collection that was created. For details about such scenarios and the necessary tasks to perform see the topic about Resetting the default search collection.

7. If your users search the portal site search collection on a secured portal site, refer to the topic *Enabling search on a secured portal site with the default configuration*.
8. The portal search crawler indexes static content pages and all pages that include portlets.

When users search a portal site, they can access portal pages of two types:

- The Public or anonymous portal pages are pages that users can view without authentication by user ID and password. The crawler can crawl public pages on the portal site on which it is located, or on a remote portal.

  If you want anonymous users to be able to search the public pages of your portal site, see *Enabling anonymous users to search public pages of your portal*.
- The secured portal pages are pages that users can view only if they authenticate themselves to the portal by logging in to the portal with a user ID and password. For details, see *configuring search on a secured portal site*.

  Note: You can crawl, index, and search secured portal pages only on your local portal installation. For security reasons, you cannot crawl secured pages of one portal site from another portal site.

If you customize search on your portal site, you might find useful information under the topics about configuring the default location for search collections and Resetting the default search collection.

If your portal site is multilingual and your users use different languages to search your portal, see the topic about Crawling a multilingual portal site.

Related tasks:

"Resetting the default search collection" on page 711
Under certain circumstances, you might want to change the configuration of the portal site search collection. In this case, you must re-create the collection, as search collections cannot be modified.

"Managing the content sources of a search collection" on page 701
Search collections consist of one or more content sources. You can administer the content sources.

"Enabling search on a secured portal site with the default configuration" on page 633
By modifying some of the settings, you can use the default search collection to configure search of a secured portal site.

"Enabling anonymous users to search public pages of your portal" on page 641
You can enable anonymous users (sometimes also called unauthenticated users) to search public pages of your portal by using the portal Search Center portlet. Search by anonymous users works only on public pages of your portal, as the users are not logged in to your portal.

"Configuring search on a secured portal site" on page 632
Crawling and searching secured portal sites might require some additional configuration.

"Configuring the default location for search collections" on page 682
You can modify the default directory location under which search collections are created on a per search service basis. View some related information.

"Crawling a multilingual portal site"
View the steps to set up search on a multilingual portal for users with different language preferences.

Related reference:

"Hints and tips for Portal Search crawls" on page 717
View some useful tips about crawls that Portal Search performs. For example, crawling can require extended memory and time, depending on your Portal Search environment and configuration.

Related information:

 Configuring the web server plug-in for Secure Sockets Layer

Crawling a multilingual portal site

View the steps to set up search on a multilingual portal for users with different language preferences.

About this task

If your portal site is multilingual and your users use different languages to search your portal, you set up multiple search collections under one scope. Proceed as follows:

Procedure

1. Before you start creating search collections, make sure that the parameter `MULTILINGUAL_COLLECTION_ENABLED` is set to `false` for the search service. If the parameter has not been set, add it and set it to `false`.
2. **Note:** CF05 If you are using the combined cumulative fix readme 05 (CF05) or later, you do not need to complete this step.
   Create a separate crawler user ID for each language that your users might use for search. Set the language preference for each user ID to a different one of the required languages.
3. Create a portal content search collection for each language that your users might use. Set that language for the collection by selecting it from the pull-down list within **Specify Collection Language**.
4. Depending upon your Portal environment, do one of the following tasks:
   - If you are using the combined cumulative fix readme 04 (CF04) or earlier, create a single content source for each of these collections. For each content source, select the crawler user ID that you created before so that the language preference setting matches the language of the collection to which each content source belongs.
   - CF05 If you are using the combined cumulative fix readme 05 (CF05) or later, create a single content source for each of these collections. When you save the content source settings, Manage Search appends the locale information of the collection to the content source URL, unless you already added locale information.
5. **Note:** CF05 If you are using the combined cumulative fix readme 05 (CF05) or later, you do not need to complete this step.
   For each content source, append `&Locale=`*`locale_name`* to the content source URL. For example, for Spanish append `&Locale=es`.

6. Populate the collections by starting crawls on them. For load and performance reasons, run the crawls one after another rather than all at the same time. Refer to the topic about Hints and tips for Portal Search crawls.
7. Create a scope, and add all collections that you just created. You can name this scope `My preferred language`.

Results

- If you are using the combined cumulative fix readme 04 (CF04) or earlier, the Search Center portlet returns results only from the collection in the user's preferred language.
- CF05 If you are using the combined cumulative fix readme 05 (CF05) or later, the Search Center portlet returns results only from the collection in the user's preferred language, unless the Search Center is configured to display search results of all available content sources regardless of their language information. For more information on setting up search in multilingual sites, go to "Configuring search for multilingual sites" on page 684.

Related reference:

"Hints and tips for using Portal Search" on page 714
View some useful tips for using Portal Search.

"Hints and tips for Portal Search crawls" on page 717
View some useful tips about crawls that Portal Search performs. For example, crawling can require extended memory and time, depending on your Portal Search environment and configuration.

Configuring search on a secured portal site

Crawling and searching secured portal sites might require some additional configuration.

About this task

For search on secured portal sites Portal Search provides a pre-configured default setup. For more information, go to "Configuring a crawler to search your local portal site" on page 628. You can use that setup as is, or you can modify it as required. You can also set up your own search collection for search on portal sites. The following sections describe all of these options.

Security notes:

1. You can crawl, index, and search secured portal pages only on your local portal installation. For security reasons, you cannot crawl secured pages of one portal site from another portal site.
2. When you create a content source for enabling search, you must enter sensitive data. For example, a user ID and password for the crawler. This sensitive data is stored on the portal server in plain text unless you choose to encrypt it. To ensure encryption of this sensitive data when it is stored, update and run the `searchsecret.xml` file with the XML configuration interface before you enable search on the secured portal site. For information, refer to "Encrypting sensitive data" on page 614.
3. When users search a secured portal, the resulting portal pages or resources are filtered based on Portal Access Control. Portal Search filters the results according to the access permissions of the user who is searching. It applies to the following resources: portal pages, portlets, and Web Content Manager content. Portal Search does not provide security-filtering for other HTTP accessible information, such as secured websites. The portal cannot filter the resulting documents by portal security for these types of content source.

"Enabling search on a secured portal site with the default configuration"
By modifying some of the settings, you can use the default search collection to configure search of a secured portal site.

"Customizing your search collection for secured portal pages" on page 635
Set up your own customized search collection for searching a secured portal site.

Related tasks:

"Configuring a crawler to search your local portal site" on page 628
Configure and run a search crawler on your local portal site to gather information and create a search collection that enables your users to search your portal site.

"Encrypting sensitive data" on page 614
When you create a new content source using the Manage Search portlet, some secured content sources require that you enter sensitive data. For example, this can be the user ID and password of the crawler user ID required for accessing the secured content source. When you later export the configuration of the search collection, you might want to protect the sensitive data. You can encrypt such sensitive data so that it is not stored as plain text on the hard drive. If you do not encrypt such data, the data is not included in the export.

Enabling search on a secured portal site with the default configuration:

By modifying some of the settings, you can use the default search collection to configure search of a secured portal site.

About this task

In order for you to use Portal Search for searching your portal site, WebSphere Portal Express already prepared a search collection and a content source during installation. For more information, see the topic about configuring a crawler to search your local portal site. To enable that search collection on a secured portal site for search by users, encrypt the user ID and activate the search collection by starting the crawl and indexing process.

Procedure

1. To ensure encryption of the user ID and password for the crawler, update and run the file `searchsecret.xml` by using the XML configuration interface. For more information, see the topic about Encrypting sensitive data.
2. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
3. Click **Search Collections**, which opens the Search Collections panel.
4. From the list of search collections, click the portal site search collection **Default Search Collection**. This step opens the Content Sources panel for the portal site search collection. It lists the Portal Search content sources.
5. For the Portal Content Source, set the crawler user ID as described in the following procedure.

   Note: It is of benefit to define a dedicated crawler user ID. The pre-configured default portal site search uses the default administrator user ID `wpsadmin` with the default password of that user ID for the crawler. If you changed the default administrator user ID during your portal installation, the crawler uses that default user ID. If you want the crawler to use the user ID `wpsadmin` and its default password, you can omit the following substeps and proceed with the next main step. If you changed the password for the `wpsadmin` or other

administrative user ID, or if you changed the default administrator user ID to an ID other than `wpsadmin`, or if you want to use a separate user ID, proceed as follows:

a. For the Portal Content Source, click the **Edit** icon.
b. Update the user ID and password as needed in the Security tab.
c. Click **Save** to save your changes.

Note: Set the preferred language of the portal site crawler user ID to match the language of the portal site search collection that it crawls. If you already started a crawl on the portal site search collection, you must reset the portal site collection. For details, see the topic about Resetting the default search collection.

6. Click the **Start Collecting** icon to start the crawl. The crawler starts collecting and indexing portal pages. By default, the crawl is scheduled to run for 1 hour. The scheduler for regular repeated crawls is disabled by default. If you enable it, the interval for scheduled crawls is every hour. You can set these parameters by using the **Manage Search** portlet:
   a. You can change the duration of the crawl, depending on the size of your portal installation. Edit the portal site content source under **General Parameters**.
   b. You enable scheduled crawls by clicking the icon **View Content Source Schedulers** for the content source and clicking **Disabled** in the status column for the scheduler. The status changes to **Enabled**.
   c. You change the interval for scheduled crawls by editing the portal site content source, selecting the **Schedulers** tab, deleting the default scheduler, and defining a new one.

Results

Notes:

1. When you crawl a portal site, be aware that a Portal Search crawl can use extended memory and time, depending on your Portal Search environment and configuration. For details, see the topic about Hints and tips for Portal Search crawls.
2. If a user tried to use the Search Center by entering a search string in the portal search box in the theme and clicking search *before* an administrator enabled the portal site search collection, the user must log out of the portal and log back in again to be able to search the portal search collection. This action includes the administrator who enabled the portal search collection.

Related tasks:

"Configuring a crawler to search your local portal site" on page 628
Configure and run a search crawler on your local portal site to gather information and create a search collection that enables your users to search your portal site.

"Encrypting sensitive data" on page 614
When you create a new content source using the Manage Search portlet, some secured content sources require that you enter sensitive data. For example, this can be the user ID and password of the crawler user ID required for accessing the secured content source. When you later export the configuration of the search collection, you might want to protect the sensitive data. You can encrypt such sensitive data so that it is not stored as plain text on the hard drive. If you do not encrypt such data, the data is not included in the export.

"Resetting the default search collection" on page 711
Under certain circumstances, you might want to change the configuration of the portal site search collection. In this case, you must re-create the collection, as search collections cannot be modified.

Related reference:

"Hints and tips for Portal Search crawls" on page 717
View some useful tips about crawls that Portal Search performs. For example, crawling can require extended memory and time, depending on your Portal Search environment and configuration.

Customizing your search collection for secured portal pages:

Set up your own customized search collection for searching a secured portal site.

Procedure

1. Plan and determine which portlets and portal pages you want to be accessed by the crawler and made available for search. This must be the sum of all of the portal resources that you want to be available for search by all users that you want to be able to search those resources.
2. Make a conscious decision about the user ID that you use when you configure crawling and indexing, and which access permissions that user ID requires.
3. To ensure encryption of sensitive data that is stored, such as the crawler user ID and password, update and run the file `searchsecret.xml` by using the XML configuration interface. For details about encryption, see the topic about Encrypting sensitive data.
4. Create a dedicated crawler user with a user ID and password in the portal.
5. Give the crawler user ID the required access permissions to the portal resources as determined by the planning step: Access to all portlets and pages that must be indexed and thus be made available for search by users.
6. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
7. Click **Search Collections** to open the Search collections panel.
8. Create a search collection for the portal site.
9. Create the content source. The root URL for the portal is already completed. Select **Portal Site** for the type of content source. Complete the user ID and password for the crawler user that you created. Complete the other fields and select the options as needed.
10. To enable search on the portal site for users, click **Start Collecting**.

    Note:

    a. You can crawl, index, and search secured portal pages only on your local portal installation. For security reasons, you cannot crawl secured pages of one portal site from another portal site.

    b. Under normal circumstances, exclude all administrative portlets and all portlets that represent highly dynamic content from being accessible to the crawler user. This includes such portlets that contain action links with Delete actions. Otherwise, the Delete action can be started through the crawler.

    c. Instead of creating a new search collection and content source for the portal site search you can also use the collection and content source of the

default portal site search setup. In this case, replace the completed user ID and password in the Edit Content Source Configuration panel with the one that you want to use.

d. For details about how to create a search collection and a content source, see the topics *Creating and configuring search collections* and *Managing the content sources of a search collection*, and the *Manage Search* portlet help.

Related tasks:

"Encrypting sensitive data" on page 614
When you create a new content source using the Manage Search portlet, some secured content sources require that you enter sensitive data. For example, this can be the user ID and password of the crawler user ID required for accessing the secured content source. When you later export the configuration of the search collection, you might want to protect the sensitive data. You can encrypt such sensitive data so that it is not stored as plain text on the hard drive. If you do not encrypt such data, the data is not included in the export.

"Managing the content sources of a search collection" on page 701
Search collections consist of one or more content sources. You can administer the content sources.

Related reference:

"Creating and configuring search collections" on page 695
Get an overview of how you manage search collections and their content sources.

"Hints and tips for using Portal Search" on page 714
View some useful tips for using Portal Search.

Configuring your portal site for search by internet search engines

You can enable your portal site for search by using search services such as Google or Yahoo! Search on your portal site by external search services works for public portal pages only, that is, for pages that users can access without a user ID and password.

About this task

To make your portal available for search by internet search engines, you configure search with external search services. Register your portal with the external search service(s) of your choice. Give the URL of your portal to the search service providers so that they can crawl and index your portal site. Depending on how you customize your portal, you might consider indicating the Site Map page rather than your portal home page, or otherwise enhancing the configuration.

Optionally, you can configure several additional aspects for search on your portal by external search services:

- By default, the Site Map portlet is available on the portal Welcome page. This way the external crawler crawls only the links of the Site Map portlet. If you change your portal, place the Site Map portlet on your portal home page. Otherwise the crawler can crawl all portal links and might trigger unwanted actions by action links, such as a Delete button. If you do not want your users to be able to see the Site Map portlet, choose one of the following options for placing the Site Map portlet:
  - Place the Site Map portlet in the theme of the portal home page.
  - Place the Site Map portlet on a separate page, and omit that page from the portal navigation.

The location of the Site Map portlet in your portal determines the URL by which you register to the external search service.

- You can configure the Site Map portlet to determine the depth to which your portal can be searched.
- You can configure client identification for the external search engines.
- You can configure the URL normalization. You do this by setting a property in the portal configuration service named portal WP State Manager Service in the WebSphere Integrated Solutions Console.

"Configuring the Search Sitemap portlet for search by external search engines"
The **Search Sitemap** portlet generates a navigable list of all public pages of the portal. You can configure the **Search Sitemap** portlet to determine the limit to the number of links that are displayed per page.

"Client identification for search of the portal by external search engines" on page 639
For the portal to recognize external search engines, portal provides a client that covers several popular search engines. This client is implemented according to the Composite Capability/Preference Profiles (CC/PP) standard. It has the capability `HTML_SEARCH` set. If you want to add more search engines, you can configure the client as required.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Related reference:

"State Manager Service" on page 347
The portal State Manager Service is the access point for managing the navigational state of the portal. The navigational state represents the current view of portal resources as displayed to a user.

Configuring the Search Sitemap portlet for search by external search engines

The **Search Sitemap** portlet generates a navigable list of all public pages of the portal. You can configure the **Search Sitemap** portlet to determine the limit to the number of links that are displayed per page.

About this task

IBM WebSphere Portal Express provides the **Search Sitemap** portlet. It serves two purposes:

- You can use the Search Sitemap portlet to navigate the portal site. For more information, see *Managing pages*.
- The Search Sitemap portlet enables external search crawlers to collect portal pages more efficiently. This topic gives more information about configuring the Search Sitemap portlet for search by external search engines.

The Search Sitemap portlet generates a list of all public pages of the portal. You can configure the Search Sitemap portlet to determine the limit to the number of links per page. The Search Sitemap portlet lists portal pages to that maximum figure per page and then starts a new page. Set this figure to a maximum of 50 - 200 links per page for the portal site. For example, the Google search engine recommends fewer than 100 links per page.

Procedure

Configure the maximum number of links per page for search of the portal by setting the value for the parameter MAX_LINKS in the portlet preferences. You can do this either by using the administration portlet **Manage Portlets** or by updating the portlet preferences in the portlet.xml file of the Search Sitemap portlet.

Option	Description
Using the administration portlet Manage Portlets	Use this option to update the MAX_LINKS parameter for the Search Sitemap portlet. Proceed as follows: 1. To open the Manage Portlets portlet, click the **Administration menu** icon. 2. Then, click **Portlet Management** > **Applications**. 3. Locate the Search Sitemap portlet by searching for wp.ap.sitemap. 4. Click the **Configure portlet application** icon for the Search Sitemap portlet. 5. Add the MAX_LINKS parameter by typing it in the **New parameter:** field. 6. Click **OK** to save your updates.
Updating the portlet preferences in the portlet.xml file of the Search Sitemap portlet	Use this option to set the general default for the parameter MAX_LINKS. Proceed as follows: 1. Locate the WAR file of the Search Sitemap portlet in your portal installation. The WAR file is named sitemap.war. 2. Edit the file portlet.xml in that WAR file. 3. In the section portlet-preferences set the parameter MAX_LINKS to the required value, for example 50. Refer to this example code snippet: `<portlet-preferences>` `<preference>` `<name>MAX_LINKS</name>` `<value>50</value>` `</preference>` `</portlet-preferences>` 4. Redeploy the Search Sitemap portlet. 5. Restart your portal.

Results

What to do next

Note: The Search Sitemap portlet lists only public pages of your portal, that is pages, which users can access without logging in to the portal with a user ID and password. Secured portal pages are not available for search by anonymous users and therefore not listed by the Search Sitemap portlet.

Search on your portal by external search engines requires more configuration. For more information, see *Search by external search services* and *Client identification for search of the portal by external search engines*.

Client identification for search of the portal by external search engines

For the portal to recognize external search engines, portal provides a client that covers several popular search engines. This client is implemented according to the Composite Capability/Preference Profiles (CC/PP) standard. It has the capability `HTML_SEARCH` set. If you want to add more search engines, you can configure the client as required.

The client has been implemented with the following settings:

User agent:

```
(.*(B|b)ot.*)|(.*BOT.*)|(.*(S|s)pider.*)|(.*(S|s)earch.*)|
(.*(C|c)rawl(er)?.*)|(.*(G|g)rabber.*)|(.*(Y|y)ahoo.*)|
(.*(S|s)lurp.*)|(.*Lycos.*)|(.*Wget.*)
```

This user agent covers most available large search engines, such as Google, Yahoo!, Lycos, or MSN. This pattern list also accommodates all other search engines that include segments of `bot`, `spider`, `search`, or `crawler`.

Capability:

For each search engine that you want to be able to crawl your portal, you need to set the capability `HTML_SEARCH`. Search engines usually visit a website twice, the first time to crawl the site, and the second time to validate the content. When a search engine visits a site for the second time, it usually does so by using a normal browser. Therefore enter additional capabilities for supporting the different browser settings. Examples: `(HTML_4_0, HTML_IFRAME, HTML_FRAME, HTML_NESTED_TABLE, HTML_2_0, HTML_JAVASCRIPT, HTML_3_2, HTML_3_0, HTML_CSS, HTML_TABLE)`.

Manufacturer:

Search

Markup:

HTML

If you want to include search engines that are not covered by the default set, you can do so by using either the administration portlet Manage Clients or the XML configuration interface. For more information see the following topics.

Notes:

1. The search mechanism works correctly for the portal only if the search engine robots are identified to the portal in advance.
2. Search on your portal by external search engines requires additional configuration beyond client identification. For more details about this see the topics about *Configuring your portal site for search by external search services* and *Configuring the Search Sitemap portlet for search by external search engines*.

   "Adding search engines by using the administration portlet Manage Clients" on page 640
   To add search engines by using the administration portlet **Manage Clients**, follow the procedure that is given here.

   "Adding search engines by using the XML configuration interface" on page 640
   To add search engines by using the XML configuration interface, you import them by an XML script file. To make sure that the search mechanism works correctly, you need to add the capability `HTML_SEARCH`.

Adding search engines by using the administration portlet Manage Clients:

To add search engines by using the administration portlet **Manage Clients**, follow the procedure that is given here.

Procedure

1. Go to the Manage Clients portlet. Click the **Administration menu** icon. Then, click **Portal Settings** > **Supported Clients**. Portal opens the Manage Clients portlet.
2. Depending on whether you want to add more search clients to the default user agent or add a complete new client, do one of the following steps:
   - Select the client that starts with `(.*(B|b)ot.*)|(.*BOT.*)|(.*(S|s)pider.*)` `. . .` from the list of clients and edit it. Use this option if you simply want to add one or more search engines.
   - Add a new client. For example, you can use this option, if you want to give the newly added search engine priority by setting it to the **First** position in the client list.

   For details about how to do this refer to the Manage Clients portlet help.
3. Update or fill the fields and select the options as required. For brief descriptions of the available fields and options, see the list later in this topic. For a more detailed description of the fields and options, refer to the Manage Clients portlet help.
4. Click **OK** to save your changes.

What to do next

Fields and options in the manage Clients portlet help:

User agent:
Type your new search engine user agent.

Markup:
html

Manufacturer
Search Engine Manufacturer. This field is optional.

Capability:
`HTML_SEARCH, HTML_4_0, HTML_IFRAME, HTML_FRAME, HTML_NESTED_TABLE, HTML_2_0, HTML_JAVASCRIPT, HTML_3_2, HTML_3_0, HTML_CSS, HTML_TABLE`

Position:
First. Set the specified search engine to the first position so that it is correctly recognized. The reason for this is that the pattern matching for the comparison of the user agents to the supported clients is done from concrete and specific to general.

For a more detailed description of the fields and options, refer to the Manage Clients portlet help.

Adding search engines by using the XML configuration interface:

To add search engines by using the XML configuration interface, you import them by an XML script file. To make sure that the search mechanism works correctly, you need to add the capability `HTML_SEARCH`.

About this task

Here is an example XML script, with the HTML_SEARCH capability highlighted:

```
<?xml version="1.0" encoding="UTF-8"?>
<request    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:noNamespaceSchemaLocation="PortalConfig_1.4.xsd"
            type="update" create-oids="true">
    <portal action="locate">

        <client action="update" uniquename="wps.client.search.Your Search Engine Name"
                manufacturer="Your Search Engine Manufacturer" markup="html">

            <useragent-pattern>Your User-Agent Pattern</useragent-pattern>

            <client-capability update="set">HTML_SEARCH</client-capability>
            <client-capability update="set">HTML_4_0</client-capability>
            <client-capability update="set">HTML_IFRAME</client-capability>
            <client-capability update="set">HTML_FRAME</client-capability>
            <client-capability update="set">HTML_NESTED_TABLE</client-capability>
            <client-capability update="set">HTML_2_0</client-capability>
            <client-capability update="set">HTML_JAVASCRIPT</client-capability>
            <client-capability update="set">HTML_3_2</client-capability>
            <client-capability update="set">HTML_3_0</client-capability>
            <client-capability update="set">HTML_CSS</client-capability>
            <client-capability update="set">HTML_TABLE</client-capability>

        </client>
    </portal>
</request>
```

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Enabling anonymous users to search public pages of your portal

You can enable anonymous users (sometimes also called unauthenticated users) to search public pages of your portal by using the portal Search Center portlet. Search by anonymous users works only on public pages of your portal, as the users are not logged in to your portal.

About this task

To enable anonymous users to search public pages of your portal, you need to complete the following tasks:

- Make the Search Center portlet available on a public page of your portal so that users can access it without having to log in to the portal.
- Enable public sessions for your portal. The reason is that the Search Center portlet needs a valid session for its run time, and by default, sessions are not enabled on anonymous pages in the portal. By default, sessions are only created when a user authenticates and logs in to the portal.
- Have or create your own custom search scope. You can edit your own custom scopes, but you cannot modify the default search scopes to give anonymous users access.

Proceed as follows:

Procedure

1. Place the Search Center portlet on a public page. For more information, see the following subtopic.
2. Give the Anonymous Portal User group access permission to the Search portlet that you make available to anonymous users, and to the page on which that portlet is. To give access permission, you can use the User and Group Permissions portlet, the Resource Permissions portlet, or the Manage Portlets and Manage Pages portlets.
3. Give the Anonymous Portal User group access to the Search collections. To complete this step, use the User and Group Permissions portlet or the Resource Permissions portlet. From the list of Resource Types, select PSE Sources, then select the required search collections that you make available to public users, and assign the Anonymous Portal User access to those search collections.
4. Optional: Create a custom search scope, if you did not create one before already. For more information, see *Managing and administering Portal Search*, section *Creating a new search scope*.
5. Give the Anonymous Portal User group access to the Search scope:
   a. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
   b. Click **Search Scopes**
   c. Locate the scope that you want the anonymous user to use and click **Edit Search Scope**.
   d. Click **Yes** to enable the option **Visible to Anonymous Users**.
6. Enable public sessions by setting the parameter `public.session` to `true` in the portal Navigator Service in the WebSphere Integrated Solutions Console. For details about the portal Navigator Service and about how to set portal service configuration parameters see the respective topics.
7. Restart both WebSphere Application Server and WebSphere Portal Express for your changes to take effect.

"Placing the Search Center on a public portal page" on page 643
Depending on your environment, you might want to place the Search Center portlet on a public page of your IBM WebSphere Portal Express and have the search box in the portal theme take users who do a search to that public Search Center.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Related reference:

"Navigator Service" on page 328
The portal Navigator Service allows you to specify a number of settings; among these are properties for cache scope and cache expiration. Depending on your configuration, you might be able improve your performance by modifying these properties.

Related information:

Manage Search

Placing the Search Center on a public portal page

Depending on your environment, you might want to place the Search Center portlet on a public page of your IBM WebSphere Portal Express and have the search box in the portal theme take users who do a search to that public Search Center.

About this task

To do this, you need to create your custom search center page, place a copy of the Search Center on it, and adapt the file `search.jsp` accordingly. Proceed as follows:

Procedure

1. Create your custom search center page. Complete the fields and select options as required. Make sure to specify a Search and Tag Center profile for the page as follows:
   - If you create the page by using the Manage Pages functionality in the site toolbar, proceed as follows:
     a. From the site toolbar, select **Page** to edit the new search center page.
     b. From the General tab, select **Edit Page Properties**.
     c. In the Manage Page Properties window, select the **Advanced** tab.
     d. Scroll to the Metadata section of the Manage Page Properties window.
     e. To specify the Search and Tag Center profile for the custom search center page, add the key `resourceaggregation.profile` with the value of `profiles/profile_search_tag.json` to the list of key value pairs.

        Note: If you created the custom search center page as a child to the original Search Center page, the Search and Tag Center profile is inherited from the parent page.
     f. Click **Save**.
   - If you create the page by using the Manage Pages administration portlet, proceed as follows:
     a. Select **Edit Page Properties** for the page.
     b. Select **Advanced options**.
     c. Select **I want to set parameters**.
     d. Add the parameter `resourceaggregation.profile` with a value of `profiles/profile_search_tag.json` .
     e. Click **OK** > **OK**.
2. Add the Search Center portlet to the custom search center page by completing the following steps:
   a. From the custom search center page, select the **Create** tab in the site toolbar.
   b. From the Applications tab, select the **Search Center** portlet.
   c. Select **Add to Page ...** to add the Search Center portlet to the page.
3. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
4. Locate your custom Search Center page in the Manage Pages portlet.
5. Give your custom Search Center page a unique name by completing the following steps:
   a. Select **Edit Page Properties**.
   b. Specify a value for the **Unique name**.

c. Click **OK**.

6. Give the portlet window for the Search Center a unique name:
   a. Select **Edit Page Layout**.
   b. From the portlet menu, select **Set portlet window unique name**.
   c. Specify a value for the **Unique name**.
   d. Click **OK** > **Done**.
7. Change to the directory *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/modules/search/jsp
8. Edit the file search.jsp.
9. Locate the following code snippet that points to the default Search Center page:

```
<input type="hidden" name="uri" value="searchCenter:query">
<input type="hidden" name="contentNode" value="ibm.portal.Search Center">
<input type="hidden" name="layoutNode" value="ibm.portal.Search Center Portlet Window">
```

10. Change the values for the two parameters contentNode and layoutNode as follows:
    - Change the value for contentNode to the unique name of your custom Search Center page.
    - Change the value for layoutNode to the unique name that you gave to the window of the copy of your Search Center portlet.

    After your updates, the code snippet might look like the following:

```
<input type="hidden" name="uri" value="searchCenter:query">
<input type="hidden" name="contentNode"
     value="ibm.portal.your_public_search_center_page_unique_name">
<input type="hidden" name="layoutNode"
     value="ibm.portal.your_public_search_center_portlet_window_unique_name">
```

11. Restart your portal server.

Results

When your portal users do a search by using the search box in the theme, they are now directed to the Search Center on your public portal page.

Configuring your custom portal themes to include the search box

Enable your portal users to use the Portal Search box and Search portlet in your own custom theme.

The Portal Search box is included as part of the themes of a portal installation. If you use your own custom themes with your portal and want your users to be able to use the Portal Search box and Search portlet, make sure that your custom themes include the Portal Search box.

The code for including the search box in a theme is located in the file *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/modules/search/jsp/search.jsp . To make the search box available for users in your custom theme, do either of the following:

- Copy the contents of the file into your custom theme files.
- Copy the file into your theme and include it directly.

Redirecting search requests from a custom search form to the Search Center

If you plan to develop a custom search form, you might want to redirect search requests issued by the search form to the Search Center.

The following example form implements a button that is named **Search**:

```
<form name="myQueryForm" method="get" action=".">
     <input type="hidden" name="uri" value="searchCenter:query">
     <input type="hidden" name="contentNode" value="ibm.portal.Search Center">
     <input type="hidden" name="layoutNode" value="ibm.portal.Search Center Portlet Window">
     <input type="text" name="query">
     <input type="Submit" name="SearchButton" value="Search">
</form>
```

If a user selects the **Search** button, the form sends a request to the Portal server. The Portal server redirects the request to the Search Center by using the following parameters:

uri
This parameter must be set to `searchCenter:query` to address the query functionality of the Search Center portlet.

contentNode
This parameter is optional and specifies the unique name of the page where the Search Center portlet is placed. The default value is `ibm.portal.Search Center.`

layoutNode
This parameter is optional and specifies the unique name of the Search Center portlet window. The default value is `ibm.portal.Search Center Portlet Window`.

query
The Search Center uses the value of this parameter to search for the terms that a user specified.

If you placed the Search Center on a different page, for example, a public page, the values for the parameters **contentNode** and **layoutNode** might be different.

For more information, see *Placing the Search Center on a public portal page.*

Related tasks:

"Placing the Search Center on a public portal page" on page 643
Depending on your environment, you might want to place the Search Center portlet on a public page of your IBM WebSphere Portal Express and have the search box in the portal theme take users who do a search to that public Search Center.

Remote search service

You can configure the search portlets for local operation, or you can configure them for remote search service. Depending on your configuration, remote search service might have performance benefits by offloading and balancing system load.

You can provide the remote search service either as an EJB or as a web service through SOAP. Security can be enabled with EJB but not with SOAP. Also, separate WebSphere Portal environments cannot use the same remote search service. Only multiple WebSphere Portal nodes in the same cluster can use the same remote search service.

Note: SOAP support for remote search services was deprecated with WebSphere Portal Express Version 8.0.

When you want to index and search portal sites, search results are filtered according to the user security credentials. This filtering occurs independently of whether security is enabled on the remote search server or not. However, if security is not enabled, an unauthorized user can connect to the remote server and obtain unfiltered search results. If you want to prevent this issue, you must use EJB and enable security on the remote server. For information about enabling security on the remote search server, read *Preparing security for remote search service in a single-signon domain*.

"Installing remote search service by using IBM Installation Manager"
View the steps to install remote search service by using IBM Installation Manager.

"Installing remote search service by using manual steps" on page 647
You can install remote search service by using manual steps instead of the IBM Install Manager.

"Configuring user repositories on the remote search server" on page 652
The remote search server must have the same user repositories that are configured in WebSphere Application Server that are configured on the WebSphere Portal Express server. For example, if your WebSphere Portal Express server is configured to an IBM Directory Server LDAP in a federated repository configuration, then the remote search server must also be configured to the same IBM Directory Server LDAP in a federated repository configuration.

"Creating a single-sign on domain between WebSphere Portal and the remote search service" on page 652
View the steps to create a single-sign on (SSO) domain between WebSphere Portal Express and the remote search service. Set up remote search service by using EJB, since SOAP support for remote search services was deprecated with WebSphere Portal Express version 8.0.

"Setting the search user ID" on page 654
If you work with EJB on a secure server, you need to set the search user ID on the remote search server.

"Removing search collections" on page 656
If you plan to use search in a cluster, you must configure a remote search server. If you created any search collections, you must re-create them on the remote search server. If your search collection has data, export the collection before you delete it. Then, import it to the remote server.

"Configuring a remote search service" on page 657
Configure a remote search service for Portal Search.

"Configuring HTTP for the seedlist servlet" on page 661
Learn how to configure HTTP for the seedlist servlet. The seedlist servlet requires HTTPs by default. Therefore, when you access the servlet through HTTP, WebSphere Application Server redirects you to HTTPs.

Installing remote search service by using IBM Installation Manager

View the steps to install remote search service by using IBM Installation Manager.

Before you begin

- WebSphere Application Server v8.5 must be installed.
- WebSphere Portal Express and Web Content Manager must not be installed.

About this task

Use the WP8.5_iim.remotesearch installation package to install remote search service by using Installation Manager.

Procedure

1. Enter the installation directory of the remote search service in the installation directory field.
2. Select a language.
3. Select a package to install. You can select either remote search service, remote document conversion service (DCS), or both.
4. Enter the location of the target WebSphere Application Server root directory.
5. Specify which user is able to access and use remote search service from the portal server by entering their user name and password.

   Notes:

   - It is possible to manually install remote search service without using Installation Manager. For more information about how to perform the manual installation of remote search service, see "Installing remote search service by using manual steps."
   - Additional installation and configuration of remote DCS is no longer required with WebSphere Portal Express Version 8.5.
   - Installing remote search service by using Installation Manager will create a new WebSphere Application Server profile for remote search service.

Installing remote search service by using manual steps

You can install remote search service by using manual steps instead of the IBM Install Manager.

Before you begin

- For SOAP: If you use SOAP, the following security and performance considerations apply:
  1. Application security: If you use SOAP over a secure server, the SOAP service itself is not secure.
  2. Java 2 security: If you use SOAP, you must disable Java 2 security.

  Note: SOAP support for remote search services was deprecated with WebSphere Portal Express Version 8.0. EJB is still supported.
- For EJB: If you use EJB, complete the following security administration tasks:
  1. Prepare security for remote search service in a single-signon domain (SSO).
  2. Add the signer certification of the remote search service server into the portal search server. To do this addition, proceed by the following steps:
     a. Access the WebSphere Integrated Solutions Console of the portal search server.
     b. Click **Security** > **SSL certificate and key management** > **Key stores and certificates** > **NodeDefaultTrustStore** > **Signer certificates** > **Retrieve from port**.

        Cluster note: In a clustered environment, the path is **Security** > **SSL certificate and key management** > **Key stores and certificates** > **CellDefaultTrustStore** > **Signer certificates** > **Retrieve from port**.

c. Enter the remote search service server host name, its HTTPS port, and an alias.
d. Click **OK**.

Procedure

1. Copy `PseLibs.zip` and depending on the requirements of your environment, copy one of the two files `WebScannerSoap.ear` or `WebScannerEjbEar.ear` to the directory *AppServer_root*`/installableApps`. You find these files in the following locations of your portal installation:
   - The files `WebScannerSoap.ear` and `WebScannerEjbEar.ear` are in the directory *PortalServer_root*`/prereq/prereq.webscanner/installableApps/`
   - The file `PseLibs.zip` is located under directory *PortalServer_root*`/search/wp.search.libs/installableApps`
2. Depending on the requirements of your environment, install one of the two applications `WebScannerEJbEar.ear` or `WebScannerSoap.ear` on a remote server. For example, this can be `server1`. Proceed by the following steps:
   a. Access the WebSphere Integrated Solutions Console.
   b. Click **Applications** > **WebSphere Enterprise Application**.
   c. Click **Install**.
   d. Browse and select **WebScannerEjbEar.ear** or **WebScannerSoap.ear**, depending on whether you are using EJB or web service through SOAP.
   e. Click **Next**.
   f. On the following panels, accept the default settings.
   g. A message confirms that the application PSEStandalone (for EJB) or the application WebScannerEar (for SOAP) was installed successfully.
   h. Click **Save to Master Configuration**.
   i. Click **Save**.
3. Required: This step is required if you use Document Conversion Services.
   a. Install remote document conversion services on the remote server.
   b. Invoke the WebSphere Integrated Solutions Console and select **Environment** > **Shared Libraries**. Create a new shared library named `PSE` with a class path as follows:

```
$(APP_INSTALL_ROOT)/cell_name/dcs_war.ear/dcs.war/WEB-INF/lib/convertors.jar
$(APP_INSTALL_ROOT)/cell_name/dcs_war.ear/dcs.war/WEB-INF/lib/Export.jar
```

      where *cell_name* is the IBM WebSphere Application Server cell name where DCS is installed.
   c. Click **Apply** > **Save** > **Save** to save your changes.
4. Extract the Portal Search libraries to the remote server and add them to the class path on the remote server. To do this step, proceed as follows:
   a. Create a directory with the name *extract* under the directory `installableApps`.
   b. Locate the file `PseLibs.zip` in the directory `installableApps` and extract its content into the directory `extract` that you created in the previous step.
   c. Open the WebSphere Integrated Solutions Console.
   d. Click **Environment** > **Shared Libraries**.
   e. Create or modify the new shared library names PSE. When you create the library, check the option **Use an isolated class loader for this shared library**.

f. Add the libraries `extract/lib` to the class path by adding a new line to the class path and giving the full path: *AppServer_root*`/installableApps/extract/lib`. *AppServer_root* is the profile directory of your WebSphere Application Server installation. For example, this path can be:

```
/usr/WebSphere/AppServer/profiles/profile_name
```

g. Click **Apply** > **Save** to save your changes to the configuration.

5. Depending on the requirements of your environment, add a reference from the application `WebScannerEJbEar.ear` or `WebScannerSoap.ear` to the shared library. To add a reference, proceed as follows:
   a. Access the WebSphere Integrated Solutions Console of the remote server.
   b. Go to **Websphere enterprise applications**.
   c. For EJB: Click the application **PSEStandalone** > **Shared library references**.
      For SOAP: Click the application **WebScannerEar** > **Shared library references**.
   d. For EJB: On the window that opens up click the check box for **PSEStandalone**, then click the button **Reference shared library**.
      For SOAP: On the window that opens up click the check box for **WebScannerEar**, then click the button **Reference shared library**.
   e. From the **Available** list, select **PSE**.
   f. Click the appropriate arrow so that **PSE** displays in the **Selected** list.
   g. Click **OK** > **OK**.
   h. Save the configuration.
   i. For EJB: Restart the application PSEStandalone.
      For SOAP: Restart the application WebScannerEar.
6. On the WebSphere Integrated Solutions Console, determine the required values for configuring the portlet parameters, depending on whether you are using EJB or web service through SOAP:
   - For EJB: Determine the value for the port under **Servers** > **Server Types** > **WebSphere application servers** > ***YourAppServer1*** > **Communications** > **Ports** > **BOOTSTRAP_ADDRESS**.
   - For SOAP: Determine the value for the port number for the SOAP URL parameter. The appropriate port number for the SOAP URL parameter is the port on which the application server runs, in other words, the HTTP transport on which the remote server is configured to run. Determine the correct port number from **Application servers** > **server1** > **Ports** > **WC_defaulthost**. The `WC_defaulthost` value is 10014; therefore, if you did not change the default, you can use this value. Make sure that the port number that is set in the following file matches this port:

```
AppServer_root/installedApps/cell/WebScannerEar.ear/WebScannerSoap.war/
     wsdl/com/ibm/hrl/portlets/WsPSE/WebScannerLiteServerSOAPService.wsdl
```

     Replace the variables as follows:
     – *AppServer_root* is the profile directory of your WebSphere Application Server installation. For example, this directory can be:

```
/usr/WebSphere/AppServer/profiles/profile_name
```

     – ***cell*** is the cell name of your remote search computer.
     – ***WebScannerEar.ear*** is the name that you gave to the Enterprise Application when you installed the `WebScannerSoap.war` file.

     Edit the file and look for the port that is given in the value for the SOAP address location. Example:

```
<soap: address location="http://localhost:your_port_no/WebScannerSOAP/servlet/rpcrouter"/> .
```

In the example the port is *your_port_no*. The default value for the WC_defaulthost is 10014.

7. In the WebSphere Integrated Solutions Console, under **Resources** > **Asynchronous beans** > **Work managers**, create a new Work manager named PSEWorkManager with the following attributes:

```
Name:                          PSEWorkManager
JNDI Name:                     wps/searchIndexWM
Minimum Number of Threads:     20
Maximum number of Threads:     60
Growable =                     True (Ensure that the Growable check box is selected.)
Service Names:                 Application Profiling Service, WorkArea, Security, Internationalization
```

8. Click **Apply** > **Save** to save your changes to the configuration.
9. Start the application:
   a. Open the WebSphere Integrated Solutions Console.
   b. Click **Applications** > **Application Types** > **WebSphere enterprise applications**.
   c. Scroll to PSEStandalone or WebScannerEar. You can use the filter feature to search for these names.
   d. Click the check box and click **Start**. A message confirms that the application started successfully.
10. This step is required only if you work with EJB on a secure server: "Setting the search user ID" on page 654.
11. Restart the WebSphere Application Server.
12. In the portal server enable CSIv2 identity assertion. To complete this step, proceed as follows:

    Cluster note: In a clustered environment, complete these steps on the Deployment Manager WebSphere Integrated Solutions Console.

    a. Enable CSIv2 Identity Assertion on the outbound connection:
       1) Access the WebSphere Integrated Solutions Console of the portal server.
       2) Go to **Security** > **Global Security** > **RMI/IIOP security** > **CSIv2 outbound communications**.
       3) Check **Use identity assertion**.
       4) When you are done, restart the portal server.
    b. Enable CSIv2 Identity Assertion on the inbound connection:
       1) Access the WebSphere Integrated Solutions Console of the remote server.
       2) Go to **Security** > **Global Security** > **RMI/IIOP security** > **CSIv2 inbound communications**.
       3) Check **Use identity assertion**.
       4) Under **Trusted identities**, enter either an asterisk (*) or the identity of the portal server.
       5) When you are done, restart the remote server.

       For more detailed information, refer to the WebSphere Application Server information center.
13. Back on your portal, configure Portal Search for remote search service.

    Cluster note: In a clustered environment, complete this step on the primary node only.

"Updating remote search service by using manual steps"
If you originally installed the remote service by using manual steps, then you must use manual steps to upgrade it after you apply the combined cumulative fix on the portal server.

Updating remote search service by using manual steps:

If you originally installed the remote service by using manual steps, then you must use manual steps to upgrade it after you apply the combined cumulative fix on the portal server.

Before you begin

Apply the combined cumulative fix on the portal server. For more information about applying the combined cumulative fix, go to IBM WebSphere Portal V8.5.0.0 combined cumulative fix instructions: remote search.

Procedure

1. Copy `PseLibs.zip` and depending on the requirements of your environment, copy one of the two files `WebScannerSoap.ear` or `WebScannerEjbEar.ear` to the directory *AppServer_root*`/installableApps`. You find these files in the following locations of your portal installation:
   - The files `WebScannerSoap.ear` and `WebScannerEjbEar.ear` are in the directory *PortalServer_root*`/prereq/prereq.webscanner/installableApps/`
   - The file `PseLibs.zip` is located under directory *PortalServer_root*`/search/wp.search.libs/installableApps`
2. Depending on the requirements of your environment, update one of the two applications `WebScannerEJbEar.ear` or `WebScannerSoap.ear` on a remote server. For example, this can be `server1`. Proceed by the following steps:
   a. Access the WebSphere Integrated Solutions Console.
   b. Click **Applications** > **WebSphere Enterprise Application**.
   c. Depending upon what you originally installed, specify either **PSEStandalone** or **WebScannerEar**.
   d. Click **Update**.
   e. Specify the option to **Replace the entire application**.
   f. Browse and select **WebScannerEjbEar.ear** or **WebScannerSoap.ear**, depending on whether you are using EJB or web service through SOAP.
   g. Click **Next**.
   h. On the following panels, accept the default settings.
   i. A message confirms that the application PSEStandalone (for EJB) or the application WebScannerEar (for SOAP) was updated successfully.
   j. Click **Save to Master Configuration**.
   k. Click **Save**.
3. Extract the Portal Search libraries to the remote server and add them to the class path on the remote server. To do this step, proceed as follows:
   a. Delete the contents of the directory `installableApps/extract` that you created when you first installed the remote search service.
   b. Locate the file `PseLibs.zip` in the directory `installableApps` and extract its contents into the empty directory `installableApps/extract`.
   c. Open the WebSphere Integrated Solutions Console.
   d. Click **Environment** > **Shared Libraries**.

e. Create or modify the new shared library named PSE. When you create the library, check the option **Use an isolated class loader for this shared library**.

f. Click **Apply** > **Save** to save your changes to the configuration.

4. Restart the WebSphere Application Server.

Configuring user repositories on the remote search server

The remote search server must have the same user repositories that are configured in WebSphere Application Server that are configured on the WebSphere Portal Express server. For example, if your WebSphere Portal Express server is configured to an IBM Directory Server LDAP in a federated repository configuration, then the remote search server must also be configured to the same IBM Directory Server LDAP in a federated repository configuration.

After the user repositories are configured the same between the WebSphere Portal Express server and the remote search server, it is possible to configure a dedicated search user from the repository as described in *Setting the search user ID*.

Follow the WebSphere Application Server product documentation *Configuring Lightweight Directory Access Protocol in a federated repository configuration* to configure LDAP settings on the remote search server.

After configuring user repositories on the remote search server, you must enable single-sign on (SSO) between WebSphere Portal Express and the remote search server. For details about how to do this step, refer to *Creating a single-sign on domain between WebSphere Portal and the remote search service*.

Related tasks:

"Setting the search user ID" on page 654
If you work with EJB on a secure server, you need to set the search user ID on the remote search server.

"Creating a single-sign on domain between WebSphere Portal and the remote search service"
View the steps to create a single-sign on (SSO) domain between WebSphere Portal Express and the remote search service. Set up remote search service by using EJB, since SOAP support for remote search services was deprecated with WebSphere Portal Express version 8.0.

Related information:

Configuring Lightweight Directory Access Protocol in a federated repository configuration

Creating a single-sign on domain between WebSphere Portal and the remote search service

View the steps to create a single-sign on (SSO) domain between WebSphere Portal Express and the remote search service. Set up remote search service by using EJB, since SOAP support for remote search services was deprecated with WebSphere Portal Express version 8.0.

Procedure

1. Export the LTPA keys from the WebSphere Portal Express server by completing the following steps.

   Cluster note: In a clustered environment, complete these steps on the Deployment Manager.

   a. Open the WebSphere Integrated Solutions Console.

b. Select **Security** > **Global Security** > **Authentication** > **LTPA**.
c. Enter a password for the key.
d. In the field for the fully qualified key name, enter a key file name and click the **Export keys** button. The keys are written to the file *profile_root*/Key File Name, where *portal_root* is either the Deployment Manager profile or the WebSphere Portal Express profile.

2. Import the key file to the remote search server. If your environment contains extra application servers, complete the following steps on all other servers that you want to be a part of this SSO domain:
   a. Copy the key file that you exported in step 1 from the WebSphere Portal Express server to the remote search server.
   b. Log in to the WebSphere Integrated Solutions Console.
   c. Select **Security** > **Global Security** > **Authentication** > **LTPA**.
   d. In the field for the fully qualified key name, enter the directory and key file name that you specified in step 2a and click **Import keys**. The keys are propagated to all servers of the SSO domain.
   e. Restart all WebSphere Application Server profiles on this server.
3. Ensure that automatic LTPA key generation is disabled on all servers of the SSO domain by completing the following steps:
   a. Log in to the WebSphere Integrated Solutions Console.
   b. Select **Security** > **Global Security**. In the Authentication mechanisms and expiration pane, click **LTPA**.
   c. Within **Key generation**, select **Key set groups**
   d. Click **NodeLTPAKeySetGroup**.

      Cluster note: In a clustered environment, click **CellLTPAKeySetGroup**.
   e. In the Key generation pane, disable the **Automatically generate keys** check box.
   f. Click **OK**.
   g. Click **Save** to save your changes to the master configuration.
   h. Log out of the WebSphere Integrated Solutions Console.
4. Verify that the system clocks are within 5 minutes of each other between the WebSphere Portal Express server or servers and the remote search service server.

   Note: Failure to have the clocks in sync will lead to an import failure in the next step.
5. Add the signer certification of the remote search service server into the portal server by completing the following steps:
   a. Access the WebSphere Integrated Solutions Console of the portal server.

      Cluster note: In a clustered environment, complete these steps on the Deployment Manager.
   b. Click **Security** > **SSL certificate and key management** > **Key stores and certificates** > **NodeDefaultTrustStore** > **Signer certificates** > **Retrieve from port**.

      Cluster note: In a clustered environment, the path is **Security** > **SSL certificate and key management** > **Key stores and certificates** > **CellDefaultTrustStore** > **Signer certificates** > **Retrieve from port**.

c. Enter the remote search service server host, its SSL port, and an alias.
d. Click **Retrieve Signer Information**.
e. Click **OK**.

6. Add the signer certification of the portal server into the remote search service server by completing the following steps:
   a. Access the WebSphere Integrated Solutions Console of the remote search service server.
   b. Click **Security** > **SSL certificate and key management** > **Key stores and certificates** > **NodeDefaultTrustStore** > **Signer certificates** > **Retrieve from port**.
   c. Enter the portal server host, its SSL port, and an alias.
   d. Click **Retrieve Signer Information**.
   e. Click **OK**.
7. In the portal server enable CSIv2 identity assertion. To complete this step, proceed as follows:

   Cluster note: In a clustered environment, complete these steps on the Deployment Manager WebSphere Integrated Solutions Console.
   a. Enable CSIv2 Identity Assertion on the outbound connection:
      1) Access the WebSphere Integrated Solutions Console of the portal server.
      2) Go to **Security** > **Global Security** > **RMI/IIOP security** > **CSIv2 outbound communications**.
      3) Check **Use identity assertion**.
      4) When you are done, restart the portal server.
   b. Enable CSIv2 Identity Assertion on the inbound connection:
      1) Access the WebSphere Integrated Solutions Console of the remote server.
      2) Go to **Security** > **Global Security** > **RMI/IIOP security** > **CSIv2 inbound communications**.
      3) Check **Use identity assertion**.
      4) Under **Trusted identities**, enter either an asterisk (*) or the identity of the portal server.
      5) When you are done, restart the remote server.

      For more detailed information, refer to the WebSphere Application Server information center.

What to do next

For more details about exporting the LTPA token, refer to the WebSphere Application Server information center by going to**Administering** > **Security** > **Managing security** > **Configuring authentication mechanisms** > **Configuring Lightweight Third Party Authentication** > **Lightweight Third Party Authentication settings**. You can also locate this topic by opening the search feature of the WebSphere Application Server information center and searching for `ltpa key export`.

If you work with EJB on a secure server, you must set the search user ID. For details about how to do this step, refer to "Setting the search user ID."

Setting the search user ID

If you work with EJB on a secure server, you need to set the search user ID on the remote search server.

Before you begin

Ensure that your **SearchAdminUser** alias matches your WebSphere Portal Express administrator information. Complete the following steps to view or change the information that is stored in your **SearchAdminUser** alias:

1. Log in to the WebSphere Integrated Solutions Console.
2. Click **Security** > **Global security**.
3. In the Authentication section, click **Java Authentication and Authorization Service** > **J2C authentication data**.
4. Edit the **SearchAdminUser** alias.
5. Update the user ID and/or password to match your WebSphere Portal Express administrator information.
6. If you are working in a clustered environment, you must synchronize the nodes of your cluster. To synchronize the nodes of your cluster, complete the following steps:
   a. Log on to the Deployment Manager.
   b. Go to **System Administration** > **Nodes**.
   c. Select the nodes to synchronize from the list.
   d. Click **Full Resynchronize**.
7. Restart the portal servers.

About this task

In a remote search environment, the remote search server must be configured for the same LDAP as WebSphere Portal Express.

Procedure

1. Open the WebSphere Integrated Solutions Console.
2. Click **Applications** > **Application Types** > **WebSphere enterprise applications**.
3. Locate the application **PSEStandalone**.
4. Enter this application and click **Security roles to user/group mapping**.
5. Select the role **SearchUser** and click **Map users**.
6. On the portal that accesses the EJB, search for the user ID that is set as the WebSphere Application Server Admin User. For example, you can find this ID by using the following procedure:
   a. Open the WebSphere Integrated Solutions Console.
   b. Click **Global Security** > **Federated LDAP registry**.
   c. Determine the primary administrative user ID.
7. To continue installing remote search service manually, proceed by the following steps:
   a. Return to the WebSphere Integrated Solutions Console of the machine where the EJB is installed.
   b. Type the name that you found as the user ID in a previous step as the search string and click **Search**. As a result, the user ID and its configuration parameters are displayed in the **Available** box.
   c. Add this user ID to the **Selected** box by clicking the double-angled bracket **> >** button.
   d. Click **OK** to save your updates.

e. Restart the WebSphere Application Server on which the PSEStandalone is installed.

Removing search collections

If you plan to use search in a cluster, you must configure a remote search server. If you created any search collections, you must re-create them on the remote search server. If your search collection has data, export the collection before you delete it. Then, import it to the remote server.

Procedure

1. Prevent the creation of search collections. To do so, complete the substeps listed here.

   Note: CF07 If you have CF07 or a later fix pack installed, start the portal configuration engine task `suppress-automatic-search-service-creation` to prevent the creation of search collections. You do not need to perform the substeps listed here. After running the configuration task, continue with the next main step that deletes all search collections from the primary node.

   a. Log in to the WebSphere Integrated Solutions Console.

      Cluster note: If this web content server is part of a cluster, log on to the Deployment Manager WebSphere Integrated Solutions Console.

   b. Go to **Resources** > **Resource Environment** > **Resource Environment Providers** > **JCR ConfigService PortalContent** > **Custom properties**.

   c. Set the **`jcr.textsearch.enabled`** parameter to `false`.

   d. Go to **Resources** > **Resource Environment** > **Resource Environment Providers** > **WP ConfigService** > **Custom properties**.

   e. Set the `search.service.suppress_automatic_creation` property to `true`. If the property does not exist, create it.

   f. Click **Apply** and save your changes to the master configuration.

   g. Restart your server.

2. Delete all existing search collections from the primary node. To do so, complete the following substeps:

   a. Log on to WebSphere Portal Express.

   b. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.and then click **Search Collections**.

   c. Click the **Delete Collection** icon for each search collection and then click **OK** until they are all deleted.

   d. Restart the WebSphere_Portal server and then go back to the **Search Collections** page to verify that all search collections are deleted.

Related tasks:

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

Related reference:

"Configuration Service" on page 297
The portal Configuration Service is responsible for collecting the most essential configuration data of the IBM WebSphere Portal Express engine.

Configuring a remote search service

Configure a remote search service for Portal Search.

About this task

To configure a remote search service for Portal Search, proceed as follows:

Cluster note: In a clustered environment, complete these steps on the primary node.

Procedure

1. Log in to your portal as an administrator.
2. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
3. Click **Search Services**.
4. Click **New Search Service**.
5. For the **Search service implementation** select **Portal Search Service Type**.
6. To configure a remote search service by using EJB, proceed as follows:
   a. Edit the search service parameter `PSE_TYPE` and change its value to `ejb`.
   b. Modify the parameter `IIOP_URL`. Set its value to `iiop://` *`your_ejb_search_server.your.example_domain.com:port`*, where *`your_ejb_search_server.your.example_domain.com`* is the name of the remote search server and *`port`* is the port number that you obtained in the step to determine the port for EJB under Preparing for remote search service. For example, this can be `iiop://ejb_server.your_company.com:2809`.
   c. Modify the parameter `EJB`. Set it to the following value: `ejb/com/ibm/hrl/portlets/WsPse/WebScannerLiteEJBHome`. This is the default JNDI name. If you modified the EJB parameter to a JNDI name of your own choice, use that name instead.
7. Modify the parameter `DefaultCollectionsDirectory` to the portal search service. Use it to determine the default directory where your search collections are created on the server that hosts the remote search service. This parameter does not have a default value.
8. Add the parameter `CONFIG_FOLDER_PATH` to the portal search service. Use it to determine where the configuration data for search collections is stored on the server that hosts the remote search service. The default is *`wp_profile_root`*`/CollectionsConfig`.
9. Depending on, whether you are using EJB or web service through SOAP, type `Remote PSE service EJB` or `Remote PSE service SOAP` for the service name.

   Note: Configuring for remote search service as a web service through SOAP is not supported on Portal 8.5.
10. Click **OK** to save the new search service. The **Manage Search** portlet now lists the new search service in the list of search services. A green check in the status column indicates that the new search service is working correctly. If the search service is not working properly, it has a red cross, and a message is displayed. Click the **View details** link of the message for more information about the problem and how to resolve it.
11. Restart all servers in your configuration for your changes to take effect.

What to do next

For more information about the search service configuration parameters, see to "Search service configuration parameters" on page 674.

1. "Creating a new search service"
   Learn about creating a new search service for Portal Search.
2. "Creating new search collections" on page 659
   Before you can begin using remote search service, you must create two new search collections, one for JCR search, and one for Portal search.
3. "Creating a new content source" on page 660
   Before you can begin using remote search service, you must create three new content sources, one for the Web Content Manager, one for your portal site, and one for JCR search.

Related tasks:

Preparing for remote search service
Get an overview of how you prepare your portal system for remote search service. You can provide remote search service by either using EJB or SOAP.

Related reference:

"Search service configuration parameters" on page 674
Learn about the portal search service parameters and possible values.

Creating a new search service:

Learn about creating a new search service for Portal Search.

About this task

Cluster note: In a clustered environment, complete these steps on the primary node.
For more detailed information about the search service configuration parameters refer to "Search service configuration parameters" on page 674.

Procedure

1. Log in to your portal as an administrator.
2. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**. Then, click **Search Services**.
3. Click **New Search Service**.
4. Depending on whether you are using EJB or Web service via SOAP, type `Remote PSE service EJB` or `Remote PSE service SOAP` for the service name.

   Note: Configuring for remote search service as a Web service via SOAP is not supported on Portal 8.5.
5. For the **Search service implementation** select **Portal Search Service Type**.
6. To configure a remote search service by using EJB, proceed as follows:
   a. Edit the search service parameter `PSE_Type` and change its value to `ejb`.
   b. Modify the parameter `IIOP_URL`. Set its value to `iiop://` *`your_ejb_search_server.your.example_domain.com:port`*, where *`your_ejb_search_server.your.example_domain.com`* is the name of the remote search server and *`port`* is the port number that you obtained in the

step to determine the port for EJB under Preparing for remote search service. For example, this can be `iiop://ejb_server.your_company.com:2809`.

c. Modify the parameter `EJB`. Set it to the following value: `ejb/com/ibm/hrl/portlets/WsPse/WebScannerLiteEJBHome`. This is the default JNDI name. If you have modified the EJB parameter to a JNDI name of your own choice, use that name instead.

7. Modify the parameter `DefaultCollectionsDirectory` to the portal search service. Use it to determine the default directory where your search collections are created on the server that hosts the remote search service. This parameter does not have a default value.
8. Add the parameter `CONFIG_FOLDER_PATH` to the portal search service. Use it to determine where the configuration data for search collections is stored on the server that hosts the remote search service. The default is *wp_profile_root*`/CollectionsConfig`.
9. Click **OK** to save the new search service. The **Manage Search** portlet now lists the new search service in the list of search services. A check mark in the status column indicates that the new search service is working correctly. If the search service is not working properly, it has a red cross, and a message is displayed. Click the **View details** link of the message for more information about the problem and how to resolve it.
10. Restart all servers in your configuration for your changes to take effect.

What to do next

Manually create new search collections for JCR search and Portal search.

Creating new search collections:

Before you can begin using remote search service, you must create two new search collections, one for JCR search, and one for Portal search.

Procedure

1. Manually create a JCR search collection by following the instructions that are outlined in *Setting up a JCR search collection* in the related links.

   Important: The steps that are outlined in *Setting up a JCR search collection* include instructions on how to create the new content source for the JCR search collection. Create the new content source for the JCR search collection before you create the Portal search collection.
2. Manually create a Portal search collection by completing the following steps:
   a. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
   b. Click **Search collections.**
   c. To create a new search collection, click **New collection**.
   d. Specify the following values for the parameters as required:

      Search Service
      Select **Remote Search Service**.

      Location of collection
      The directory location for the collection where you intend the search collection to be created. This parameter is to be specified as *index directory location*`/`*collection name*.

Name of collection.
Specify the name of the collection.

Description of collection
This parameter is optional.

Specify Collection language
Specify the collection language. By default this parameter is set to English (United States).

e. Click **OK**. The collection displays in the Collections from All Services pane.

What to do next

Manually create new content sources for the Portal search collection.

Related tasks:

"Setting up a JCR search collection" on page 698
A JCR search collection is a special purpose search collection that is used by WebSphere Portal applications. It is not designed to be used alongside user-defined search collections. A JCR search collection requires a special setup. This setup includes the creation of a new content source for the search collection. Under normal circumstances, you do not need to re-create the JCR search collection. However, in rare cases you might need to re-create it, for example if you deleted the default JCR search collection.

Creating a new content source:

Before you can begin using remote search service, you must create three new content sources, one for the Web Content Manager, one for your portal site, and one for JCR search.

Before you begin

Create the new search collection for the remote search service. For detailed instructions about how to create new search collections, see *Creating new search collections*.

Important: The steps that you used to create the JCR search collection in *Setting up a JCR search collection* also included instructions about how to create the new JCR content source. If you created the JCR content source immediately after creating the JCR search collection, you do not need to create the JCR content source again.

Procedure

1. If you did not manually create a new JCR content source when you created the JCR search collection, create the JCR content source now. For detailed instructions about how to create JCR content sources, see *Setting up a JCR search collection*.
2. Manually create the new content source for your Portal site by completing the following steps:
   a. Click the name of the search collection that you created in *Creating new search collections*.
   b. Click **New Content Source**.
   c. In the **Content source type** field, specify **Portal site**. The portal URL is completed by default.

Note: If you are using a webserver, change the portal URL to specify the webserver host name and port.

d. Click the **Security** tab.

e. Specify your portal user name and password.

f. Specify the host name.

g. In the Define security realm pane, click **Create**. The security realm displays in the Security realms pane.

h. In the Manage Search pane, click **Create**. If the content source was created successfully, the following message displays: `EJPJB0025I: Content source source_name in collection collection_name is OK.`

i. Start the crawler to verify that the content source is working.

3. Manually create the new content source for your Web Content Manager site by completing the following steps:

a. Click the name of the search collection that you created in *Creating new search collections*.

b. Click **New Content Source**.

c. In the **Content source type** field, specify **WCM site**.

d. In the **Collect documents linked from this URL** field, specify the following URL: `http://server_name:port_number/wps/seedlist/myserver?SeedlistId=&Source=com.ibm.workplace.wcm.plugins.seedlist.retriever.WCMRetrieverFactory&Actio`

Note: If you are using a webserver, in the **Collect documents linked from this URL** field, specify the webserver host name and port.

e. Click the **Security** tab.

f. Specify your portal user name and password.

g. Specify the host name.

h. In the Define security realm pane, click **Create**. The security realm displays in the Security realms pane.

i. In the Manage Search pane, click **Create**. If the content source was created successfully, the following message displays: `EJPJB0025I: Content source source_name in collection collection_name is OK.`

j. Start the crawler to verify that the content source is working.

Configuring HTTP for the seedlist servlet

Learn how to configure HTTP for the seedlist servlet. The seedlist servlet requires HTTPs by default. Therefore, when you access the servlet through HTTP, WebSphere Application Server redirects you to HTTPs.

About this task

This is an optional step that can be completed while installing and configuring remote search service.

Procedure

1. On the portal server, open the following file with an editor: `PortalServer_root/search/wp.search.servlets/seedlist/servletEAR/installableApps/wp.search.seedlist.ear/wp.search.servlets.seedlist.war/WEB-INF/web.xml`

Cluster note: In a clustered environment, complete this step on the primary node and all secondary nodes.

2. Update the following code:

   Cluster note: In a clustered environment, complete this step on the primary node and all secondary nodes.

   ```
   <user-data-constraint>
        <transport-guarantee>CONFIDENTIAL</transport-guarantee>
   </user-data-constraint>
   ```

   Replace it by the following code:

   ```
   <user-data-constraint>
        <transport-guarantee>NONE</transport-guarantee>
   </user-data-constraint>
   ```

3. Save the file.

   Cluster note: In a clustered environment, complete this step on the primary node and all secondary nodes.

4. Run the following portal ConfigEngine script:

   Cluster note: In a clustered environment, complete this step on the primary node only.

   ```
   ./ConfigEngine.sh action-update-ear-wp.search.servlets/seedlist/servletEAR
   ```

5. Restart all servers in your configuration for your updates to take effect.

What to do next

Remove any local search collections that are defined in WebSphere Portal Express and create new search collections by using the remote search server. For more information about deleting and creating search collections, see the following topics:

- If your system is operating in a portal farm, continue on to *Configuring search in a portal farm*.
- If your system is operating in a clustered environment, continue on to *Configuring search in a cluster*.

Configuring search in a cluster

IBM WebSphere Portal Express provides two distinct search capabilities. You can use both types of search capabilities in a clustered environment.

"Configuring Portal Search in a cluster"
To support Portal Search in a cluster, you must install and configure a remote search service on an IBM WebSphere Application Server node that is not part of the cluster.

"Configuring JCR search in a cluster" on page 663
To enable search in a cluster for content that is stored in the JCR database, you must configure each server in the cluster to access a directory. JCR-based content includes content that is created with Web Content Manager or Personalization.

Configuring Portal Search in a cluster

To support Portal Search in a cluster, you must install and configure a remote search service on an IBM WebSphere Application Server node that is not part of the cluster.

Procedure

1. Install the remote search service. For more information on installing remote search service, see *Remote search service* in the related links section.

2. Configure the remote search service. For more information on configuring the remote search service, see *Configuring a remote search service* in the related links section.

Related concepts:

"Remote search service" on page 645
You can configure the search portlets for local operation, or you can configure them for remote search service. Depending on your configuration, remote search service might have performance benefits by offloading and balancing system load.

Related tasks:

"Configuring a remote search service" on page 657
Configure a remote search service for Portal Search.

Configuring JCR search in a cluster

To enable search in a cluster for content that is stored in the JCR database, you must configure each server in the cluster to access a directory. JCR-based content includes content that is created with Web Content Manager or Personalization.

Before you begin

Set up remote search service on the primary node of the cluster. For more information about setting up remote search service, see *Configuring a remote search service* in the related links.

Procedure

1. Log in to the WebSphere Integrated Solutions Console of the deployment manager.
2. Select **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. In the Resource environment providers page, make the appropriate selection to update the custom properties for all of the servers in the cluster. Choose one of the following two options:
   - Select the appropriate cluster from the Scopes list.
   - Clear the **Show Scope** check box and select **Browse Clusters** to specify the portal cluster.
4. Select **JCR ConfigService PortalContent** > **Custom properties**.
5. Change the value of the **jcr.textsearch.indexdirectory** property to point to a directory on the remote search server. For example, **jcr.textsearch.indexdirectory**=C:/JCR.
6. Change the **jcr.textsearch.PSE.type** property to EJB.
7. Change the **jcr.textsearch.EJB.IIOP.URL** property to the URL of the naming service that is used to access the WebScanner EJB. For example, iiop://localhost:2811.
8. Change the **jcr.textsearch.EJB.EJBName** property to the name of the WebScanner EJB. For example, ejb/com/ibm/hrl/portlets/WsPse/WebScannerLiteEJBHome.
9. Change the **jcr.textsearch.enabled** value to true.
10. Save your changes.
11. Restart your servers.
12. Required: In a cluster, you must drop the JCRCollections in the default search service and then re-create them in a remote search service, otherwise you receive display errors in your search. Complete the following steps to delete the JCRCollections from the Manage Search portlet:

a. Log on to WebSphere Portal Express as an administrator.
b. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
c. Click **Search Collections**.
d. Click the **Delete Collection** icon for the JCRCollection1 search collection.
e. Click **OK**.
f. Restart the WebSphere Portal Server.
g. Go to the Manage Search portlet and confirm that the JCRCollection1 search collection was deleted.
h. Manually create a JCR search collection called `JCRCollection1`. For more information about creating the JCR search collection, see *Setting up a JCR search collection* in the related links.

Related tasks:

"Setting up a JCR search collection" on page 698
A JCR search collection is a special purpose search collection that is used by WebSphere Portal applications. It is not designed to be used alongside user-defined search collections. A JCR search collection requires a special setup. This setup includes the creation of a new content source for the search collection. Under normal circumstances, you do not need to re-create the JCR search collection. However, in rare cases you might need to re-create it, for example if you deleted the default JCR search collection.

"Configuring a remote search service" on page 657
Configure a remote search service for Portal Search.

Related reference:

"JCR search service configuration parameters" on page 680
The following search service configuration parameters can be modified to enable and configure searching for content that is stored in the JCR database. These JCR search service configuration parameters can be modified by accessing the **JCR ConfigService PortalContent** resource environment provider.

Configuring search in a portal farm

IBM WebSphere Portal Express provides two distinct search capabilities. You can use both types of search capabilities in a portal farm environment.

"Configuring a remote Search in a portal farm"
To support Search in a portal farm, you must install and configure a remote search service. Install and configure it on an IBM WebSphere Application Server instance that is not part of the farm.

"Configuring JCR search in a portal farm" on page 665
JCR-based content includes content that is created with Web Content Manager or Personalization.

Configuring a remote Search in a portal farm

To support Search in a portal farm, you must install and configure a remote search service. Install and configure it on an IBM WebSphere Application Server instance that is not part of the farm.

About this task

Procedure

1. Install the remote search service. Go to the installation topics in "Remote search service" on page 645 for specific steps.

The service is on a remote WebSphere Application Server instance that is not part of the portal farm. You can provide the remote search service either as an EJB or as a web service with SOAP. Deploy the appropriate EJB or SOAP EAR file on the remote WebSphere Application Server instance. For details, refer to the WebSphere Application Server documentation.

2. Configure the search portlets for remote search service so that they access the remote server.
3. Complete the following steps to ensure that the Seedlist Servlet is started:
   a. Log on to the WebSphere Integrated Solutions Console on the remote server.
   b. Go to **Applications** > **Application Types** > **WebSphere enterprise applications**.
   c. Search for the **Seedlist Servlet**.
   d. Ensure that the **Application Status** is set to the arrow. If it is not started, click **Start**.
   e. Save your changes and then log out of the WebSphere Integrated Solutions Console.

What to do next

Notes:

1. You must configure the default location for search collections to a directory on the remote server that has write access.
2. The portal site default search collection is created the first time when an administrator selects the Manage Search portlet. If this occurred *before* you configure the remote search portlet, the default portal site search collection is only available on the primary farm instance. However, it is not available on the remote server. In this case, you must re-create the portal site collection to make it available for search on all instances of the farm.

Configuring JCR search in a portal farm

JCR-based content includes content that is created with Web Content Manager or Personalization.

About this task

Note: If you create content in a portal farm with the **Authoring** portlet, extra configuration steps are required. The steps enable content that is created by these content features to be searchable in a farm.

Procedure

1. Log in to WebSphere Integrated Solutions Console.

   Cluster note: In a clustered environment, log in to the Deployment Manager WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers** > **JCR ConfigService PortalContent** > **Custom properties**.
3. Change the value of the **jcr.textsearch.enabled** property to `false`.
4. Change the value of the **jcr.textsearch.indexdirectory** property to the index directory on the remote search server.

   For example, `jcr.textsearch.indexdirectory=\\\\your_server\\your_remotesearch\\jcr\\search`. You can specify the directory value in the following format:

Universal Naming Convention (UNC) format

`\\\\your_server\\your_remotesearch\\jcr\\search`

Example: `\\\\hostname.example.com\\share\\jcr\\search`

5. Based on the configuration of your remote search service, change the **jcr.textsearch.PSE.type** property to either `EJB` or `SOAP`; then choose the appropriate extra steps:

*Table 78. The steps depend on the value for the **jcr.textsearch.PSE.type** property.*

Value	Extra steps
EJB	Complete the following steps if you have an EJB service: 1. Change the **jcr.textsearch.EJB.IIOP.URL** property to the URL of the naming service that is used to access the WebScanner EJB; for example `iiop://localhost:2811`. 2. Change the **jcr.textsearch.EJB.EJBName** property to the name of the WebScanner EJB; for example `ejb/com/ibm/hrl/portlets/WsPse/WebScannerLiteEJBHome`.
SOAP	If you have a SOAP service, change the **jcr.textsearch.SOAP.url** property to the SOAP URL of the WebScanner for the search service.

6. Save your changes.
7. Restart your server.
8. Required: Complete the following steps to delete the default search collections from the Manage Search portlet:
   a. Log on to WebSphere Portal Express as an administrator.
   b. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
   c. Click **Search Collections**.
   d. Click the **Delete Collection** icon for the **Portal Content** search collection.
   e. Click **OK**.
   f. Restart the WebSphere_Portal server.
   g. Go to the Manage Search portlet and confirm that the **Portal Content** search collection was deleted.
   h. Manually create a JCR collection called `JCRCollection1`; refer to Setting up JCR Search Collection for information.

Related reference:

"JCR search service configuration parameters" on page 680
The following search service configuration parameters can be modified to enable and configure searching for content that is stored in the JCR database. These JCR search service configuration parameters can be modified by accessing the **JCR ConfigService PortalContent** resource environment provider.

Configuring search collections for a virtual portal

Configuring JCR search collections for a virtual portal might require additional administration, depending on how you set up the virtual portal.

When you create a virtual portal, the creation of the JCR search collection depends on whether you create the virtual portal with or without content:

- If you create the virtual portal with content, the portal creates the JCR collection for the virtual portal by default.
- If you create only the virtual portal and add no content to it, the portal creates no JCR collection with it. It gets created only when content is added to the virtual portal.

You can view the URL of the JCR search collection in the search administration portlet Manage Search of the virtual portal. The URL looks as follows:

```
 http://host_name:port_number/wps/seedlist/myserver/
hello?Action=GetDocuments&Format=ATOM&Locale=en_US&Range=100
&Source=com.ibm.lotus.search.plugins.seedlist.retriever.jcr.JCRRetrieverFactory&Start=0&
```

Where *wsid* is the actual workspace ID of the virtual portal. The workspace ID is the identifier of the workspace in which the content item is created, stored, and maintained. For example, if the workspace ID of the virtual portal is 10, then the URL looks as follows:

```
http://host_name:port_number/wps/seedlist/myserver/
hello?Action=GetDocuments&Format=ATOM&Locale=en_US&Range=100
&Source=com.ibm.lotus.search.plugins.seedlist.retriever.jcr.JCRRetrieverFactory&Start=0&
```

If the JCR search collection was deleted, or if you added content to an originally empty virtual portal and the JCR search collection was not automatically created, complete the following steps:

- If you are using a virtual portal, go to the Security tab of the content source to verify that the workspace ID of the virtual portal is correct.
- If the JCR search collection was deleted, run the **`ConfigEngine`** task `create-textsearch-collections` to re-create the JCR search collection.

If neither of the preceding options succeed in creating the JCR search collection, manually set up the JCR search collection.

To view the steps to manually set up a JCR search collection, see *Setting up a JCR search collection* in the related links.

Related tasks:

"Setting up a JCR search collection" on page 698
A JCR search collection is a special purpose search collection that is used by WebSphere Portal applications. It is not designed to be used alongside user-defined search collections. A JCR search collection requires a special setup. This setup includes the creation of a new content source for the search collection. Under normal circumstances, you do not need to re-create the JCR search collection. However, in rare cases you might need to re-create it, for example if you deleted the default JCR search collection.


Portal Search

Use Portal Search to facilitate indexing content sources and searching for information. You can administer search services, search collections, and search scopes, as well as enhance the search experience of your portal site with the portal search portlets.

"Portlets for working with Search" on page 668
Get an overview of the portal search portlets.

"Administering Portal Search" on page 671
You can administer and configure many details for Portal Search.

"Hints and tips for using Portal Search" on page 714
View some useful tips for using Portal Search.

"Setting limits on searches for users and groups" on page 722
Searching for users or groups is a time consuming task. A search may time out or return more results than the system can handle or the user may expect. To prevent this behavior, you can set limits on searches for users or groups.

"LDAP search filter expressions" on page 723
The rules for rule-based user groups are based on the LDAP search filter syntax.

"Creating the portal site search collection can fail" on page 721
Creating the portal site search collection can fail due to a file path length restriction.

"Portal Search trace and log files" on page 725
Portal Search provides logging and tracing so that you can get additional information for resolving possible problems.

Related reference:

"Portal Search trace and log files" on page 725
Portal Search provides logging and tracing so that you can get additional information for resolving possible problems.

Related information:

 Technotes for portal search

Portlets for working with Search

Get an overview of the portal search portlets.

The portal search portlets are briefly described in the following sections:

- Overview of Portal Search portlets
- Search administration portlet **Manage Search**
- **Search Center** portlet for search by users

For more details about enhancements to the search portlets for IBM WebSphere Portal Express Version 8.5, refer to Portal Search.

Overview of Portal Search portlets

The following portlets are all installed and deployed as part of the default portal installation:

- For use by administrators: Manage Search
- For users: Search Center

If you want to use the other search-related portlets in your portal, you have to install them and configure them according to your requirements. Here is an overview of the Portal Search administration and user portlets:

Table 79. Administration portlet and installation status

Administration portlet	Installation status
Manage Search	Installed and deployed

Table 80. User portlets and installation status

User portlets	Installation status
Search Center	Installed and deployed
Suggested Links	Installed and deployed
External Search Results	Installed and deployed

You can install additional copies of the Suggested Links and External Search Results portlets and apply special configurations. For details refer to the topics about adding and configuring these portlets and about customizing the Search Center.

Portal Search portlets are not compatible with WSRP

The Portal Search portlets cannot be provided as WSRP services, as some additional and more advanced WebSphere Portal Express concepts and features are not yet reflected by the current WSRP standard. This includes the Portal Search portlets Manage Search and the Search Center.

Search administration portlet Manage Search

Manage Search is the Portal Search administration portlet. It has three main sections:

- **Search Services**: Use this section to manage search services.
- **Search Collections**: Use this section to manage search collections and their content sources.
- **Search Scopes**: Use this section to manage search scopes and custom links to web search locations.

You can use the Manage Search portlet to perform the following administrative tasks:

1. Manage search services. You can use the search services that are provided with portal, or you can add one or more search services to Portal Search. For example, this is required in scenarios where you set up remote search or your portal is set up in a cluster.
2. Manage search collections. You can define one or more search collections for a search service.
3. Manage the content sources of search collections. You can combine content sources of different types in one search collection.
4. Perform administrative tasks required as preparatory steps for the search feature, such as indexing.
5. Manage search scopes. You can define search scopes to limit search results to specific content locations and specific document types. This enables users to target their searches better.
6. Manage custom links. You can add custom links with web link shortcuts to search locations. This enables users to do direct searches to popular web search engines, such as Google or Yahoo!
7. Search and browse a search collection. Manage Search provides a user interface with administrative tasks for search collections. For example, you can edit documents, or you can select documents that you do not want to make available to users for search and delete them.

For detailed information about how to use the Manage Search portlet, refer to the portlet help. Depending on your requirements, you might have to perform other administrative tasks for Portal Search. For example, you can prepare security for Portal Search, or configure remote search or search in a portal cluster. For more information about the Manage Search portlet and other administrative tasks related to search, refer to the other topics in the Portal Search section.

Search Center portlet for search by users

The Search Center portlet provides a central starting point to all searchable content sources made available to the Portal. Users can use the Search Center to search documents and content. The default search scope is **All Sources**. This searches through all search collections that are available to the user by the access permission rights. You can allow more focused searches by providing additional search scopes. Refer to Search scopes for more information.

The Search Center portlet is installed as part of the default portal installation. Note that it is deployed and placed on a hidden portal page. The portal takes users to the Search Center when they enter a search using the portal Search Box.

Starting with Version 8 of WebSphere Portal Express the Search and Browse portlet provided with earlier portal versions is no longer available. The Search Center has been enhanced with advanced search options previously available in the Search and Browse portlet. Categorization and taxonomy are no longer available.

Note: By default both the Search Center portlet and the Search box are available only to authenticated users, that is for users who have logged on with a user ID and password. Anonymous users, that is users who access the portal without logging in, cannot use the Search Center or the search box. For more information about how to use the search portlets on anonymous pages refer to Enabling anonymous users to search public pages of your portal.

For more details about the Search Center portlet, refer to the following information:

- For more information about how to work with the Search Center, refer to the portlet help.
- For hints about using the Search Center for remote search on web sites with different languages, refer to Hints and tips for using Portal Search.

Related tasks:

"Administering Portal Search" on page 671
You can administer and configure many details for Portal Search.

"Managing search scopes and custom links" on page 687
Get an overview of search scopes and custom links and how you can manage them.

"Enabling anonymous users to search public pages of your portal" on page 641
You can enable anonymous users (sometimes also called unauthenticated users) to search public pages of your portal by using the portal Search Center portlet. Search by anonymous users works only on public pages of your portal, as the users are not logged in to your portal.

"Customizing the Search Center" on page 687
Customize the Search Center by adding, removing and configuring additional portlets, such as External Search Results or Recommended Links. The external search results portlet displays search results from third-party external search engines such as Yahoo and Google. Using the recommended links portlet, display search results from a collection of predefined links with predefined keywords. You can also configure the All Sources scope or replace it with a customized scope.

"Configuring the Suggested Links view" on page 692
Customize the display of search results to show users the preferred or recommended results and associated links.

"Adding and configuring the External Search Results portlet" on page 688
Configure a portlet that retrieves and displays search results from third-party search engines, then add the External Search Results portlet to the Search Center.

Related reference:

"Hints and tips for using Portal Search" on page 714
View some useful tips for using Portal Search.

Administering Portal Search

You can administer and configure many details for Portal Search.

About this task

Note: Before you start administering Portal Search, review the topic about Planning and Preparing for Portal Search.

Manage Search is the Portal Search administration portlet. It has three main sections:

- Search Services: Use this section to manage search services.
- Search Collections: Use this section to manage search collections and their content sources.
- Search Scopes: Use this section to manage search scopes and custom links to Web search locations.

In order to enable Portal Search and make documents available for search by users, you perform administrative tasks such as the following:

Procedure

1. Create search services. You can use the default search services provided with portal, or you can create additional search services, for example, for setting up remote search or search in a portal cluster.
2. Configure Portal Search. You do this by configuring the search service.
3. Configure the search portlets for various environments and requirements, for example, for local or remote search service, or for search on anonymous pages.
4. Manage Portal Search for users:
   a. Create a search collection and define its properties, thereby allowing for fast and efficient searches.
   b. Create one or more content sources for that collection and have them crawled.

   For more details about these tasks refer to the topic about Setting up search collections.

Migrating search collections: Starting with WebSphere Portal Express Version 6.1 the syntax of the seedlist URL has changed to seedlist format 1.0. Older search collections created before portal V 6.1 use seedlist format 0.9 and cannot be reused or migrated to the new format. Starting with WebSphere Portal Express Version 7 JCR was added as 1.0 seedlist format. Be sure that you index all content again before using seedlist format 1.0.

Results

What to do next

Portal Search is preconfigured with a search service, a portal site search collection, and scopes. You can add and remove search services, collections, and scopes.

You can perform most of the configuration tasks for Portal Search by using the administration portlet **Manage Search**. By alternative, you can also administer Portal Search by using the WebSphere Integrated Solutions Console and resource providers in XML format.

All search collections are available by the **All Sources** selection option of the Search Center portlet.

"Managing search services" on page 673
Get an overview of how you manage the portal search services. This task includes creating a new search service or editing an existing search service.

"Search service configuration parameters" on page 674
Learn about the portal search service parameters and possible values.

"Configuring the default location for search collections" on page 682
You can modify the default directory location under which search collections are created on a per search service basis. View some related information.

"Configuring the Search Center portlet" on page 684
Get an overview of how you configure the Search Center.

"Replacing the search administrator user ID" on page 686
If you changed the portal administrator user ID or password, you need to update the search administrator user ID to match the same values.

"Customizing the Search Center" on page 687
Customize the Search Center by adding, removing and configuring additional portlets, such as External Search Results or Recommended Links. The external search results portlet displays search results from third-party external search engines such as Yahoo and Google. Using the recommended links portlet, display search results from a collection of predefined links with predefined keywords. You can also configure the All Sources scope or replace it with a customized scope.

"Using the WebSphere Integrated Solutions Console to administer Portal Search" on page 694
You can administer Portal Search by using the WebSphere Integrated Solutions Console and using resource providers in XML format.

"Setting up search collections" on page 695
View information on setting up search collections for search by users. This also includes creating content sources and managing search scopes and custom links.

"Searching and crawling portal and other sites" on page 710
Configure your local portal site, and crawl remote portal sites, so that they are searchable by users. Run crawlers against other, external Web sites to make them searchable by local portal users.

Related concepts:

"Setting up search collections" on page 695
View information on setting up search collections for search by users. This also includes creating content sources and managing search scopes and custom links.

Related information:

Using the search seedlist 1.0 format

Managing search services

Get an overview of how you manage the portal search services. This task includes creating a new search service or editing an existing search service.

About this task

To create or edit a portal search service, proceed as follows:

Procedure

1. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
2. Click **Search Services**. Manage Search displays the Search Services panel.
3. If you want to create a new search service, click the **New Search Service** button. If you want to modify an existing search service, click the **Edit** icon for that search service. Manage Search lists the parameter key and value pairs for the search service.

Results

Set or edit the parameter values depending on your requirements and configuration.

Notes for configuring a search service:

1. Unless otherwise stated, the values that you set for parameters of a portal search service apply to that search service and all its collections. They do not affect other search services of the portal or their search collections.
2. Unless otherwise stated, changing the value of a parameter changes both the existing search collections and newly created search collections. Some parameters affect only newly created search collections. These parameters cannot be updated for existing search collections.
3. The search administration portlet Manage Search lists the Default Portal Search Service and its collection Portal Content or other collections in the default portal language and not in the language that the user selected as preferred language for the portal or set in the browser. For example, if the portal default language is set to English and the user selected German as the preferred portal language or set the browser language to German, the Default Portal Search Service and its collections show in English.
4. SOAP support for remote search services was deprecated with WebSphere Portal Express Version 8.0. EJB is still supported.
5. If you delete a search service, the portal does not delete the search collections that are related to this search service. Delete the search collections by using the Manage Search administration portlet. If you delete the default search service, it is re-created new when you restart the portal.

What to do next

For more detailed steps about how to manage search services refer to the Manage Search portlet help. For a list of the search service parameters and possible values, refer to *Search service configuration parameters*.

If you want to create a search service for remote Portal Search, refer to *Configuring a remote search service*.

Related tasks:

"Configuring a remote search service" on page 657
Configure a remote search service for Portal Search.

Related reference:

"Search service configuration parameters"
Learn about the portal search service parameters and possible values.

Search service configuration parameters

Learn about the portal search service parameters and possible values.

To configure a portal search service, use the following parameters. For details about how to set the values for these parameters refer to *Managing search services* or the Manage Search administration portlet help.

Notes for configuring a search service:

1. Unless otherwise stated, the values that you set for parameters of a portal search service apply to that search service and all its collections. They do not affect other search services of the portal or their search collections.
2. Unless otherwise stated, changing the value of a parameter changes both the existing search collections and newly created search collections. Some parameters affect only newly created search collections. These parameters cannot be updated for existing search collections.
3. The search administration portlet Manage Search lists the Default Portal Search Service and its collection Portal Content or other collections in the default portal language and not in the language that the user selected as preferred language for the portal or set in the browser. For example, if the portal default language is set to English and the user selected German as the preferred portal language or set the browser language to German, the Default Portal Search Service and its collections show in English.
4. SOAP support for remote search services was deprecated with WebSphere Portal Express Version 8.0. EJB is still supported.
5. If you delete a search service, the portal does not delete the search collections that are related to this search service. Delete the search collections by using the Manage Search administration portlet. If you delete the default search service, it is re-created new when you restart the portal.

Notes related to the search service configuration parameter list:

1. The parameter list in both the search services pane of the Manage Search portlet and in the following information shows several parameters that end with the suffix `_EXAMPLE`. These example parameters are not used by the portal. They serve as an example for the same parameter without the suffix `_EXAMPLE`. They give an example value that you might use. Deleting these parameters or modifying their value has no effect.

2. If you want to set a parameter that is listed here, but not in the portlet, add it. To add a parameter, type the parameter and the value in the entry fields **Parameter key:** and **New parameter value:** and click the **Add Parameter** button.
3. In the following list, the abbreviation **pse** in parameters or values stands for Portal Search Engine.
4. The following list is arranged in alphabetical order. Parameters might be listed in a different order in the portlet.

boostingSettings

Use this parameter to specify which metadata fields are given extra weight in an overall rank score during a search. You can also specify how much the selected metadata fields contribute to relevance circulation when you run a search. Specify the following values:

fieldBoost

This value defines which metadata fields have extra weight when search results are returned, and how much extra weight is given to the specified fields. Provide the following attributes:

field The relevant string-based metadata field that you would like search to focus on. Some common or default field values are `title`, `description`, and `keywords`.

boost The relative amount of extra weight added to the rank score. This value should be set between a range of 1.0 and 10.0. However, the suggested range of the attribute is between 1.0 and 3.0.

phraseBoost

This value is a nonessential variable that focuses search results on specified languages, for example, English.

boostingSetting_Example

The following is an example value for the parameter **boostingSettings**:

```
phraseBoost: {Enabled:true},
fieldBoost: {field:title, boost:3.0},
            {field:description, boost:3.0},
            {field:keywords, boost:2.0}
```

CLEAN_UP_TIME_OF_DAY_HOURS

Time of day at which the portal runs the maintenance process for search collections to remove outdated files and broken links. Possible values are positive integers 0 - 24 for the full hours of the day. The default value is `0` , which runs the cleanup at midnight.

Note: If you modify the value for this parameter, the new value is applied only to newly created collections of the search service. You cannot update this parameter for existing search collections.

DefaultCollectionsDirectory

You can use this parameter to specify the default directory for search collections. If you use Portal Search locally, this parameter is optional. If you specify no value for this parameter, the default collection directory is *wp_profile_root*`/PortalServer/collections`. If you set up a remote search service, this parameter is mandatory. For details about setting this parameter, refer to "Configuring the default location for search collections" on page 682.

DEFAULT_SEARCH_OPERATOR
Use this parameter to specify how the Portal search engine responds to search queries with two or more terms. The default value is `or`. Only one search term must be in the document in order for that document to be displayed in the search results list. Change this value to `and` to retrieve only those documents that contain all of the search terms that are listed in the query.

Note: After you change this parameter, you must restart the Portal server and remote search service.

CONFIG_FOLDER_PATH
Use this parameter to determine where the configuration data for search collections is stored. The default is *wp_profile_root*`/CollectionsConfig`.

EJB If you set up a remote search service by using EJB, use this parameter to specify the EJB name in JNDI. An example value is `ejb/com/ibm/hrl/portlets/WsPse/WebScannerLiteEJBHome` .

If you set this parameter, you also need to set the `IIOP_URL` parameter.

EJB_Example
This parameter is an example that gives an example value for the parameter `EJB` . The example value is `ejb/com/ibm/hrl/portlets/WsPse/WebScannerLiteEJBHome` .

ExternalSecurityResolverUrl
Use this parameter to configure the Portal Search service with the information about an external security resolver. This parameter is required for security filtering of Connections resources to function properly. An example value of the resolver URL is `https://host:port/`*`ConnectionsResourceId`*`/seedlist/authverify/getACLTokens` where *ConnectionsResourceID* is any Connections resource identifier.

HTTP_MAX_BODY_SIZE_MB
Use this parameter to limit how much content is fetched during a crawl from application files, such as PDF or Microsoft Word. The specified unit is MB. The default value is `20` MB. If a file exceeds the specified limit, the document is truncated, and Portal Search indexes the fetched portion as is possible. However, indexing might fail on truncated documents; in this case the document is not listed under search results at all.

Notes:

1. If you modify the value for this parameter, the new value is applied only to newly created collections of the search service. You cannot update this parameter for existing search collections.
2. Document Conversion Services might not be able to convert the content of truncated application files. If Document Conversion Services fails to convert a truncated application file, it logs an error to the `SystemErr.log` file. If tracing is enabled for the portal, Portal Search logs a warning message to the portal log file.

HTTP_MAX_SEEDLIST_SIZE_MB
This parameter limits how much portal content is fetched during a crawl from your own portal site. It determines the amount of space that is reserved for listing portal site resources or managed web content resources. The specified unit is MB. The default value is `4` MB. If a crawl exceeds the limit set for this parameter, the crawl fails, and Portal Search logs an error

message. In this case, or if returned search results do not represent to complete extent of your portal site resources, increase this value.

Note: If you modify the value for this parameter, the new value is applied only to newly created collections of the search service. You cannot update this parameter for existing search collections.

HTTP_NON_APPL_MAX_BODY_SIZE_MB
Use this parameter to limit how much content of each HTML page is fetched from websites of collections that belong to this search service. The specified unit is MB. The default value is `0.2` MB. This value means that the amount of content that is sent for indexing is always the first 0.2 MB of text.

Note: If you modify the value for this parameter, the new value is applied only to newly created collections of the search service. You cannot update this parameter for existing search collections.

IIOP_URL
If you set up a remote search service by using EJB, use this parameter to specify the IIOP URL. An example value is `iiop://localhost:2811`.

IIOP_URL_Example
This is an example that gives an example value for the parameter `IIOP_URL`. The example value is `iiop://localhost:2811`.

PSE_TYPE
Use this parameter to specify the type of search service. Possible values are `localhost`, `ejb`, and `soap`. The default value is `localhost` for local search service.

If you use Portal Search locally, this parameter is optional.

If you set up a remote search, this parameter is mandatory. In this case specify the type of remote service that you use, EJB or SOAP. If you specify `ejb` here, you also need to specify the values for the parameters `EJB` and `IIOP_URL`. If you specify `soap` here, you also need to specify the values for the parameter `SOAP_URL`.

SEARCH_SECURITY_MODE
This parameter defines access control enforcement during search. Three filter modes are supported. Specify one of the following values, depending on the filter mode that you want to use:

SECURITY_MODE_PREFILTER
Specify this value to use pre-filtering mode. Pre-filtering provides the fastest filtering, as it is performed in the search index level. An extra advantage of this filtering mode is that remote secured content sources can be searched from portal. However, as it is based on search index only, the search result list can be temporarily inconsistent with user access rights if these access rights were changed after the last crawl:

- If users who had their access rights restricted after the last crawl, they might get search results listed to which they had access before, but to which they no longer have access. When these users click such a link in the search result list, they cannot access the document.

- If a user was given access rights on documents after the last crawl, the user will not get these documents listed among the search results until after the next crawl.

Note: If the search service contains Portal content (a collection that contains a content source of type **`Portal site`**), then this security mode is invalid and must not be used.

SECURITY_MODE_POSTFILTER
Post-filtering

Specify this value to use post-filtering mode. Post-filtering provides the safest but costly filtering approach. It checks access permission in real time for each returned search result against Portal Access Control. As a result you can use it only for local content sources. This was the only filtering mode available before portal V 7.0.

SECURITY_MODE_PRE_POST_FILTER
Pre-post-filtering

Specify this value to use pre-post-filtering mode. This is the default. Pre-post-filtering combines the two filter modes previously mentioned. It provides a balanced method for enforce access control. It filters most irrelevant documents at the pre-filtering phase based on the search index. This results in fewer rejections in the post-filtering phase. As it still uses post-filtering, you can apply it only for local content sources. As it uses pre-filtering, search result lists might be temporarily inconsistent with users' access rights until after the next crawl.

SEEDLIST_PAGE_TIMEOUT
Use this parameter to increase the timeout for fetching the seedlist page. The specified unit for the value is seconds. The default value is 150 sec. This value means that the portal search attempts to fetch the seedlist main URL for 150 seconds.

Note: If you modify the value for this parameter, the new value is applied only to newly created collections of the search service. You cannot update this parameter for existing search collections.

SOAP_URL
If you set up a remote search service by using SOAP, use this parameter to specify the SOAP URL. An example value is `http://localhost:10000/WebScannerSOAP/servlet/rpcrouter`.

SOAP_URL_Example
This is an example that gives an example value for the parameter `SOAP_URL`. The example value is `http://localhost:10000/WebScannerSOAP/servlet/rpcrouter`.

CF05 **dateFieldPattern**
By default, portal search does not know whether a field contains a date. Use this parameter to enable search for documents by date. A regular expression is used to check whether a field must be handled as a date field or not. The default pattern is `".*date$"`, and matches all fields that end with the word “date”.

Note: After you change this parameter, you must restart the Portal server and remote search service.

CF05 **dateFormat**

Specify the format that is used for date queries. The default is `yyyy-MM-dd`. You can specify a different format by using the Java date syntax with the exception that spaces cannot be used, since that would break the date range queries. Make sure to communicate any changes in format to search users. To verify that the format is supported, you can enable tracing for `com.ibm.lotus.search.index.lucene.search.PseSiapiQueryParser=all` and then perform a series of searches. Do not forget to disable the trace after you verify that the format is supported.

Note: After you change this parameter, you must restart the Portal server and remote search service.

CF05 **dateTimeFormat**

Specify the format that is used for date queries with a time part. The default is `yyyy-MM-dd,hh:mm`. You can specify a different format by using the Java date syntax with the exception that spaces cannot be used, since that would break the date range queries. Make sure to communicate any changes in format to search users. The typical letters that are used in the format are:

yyyy Specifies the year.

MM Specifies the numerical month in the year. For instance, the month of December would be represented by 12.

dd Specifies the numerical day in the month.

hh Specifies the hour in the day. By default, the hour is specified in the 24-hour format. For instance, the number 18 specifies 6 PM. However, you can specify a custom format that uses the 12-hour time format instead.

mm Specifies the minute in the hour.

ss Specifies the second in the minute.

Z Specifies the time zone. For example, `-0800`.

To verify that the format is supported, you can enable tracing for `com.ibm.lotus.search.index.lucene.search.PseSiapiQueryParser=all` and then perform a series of searches. Do not forget to disable the trace after you verify that the format is supported.

Note: After you change this parameter, you must restart the Portal server and remote search service.

CF05 **dateFormatLocale**

This parameter specifies the locale that is used when parsing a date. The portal default locale is used as the default value.

Note: After you change this parameter, you must restart the Portal server and remote search service.

The following parameters are reserved for internal use only. Do not change their values.

CONTENT_SOURCE_TYPE_FEATURE_NAME

This parameter is reserved for internal use only. Do not change its value. The default value is `ContentSourceType`.

CONTENT_SOURCE_TYPE_FEATURE_VAL_PORTAL
This parameter is reserved for internal use only. Do not change its value. The default value is `Portal`.

CONTENT_SOURCE_TYPE_FEATURE_VAL_WEB
This parameter is reserved for internal use only. Do not change its value. The default value is `Web`.

SecurityResolverId
This parameter is reserved for internal use only. Do not change its value. The default value is `com.ibm.lotus.search.plugins.provider.core.PortalSecurityResolverFactory`.

SetProperties
This parameter is reserved for internal use only. Do not change its value. Possible values are `on` or `off`. The default value is `on`.

startup
This parameter is reserved for internal use only. Do not change its value. The default value is `false`.

VALIDATE_COOKIE
This parameter is reserved for internal use only. Do not change its value. The default value is `123`.

WORK_MANAGER
You can use this parameter to specify the work manager. This parameter is reserved for internal use only. Do not change its value. The default value is `wps/searchIndexWM`.

WORK_MANAGER_DEPLOY
This is an example of the deployed `WORK_MANAGER` parameter. The example value is `wps/searchIndexWM`.

WORK_MANAGER_NATIVE
This is an example of the parameter `WORK_MANAGER` for native threads for debug purposes only. The example value is `force.hrl.work.manager.use.native.threads`.

WORK_MANAGER_NAME
This parameter specifies the JNDI name of the work manager that Portal Search uses.

"JCR search service configuration parameters"
The following search service configuration parameters can be modified to enable and configure searching for content that is stored in the JCR database. These JCR search service configuration parameters can be modified by accessing the **JCR ConfigService PortalContent** resource environment provider.

Related tasks:

"Managing search services" on page 673
Get an overview of how you manage the portal search services. This task includes creating a new search service or editing an existing search service.

"Configuring the default location for search collections" on page 682
You can modify the default directory location under which search collections are created on a per search service basis. View some related information.

JCR search service configuration parameters

The following search service configuration parameters can be modified to enable and configure searching for content that is stored in the JCR database. These JCR search service configuration parameters can be modified by accessing the **JCR ConfigService PortalContent** resource environment provider.

To access the JCR search service configuration parameters that are stored in the **JCR ConfigService PortalContent** resource environment provider, complete the following steps:

1. Log in to WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers** and select **JCR ConfigService PortalContent**.
3. In the Additional Properties section of the Configuration window, select **Custom properties**.

Important: There are many configuration parameters in the **JCR ConfigService PortalContent** resource environment provider that must not be modified. Do not modify any configuration parameters that are not referenced in the following list, unless you are directed to do so by IBM support.

jcr.textsearch.enabled

This parameter enables or disables text search. The default value is `true`. Set this value to `false` to disable the text search run time. This parameter is required.

jcr.textsearch.indexdirectory

This parameter specifies the directory where indexes are stored, for example,`/opt/IBM/WebSphere/wp_profile/PortalServer/jcr/searchIndexes`. This parameter is required if text search is enabled (**jcr.textsearch.enabled**=`true`).

jcr.textsearch.PSE.type

This parameter specifies whether the search service is a local search service or a remote search service. The default value is `localhost`. Specify one of the following values:

localhost
This value specifies the search service as a local search service.

EJB
This value specifies the search service as a remote search service.

Notes:

- SOAP support for remote search service was deprecated with WebSphere Portal Express Version 8.0.
- If you are using a remote search service, you must also specify the**jcr.textsearch.EJB.IIOP.URL** and**jcr.textsearch.EJB.EJBName** parameters.

jcr.textsearch.EJB.IIOP.URL

This parameter specifies the URL of the naming service that is used to access the WebScannerEJB, for example, `iiop://localhost:2811`. Specify this parameter if you are using a remote search service (**jcr.textsearch.PSE.type**=`EJB`).

jcr.textsearch.EJB.EJBName

This parameter specifies the name of the WebScanner EJB, for example, `ejb/com/ibm/hrl/portlets/WsPse/WebScannerLiteEJBHome`. Specify this parameter if you are using a remote search service (**jcr.textsearch.PSE.type**=`EJB`).

jcr.query.collation.db2.enabled

This parameter enables or disables collation support for the ordering of results in the JCR XPath queries. This parameter is for all DB2 platforms. The default value is `false`. Specify `true` to enable collation support.

Configuring the default location for search collections

You can modify the default directory location under which search collections are created on a per search service basis. View some related information.

About this task

The default directory location under which search collections are created is as follows:

- Linux: *wp_profile_root*`/PortalServer/collections`
- IBM i: *wp_profile_root*`/PortalServer/collections`
- Windows: *wp_profile_root*`\PortalServer\collections`

This default can be applied when you create a new search collection, depending on the value you specify in the entry field **Location of Collection** when you create a search collection:

- If you type a relative directory location of your choice, the location for the new search collection is combined from the default directory for search locations and the location that you type. Example: If you type `my_collection_location`, the new search collection is created under the directory *wp_profile_root*`/PortalServer/collections/my_collection_location`.
- If you want to create the search collection in a location that is different from the default search collection location, type the full directory location. The new search collection is created under the directory location that you specified.

You can customize this default directory location under which search collections are created. To complete this process, you set the directory location of your choice as the value for the parameter `DefaultCollectionsDirectory` for the search service. You can configure this parameter for each search service separately. The value that you set for this parameter is prefixed to the relative location that you specify when you are creating a search collection for that search service.

The specified value determines the default directory location for the search collections that you create. When you create a search collection and specify only a relative path (not a full path location), the default value that is set by this parameter and the relative path that you specify are combined. These values form the full directory path under which the search collection is created. If you do not specify a value for `DefaultCollectionsDirectory`, the default directory for search collections remains as set by the installation as described previously.

To configure a different default search collection location, proceed by the following steps:

Procedure

1. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
2. Click **Search Services**.
3. Click the **New Search Service** button. Or, if you want to customize the default collection location for an existing search service, click the **Edit** icon for that search service.

4. Set the value of your choice for the parameter `DefaultCollectionsDirectory` for the search service. The value that you set for this parameter is prefixed to the relative location that you specify when you are creating a search collection. For details about the value, refer to the previous description.
5. Click **OK** to save your updates. Manage Search returns to the Search Services panel.

   Note:

   a. The directory location that you specify can be on a different hard disk drive or workstation.
   b. If you change the value for the **DefaultCollectionsDirectory** parameter, make sure that you do all of the following steps:
      - Specify a full directory path, for example, `/root/our_search_collections`.
      - On the specified hard disk drive, create the directory that you specify as the value for the **DefaultCollectionsDirectory** parameter.
      - Manage the write access to the directory to enable creation of search collections in that directory.
   c. The value that is set for this parameter is used only if during the creation of a new search collection you specify a relative directory for the location of the new search collection. If you specify a full directory for the location of the search collection, the collection is created in that directory. In this case, the value that is set for the default directory under the search service has no effect. Instead, it is overwritten by the full directory location that you specify for the search collection.
   d. The initial default directory is created during the installation. The process has write access to this default directory.
   e. The default site collection that is part of the installation is created under the default directory. Creation of the site collection is started when you navigate to the Manage Search portlet. If you created the site collection by navigating to the Manage Search portlet before you changed the default directory for the collection location, you might want to relocate the site collection to your new default directory. For details about how to do this step, see the topic about Resetting the default search collection.
   f. If you set up a remote search service, for example for a cluster, then this parameter is mandatory. You must configure the default location for search collections to a directory on the remote server that has write access.
   g. The file path length for search collections is limited to 118 characters. If this limit is exceeded, the collection cannot be created. In this case specify a shorter value for the parameter **DefaultCollectionsDirectory**. For details about this process, read *Creating the portal site search collection fails*.

      This issue can occur particularly when the site collection is created under one of the following operating systems.
      - Linux

Related tasks:

"Resetting the default search collection" on page 711
Under certain circumstances, you might want to change the configuration of the portal site search collection. In this case, you must re-create the collection, as search collections cannot be modified.

Related reference:

"Creating the portal site search collection can fail" on page 721
Creating the portal site search collection can fail due to a file path length restriction.

"Hints and tips for using Portal Search" on page 714
View some useful tips for using Portal Search.

Configuring the Search Center portlet

Get an overview of how you configure the Search Center.

About this task

The Search Center is installed and ready to use in a default portal installation. It is available to authenticated users through the search box in the portal theme header action bar. Users can use the Search Center for basic searches.

If required, you can configure the Search Center for search by metadata that is contained within the meta elements and by search scopes. See the following topics for details.

If your users use external search services to perform searches in different languages, refer to "Using the Search Center with external search services with different languages" on page 714.

CF05 "Configuring search for multilingual sites"
By default, the Search Center uses the preferred language of the user to analyze search terms and to refine the search results. Search results in different languages are not displayed. You can enable users to search for terms in a language other than their preferred language.

"Configuring search scopes for the Search Center portlet" on page 685
The Search Center provides scopes for users to select different sources for the search. A search scope is a means of filtering the search request to a predefined set of content or content source.

"Configuring search by metadata for the Search Center portlet" on page 686
You can configure the Search Center so that users can refine their search by metadata.

Related tasks:

"Customizing the Search Center" on page 687
Customize the Search Center by adding, removing and configuring additional portlets, such as External Search Results or Recommended Links. The external search results portlet displays search results from third-party external search engines such as Yahoo and Google. Using the recommended links portlet, display search results from a collection of predefined links with predefined keywords. You can also configure the All Sources scope or replace it with a customized scope.

"Managing search scopes and custom links" on page 687
Get an overview of search scopes and custom links and how you can manage them.

Related reference:

"Hints and tips for using Portal Search" on page 714
View some useful tips for using Portal Search.

Configuring search for multilingual sites

By default, the Search Center uses the preferred language of the user to analyze search terms and to refine the search results. Search results in different languages are not displayed. You can enable users to search for terms in a language other than their preferred language.

Procedure

- Complete the following steps to configure the Search Center for use in a multilingual environment where users have different preferred languages and information is distributed in multilingual content sources:
  1. Click the **Administration menu** icon. Then, click **Portlet Management** > **Portlets**.
  2. Locate the Search Center portlet by either navigating through the pages or searching for it in the search field.
  3. Click the **Configure portlet** icon for the Search Center portlet.
  4. Click the **Edit value** icon for the preference **SEARCH_LANG**, and set it to ALL. This enables the display of search results of all available content sources regardless of their language information.
  5. Click the **Edit value** icon for the preference **QUERY_LANG**, and set it to COLLECTION. This enables the Search Center to analyze the search terms by using the language of each collection to be searched.
  6. Click **OK** to activate the changes.
- To enable the default behavior, complete the following steps:
  1. Click the **Administration menu** icon. Then, click **Portlet Management** > **Portlets**.
  2. Locate the Search Center portlet by either navigating through the pages or searching for it in the search field.
  3. Click the **Configure portlet** icon for the Search Center portlet.
  4. Click the **Edit value** icon for the preference **SEARCH_LANG**, and set it to PREFERRED.
  5. Click the **Edit value** icon for the preference **QUERY_LANG**, and set it to PREFERRED.
  6. Click **OK** to activate the changes.

Configuring search scopes for the Search Center portlet

The Search Center provides scopes for users to select different sources for the search. A search scope is a means of filtering the search request to a predefined set of content or content source.

About this task

For search collections provided by Portal Search you can configure the search scopes in the two following ways:

- Each individual search collection is available through its own scope and selection option in the scope pull-down menu.
- The search scope **All Sources** makes all search collections available for search. In other words, if a user selects the option **All Sources** selection option from the pull-down menu, the user's search will search all search collections that are available in the portal.

You can configure additional scopes. For more information about search scopes and how to configure them refer to Managing search scopes and custom links.

Related tasks:

"Customizing the Search Center" on page 687
Customize the Search Center by adding, removing and configuring additional portlets, such as External Search Results or Recommended Links. The external search results portlet displays search results from third-party external search engines such as Yahoo and Google. Using the recommended links portlet, display search results from a collection of predefined links with predefined keywords. You can also configure the All Sources scope or replace it with a customized scope.

"Managing search scopes and custom links" on page 687
Get an overview of search scopes and custom links and how you can manage them.

"Customizing the All Sources scope" on page 692
Delete or replace the All Sources default scope.

Configuring search by metadata for the Search Center portlet

You can configure the Search Center so that users can refine their search by metadata.

About this task

To do this, proceed as follows:

Procedure

1. Open the **Manage Portlets** portlet.
2. Locate the Search Center portlet.
3. Click **Configure portlet** for the Search Center portlet.
4. Click **Edit value** for the preference `displaySearchFilters` , and set it to `true` . This enables search by metadata filters for your users. By default, the meta elements title, keywords, and description of documents are provided as search filters. The actual set of available meta elements depends on the types of content sources that you indexed.
5. Optional: You can add your own custom metadata search filters. Proceed as follows:
   a. Determine which metadata you want to use as search filters. This depends on the metadata that the documents in your search collections provide.
   b. Click **Edit value** for the preference `searchFiltersFields` .
   c. Add the required filters to the list of values.

Replacing the search administrator user ID

If you changed the portal administrator user ID or password, you need to update the search administrator user ID to match the same values.

About this task

Proceed as follows:

Procedure

1. Complete the following steps to change the information stored in the **SearchAdminUser** alias:
   a. Log in to the WebSphere Integrated Solutions Console.
   b. Click **Security** > **Global security**.
   c. Under Authentication, click **Java Authentication and Authorization Service** > **J2C authentication data**.

d. Edit the **SearchAdminUser** alias.
e. Update the user ID and/or password to match your WebSphere Portal Express administrator information.

2. To make sure that the changed administrator user ID takes effect, use one of the following two options:
   - Change the administrator user ID and the password by using the Manage Search administration portlet.
   - Delete the default portal search services and collections. After a portal restart as by the next step, the portal recreates these search services and collections.
3. Restart the WebSphere Portal Express servers.

Customizing the Search Center

Customize the Search Center by adding, removing and configuring additional portlets, such as External Search Results or Recommended Links. The external search results portlet displays search results from third-party external search engines such as Yahoo and Google. Using the recommended links portlet, display search results from a collection of predefined links with predefined keywords. You can also configure the All Sources scope or replace it with a customized scope.

"Managing search scopes and custom links"
Get an overview of search scopes and custom links and how you can manage them.

"Adding and configuring the External Search Results portlet" on page 688
Configure a portlet that retrieves and displays search results from third-party search engines, then add the External Search Results portlet to the Search Center.

"Adding and configuring suggested links" on page 690
As an administrator you can promote specific pages, documents, or other pieces of content by adding search keywords to them in the search results list. The portal then lists these documents as suggested links.

"Customizing the All Sources scope" on page 692
Delete or replace the All Sources default scope.

Managing search scopes and custom links

Get an overview of search scopes and custom links and how you can manage them.

About this task

Search Scopes: You can define Search Scopes to limit search results to specific content locations and specific document types. This enables users to target their searches better. The portal is shipped with two scopes:

All sources
This includes documents with all features from all content sources in the search by a user.

Managed Web Content
This restricts the user's search to sites that were created by IBM Web Content Manager. This scope is only enabled when there is a search collection that is associated with a Web Content Manager site and the site has been crawled.

You can create your own custom search scopes and specify which search locations or content sources they cover.

Note: If you delete a content source, then the documents that were collected from this content source remains available for search by users under all scopes, which included the content source before it was deleted. These documents are available until their expiration time ends. You can specify this expiration time under **Links expire after (days):** under **General Parameters** when you create the content source.

Custom Links: You can add Custom Links with web link shortcuts to search locations. This includes links to external web locations. For example, you can enable users to do direct searches to popular web search engines, such as Google or Yahoo!

Users can select the scopes and custom links from a selection menu that is provided with the search box in the theme and with the Search Center portlet.

You can add icons for the scopes and custom links. Users see these icons in the pull-down selection list of scopes with the Search box and the Search Center.

Note: When you create a custom link, you enter the URL to the target web search engine. Be careful to use the correct format for the URL, as the user query search terms are appended to the URL. For the correct web interface syntax, refer to the help documentation of the target search engine. In some cases it might be possible to determine the web interface syntax as follows:

1. Perform a search with some distinctive search text on the target search engine, for example, an unusual name.
2. Review the browser **URL** field and locate your search string. The part of the URL that precedes your search string is likely to be the Link URL for your target search engine.
3. If your search string is not at the end of the URL, it might be helpful to edit the URL and experiment with different versions with a search string added.

Examples for web interface syntax are:

- For Google: `http://www.google.com/search?&q=`
- For Yahoo: `http://search.yahoo.com/search?p=`

Working with search scopes and custom links: To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**. To work with search scopes and custom links, click **Search Scopes** Portal Search displays the Search Scopes panel. It lists the search scopes and the custom links, shows their status and related information. You can do tasks on the scopes and custom links. For more information about these tasks and the available options, see the **Manage Search** portlet help.

Results

What to do next

Adding and configuring the External Search Results portlet

Configure a portlet that retrieves and displays search results from third-party search engines, then add the External Search Results portlet to the Search Center.

About this task

The External Search Results is a specialized portlet that administrators can add to Portal Search. When you add that portlet to Portal Search, the results of a search that is initiated from the Search page include results from third-party external

search engines. You can add more than one copy of the External Search Results portlet to the Portal Search page. You can also configure each of these portlets to display a specific number of search results.

Note:

1. You must be logged in as Administrator to add an External Search Results portlet to the Search Center.
2. The External Search Results portlet can be added only to the Search Center on the Portal Search page. It does not function anywhere else.

To add the External Search Results portlet to the Portal Search page and configure it, proceed as follows:

Procedure

1. Optional: Add the portlet to the Search Center. You must add the portlet only if the External Search Results portlet was removed from the Search Center.
   a. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
   b. Locate the Search Center page, under **Content Root** > **Search**.
   c. Click **Edit Page Layout**.
   d. Click **Add Portlets**.
   e. Select the **External Search Results** portlet by clicking the check box.
   f. Click **OK** to add the portlet to the page.
   g. Click **Done**.
   h. Go to the Search Center page.
2. Configure the External Search Results portlet:
   a. Click the **Administration menu** icon. Then, click **Portlet Management** > **Portlets**.
   b. In the list of portlets, locate **External Search Results**, and click the **Configure** icon next to the portlet name.
   c. Edit the value of the **searchEngineUrl** parameter. This value represents the URL of the third-party search engine that you want to be queried. The value of the **searchEngineUrl** parameter returns a feed of search results for the specified search terms. The feed is in RSS or ATOM syndication format. The string `${searchTerms}` must be included in the value. This string is replaced by the actual query during the search. For example:

      ```
      http://www.ibm.com/developerworks/views/rss/customrssatom.jsp?zone_by=Lotus&search_by=${searchTerms}
      ```

   d. Optional: Edit the value of the parameter **searchEngineFullPageUrl**. This parameter is optional. You can delete it or leave it empty. When you set this parameter, a **More** link is added to the portlet. It links to the website of the external search engine. The value represents a parameterized form of the URL used to locate the search engine that is queried. The string `${searchTerms}` must be included in the parameter value. During the search, the portlet replaces the string by the actual query. For example:

      ```
      http://www.ibm.com/developerworks/search/searchResults.jsp?searchSite=dW&searchScope=dW&query=${searchTerms}
      ```

      This URL returns the public HTML page of the search engine.
   e. Edit the value of the **numOfEntries** parameter. This parameter determines the maximum number of search results, which are displayed. The default number is 3, but you can enter a different numeric value to increase the maximum.

f. Optional: If the external search engine returns search results in a format that is not supported, or if the provided rendering of the search results is not acceptable, an XSLT file can be specified. Edit the value of the **externalXsltUrl** parameter to supply a URL for such an external XSLT file.

   Note: The XSLT does not create an entire HTML document, but an HTML fragment that can be embedded inside a page.

g. On the page for editing the preference `searchEngineFullPageUrl`, click **OK**.

h. On the page for configuring the External Search Results portlet, click **OK** to save your changes.

Users can now use the External Search Results portlet.

3. If you want to add external search engines, or modify the existing search engine, you must modify the proxy configuration of the portlet. All HTTP requests from the portlet to the external search engines are directed through the outbound HTTP connection service, which is included in the portal. The External Search Results portlet includes an application-specific configuration for this service. In a standard portal installation, it allows only connections to the URL of the default external search engine `http://www.ibm.com/developerworks/views/rss/*`. To change this, you must modify the `proxy-config.xml` file that is bundled with the portlet. To modify the file, proceed as follows:

   a. Go to the file `searchCenter.war`. It is in the directory *`PortalServer_root`*`/search/wp.search.portlets/search/portlet/installableApps` .

   b. Edit the file `proxy-config.xml`. It is in the directory `WEB-INF`.

   c. Add an `proxy:policy` element, or modify the existing one. For more information, read *Creating an outbound HTTP connection configuration profile*. The current policy element allows connections to the following URL: `http://www.ibm.com/developerworks/views/rss/*`. You can change the URL attribute to specify another site, for example `http://www.ibm.com/products/*`. If you want to add several instances of the portlet on a page that show search results from different sources, duplicate the `proxy:policy` element. Add a `policy` element for each instance of the portlet, with the URL attribute that matches the search source.

   d. Repackage the WAR file and update the Search Center web module.

   Searches by users are now done by the modified search configuration.

Related tasks:

"Creating an outbound HTTP connection configuration profile" on page 3014
This configuration task creates an outbound HTTP connection configuration profile by using the settings that you specify in an XML document. Use this task if you want to initially create outbound HTTP settings for your configuration. You can create a global configuration or an application-scoped configuration.

Adding and configuring suggested links

As an administrator you can promote specific pages, documents, or other pieces of content by adding search keywords to them in the search results list. The portal then lists these documents as suggested links.

Before you begin

- You must be logged in as an administrator to add the Suggested Links portlet to the search center.
- Ensure that tagging is enabled. For more information, see "Tagging and rating" on page 1297.

- Ensure that Dojo tagging and rating options are enabled for your Search and Tag Center profile. For more information, see "Enabling and disabling the Dojo tagging and rating options for additional profiles" on page 1341.

About this task

To add or edit keywords for suggested links, proceed as follows:

Procedure

1. Use the portal Search Center to search for the kind of documents that you want to promote. The Search Center lists the documents in the search results list.
2. Click the option **Edit keywords for Suggested Links** for the document that you want to promote. The Search Center opens a dialog window named **Edit keywords for Suggested Links**. This option is available only for users with the administrator role on the Search Center.
3. Add one or more keywords to the selected document. Separate the individual keywords by blanks. You can also edit or delete keywords as required.
4. Click **Save**.
5. Add more keywords as required and save them.
6. Click **Close**.

Results

- When your users search for a search string that you added to documents as a keyword, these documents display in the **Suggested Links** portlet.
- For assigned administrators, the **Suggested Links** portlet provides an **Edit** link for each suggested link. When an administrator clicks that link, the portlet opens the dialog box **Edit keywords for Suggested Links**. The administrator can now add more tags or delete existing tags. This administrator role assignment must be the same as for the Search Center to display the option **Edit keywords for Suggested Links**.

Adding keywords to documents uses the portal tagging functions. You can configure the Tag Cloud to display a **Suggested links** tab and a corresponding view to users. When you click that tab, the tag cloud shows only tags that you added as keywords, and the related documents to which those keywords were added. You can use the **Suggested Links** tab of the Tag Center to view all suggested links that you created by adding your keywords.

In a typical example scenario, you can work in two stages:

1. To add new tags or edit or delete existing keywords, you can use the Search Center, either from the main results view or from the Suggested Links portlet.
2. To browse an overview of the keywords that exist already, you can use the Suggested Links tab of the Tag Center. You can add and remove keywords from documents as required.

Note: Suggested links are case-sensitive. The tagging and rating normalization parameter `com.ibm.wps.cp.tagging.normalization.displayNormalizedNames` does not apply to suggested links. For example, if an administrator adds a suggested link of `WebSphere`, a user search for `websphere` does not return `WebSphere` as a result. This action is independent of whether the tagging and rating parameter `com.ibm.wps.cp.tagging.normalization.displayNormalizedNames` is set to `true` or `false`.

"Configuring the Suggested Links view"
Customize the display of search results to show users the preferred or recommended results and associated links.

Configuring the Suggested Links view:

Customize the display of search results to show users the preferred or recommended results and associated links.

About this task

The Suggested Links view of the Tag Cloud displays predefined search results and links to users separately from the regular result set. You can add keywords to documents in the Search Center result list to control which results appear in the Suggested Links list.

To configure the Suggested Links view, proceed as follows:

Procedure

1. Click the **Administration menu** icon. Then, click **Portlet Management** > **Portlets**.
2. Click the **Configure** icon next to the **Suggested Links** portlet.
3. Add or edit the following parameter and the appropriate value:

   numberOfLinks = (3)
   Determines the maximum number that is displayed by the Suggested Links portlet. The default number is 3.
4. Click **OK** to save the portlet configuration. Users can now use the Suggested Links portlet.

Customizing the All Sources scope

Delete or replace the All Sources default scope.

About this task

When the user clicks the **Search Center** menu list to select a scope, the first scope on the list is the default scope. By default, All Sources is set to be the default scope.

All Sources scope is a special scope in portal. This scope has a unique ID and it searches in all the search collections that are accessible to the user, including collections from existing local and remote search services. The All Sources scope can be deleted or customized just like other search scopes. Three possible actions can be done on the All Sources scope:

- Choose a different default scope
- Delete the All Sources scope
- Readd the All Sources scope

To change the default search scope from All Sources to a different scope, reorder the scopes by using search administration:

1. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
2. Click **Search Scopes**.

3. Move the scope that you want to be the default scope to the beginning of the scopes list by clicking the up-arrow icon next to the scope name. The first scope on the list becomes the default search scope.

Note: Users must clear their browser cache for the new scope to be available and displayed in the correct position.

The All Sources scope is created by using a mechanism that is called out-of-the-box (OOB) scopes registration and it is stored as a property in the WebSphere Integrated Solutions Console. Since the All Sources scope is added by using the OOB scopes registration mechanism, you must delete the All Sources scope by using both the WebSphere Integrated Solutions Console and the search administration interface:

Procedure

1. Remove the All Sources scope from the scopes list by using search administration:
   a. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
   b. Click **Search scopes**.
   c. Delete the All Sources scope.
2. Remove the All Sources property from the WebSphere Integrated Solutions Console:
   a. In the navigation click **Resources** > **Resources Environment** > **Resource Environment Providers**.
   b. Make the appropriate selection, depending on your version of WebSphere Application Server and your portal environment:
      - In the Resource Environment Providers page, select the appropriate node or cluster from the scopes drop-down list, or uncheck the Show Scope selection drop-down check box and select one of the following options, depending on your portal environment:
        – If your portal is running as a single server, select **Browse Nodes** and select the node.
        – If your portal is installed in a cluster, select **Browse Clusters** and select the portal cluster.
   c. Select the **WP ScopeConfigService** service.
   d. Click **Custom Properties**.
   e. Select the **All Sources** property and delete it.

Results

You can readd the All Sources scope only if this scope does not exist in the current scopes list. The All Sources scope is added by using the WebSphere Integrated Solutions Console. Once you add it using the console, it is automatically added to the Search Center scopes list. A service that is called ScopeConfigService is registered and the configured All Sources scope is added as custom property of this service. The scope is implemented based on a scope XML element.

1. In the navigation click **Resources** > **Resources Environment** > **Resource Environment Providers**.
2. Make the appropriate selection, depending on your portal environment:

- In the Resource Environment Providers page, select the appropriate node or cluster from the scopes drop-down list, or uncheck the Show Scope selection drop-down check box and select one of the following options, depending on your portal environment:
  - If your portal is running as a single server, select **Browse Nodes** and select the node.
  - If your portal is installed in a cluster, select **Browse Clusters** and select the portal cluster.

3. Select the **WP ScopeConfigService** service.
4. Click **Custom Properties**.
5. Click **New** to create a new scope property.
6. Name the new property **All Sources**. Enter this XML scope element as the property value:

```
<scope id="com.ibm.lotus.search.ALL_SOURCES">
<title xml:lang="en">All Sources</title>
<description xml:lang="en">All Sources accessible by the user</description>
<scopeProperty key="iconURI" value="/wps/images/icons/scope_search_all.gif"/>
<scopeProperty key="isVisibletoAnonymousUser" value="true"/>
<scopeElement></scopeElement>
</scope>
```

   Note: You can change the name, description, and icon for the new scope. To create a scope in a language other than English, change the xml:lang attribute to the required locale, such as *de* for German.
7. Click **Save** to save the new property.

Using the WebSphere Integrated Solutions Console to administer Portal Search

You can administer Portal Search by using the WebSphere Integrated Solutions Console and using resource providers in XML format.

About this task

You can administer search services, search collections, and search scopes. You can create, configure, or delete these.

To administer Portal Search by using the WebSphere Integrated Solutions Console, proceed by the following steps.

Procedure

1. Open the WebSphere Integrated Solutions Console.
2. Select **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Locate and then click **WP_SearchConfigService**.
4. Under **Additional Properties**, click **Custom properties**.
5. Click **PortalSearchService** to edit the resource.
6. Edit the **Value** field, which displays the value from the XML file that is used to configure Search, and set the parameter as required.

   Tip: Refer to the following file for information about the XML schema : *PortalServer_root*/search/wp.search.provider/core/service/schema/WplcSearchService.xsd.
7. Apply and then save the change.

8. Restart the portal.

Results

What to do next

Related reference:

"Search service configuration parameters" on page 674
Learn about the portal search service parameters and possible values.

Setting up search collections

View information on setting up search collections for search by users. This also includes creating content sources and managing search scopes and custom links.

"Creating and configuring search collections"
Get an overview of how you manage search collections and their content sources.

"Setting up a JCR search collection" on page 698
A JCR search collection is a special purpose search collection that is used by WebSphere Portal applications. It is not designed to be used alongside user-defined search collections. A JCR search collection requires a special setup. This setup includes the creation of a new content source for the search collection. Under normal circumstances, you do not need to re-create the JCR search collection. However, in rare cases you might need to re-create it, for example if you deleted the default JCR search collection.

"Managing the content sources of a search collection" on page 701
Search collections consist of one or more content sources. You can administer the content sources.

"Exporting and importing search collections" on page 707
View the steps to export search collections from a source portal and import them into a target portal.

Creating and configuring search collections

Get an overview of how you manage search collections and their content sources.

To administer search collections, go to the **Manage Search** portlet. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**. Then, click **Search Collections**. This panel includes creating, updating, and removing search collections, and other administrative tasks that refer to search collections. For more information and step-by-step instructions for performing administrative tasks, refer to the portlet help.

Managing search collections:

When you select **Search Collections**, **Manage Search** displays the Search Collections panel. It lists the search collections in your portal and related information, and it allows you to select options and do tasks on the search collections and their content sources.

Note: The selectable options that are displayed and available for collections and content sources depend on their type and setup.

In the Search Collections panel you can select the following option icons and do the following tasks:

- **Change the search collection** with which you want to work. To do this, select another search collection from the pull-down list.
- **New collection**. Select this option to create a new search collection.

  Notes:

  1. You cannot create additional search collections for the default Content Model search service.
  2. When you specify the directory location for the collection, be aware that creating the collection can overwrite files in that directory.
- **Refresh** the list of collections.
- **Locate a collection** and do one of the following tasks by clicking the appropriate icon for that collection:
  - **Search and Browse Collection**. Use this option to work with the documents of the selected collection. You can complete the following administrative tasks:
    - Browse the documents of the selected collection.
    - View the individual documents of the selected collection.
    - Search the documents of the selected collection.
    - Edit the fields of the documents in the selected collection.
    - Delete documents from the selected collection.
  - **Import or Export Collection**. Use this option to import or export the selected search collection. Portal Search provides a Portal Search XML interface for this feature. The export and import operations can be of benefit when you upgrade to software levels, which are not necessarily compatible with the data storage format of older versions of the software. To prevent loss of data, you export all data of search collections to XML files before you upgrade the software. Then after you upgrade the software level, you can use the previously exported files to return the search collection data back into the new software level.

    Notes:

    1. Before you export a collection, make sure that the user who is running the portal application process has write access to the target directory location. Otherwise, you might get an error message, such as `File not found`.
    2. You can import collection data only into an empty collection. You cannot import collection data into a target collection that has content sources or documents already.
    3. When you import collection data into a collection, all collection settings are overwritten by possibly imported settings. For example, the language setting is overwritten.
    4. When you import a collection, a background process fetches, crawls, and indexes all documents that are listed by URL in the previously exported file.
  - **Delete Collection**. Use this option to delete the selected search collection.
- **Select a collection** by clicking the collection name link. Portal Search displays the Content Sources and the Status of the selected collection. You can select the following option icons and perform the following tasks:
  - **New Content Source**. Use this option to create a new content source for this collection. You can create more than one content source for a search collection.
  - **Refresh** the list of content sources and the status that is shown for this collection.

- **Work with the content sources of the collection**.
- **View** the **Collection Status** information of the selected search collection. The status fields show the following data that changes over the lifetime of the search collection:

  Search Collection Name:
  Shows the name of the selected search collection.

  Search Collection Location:
  Shows the location of the selected search collection in the file system. This is the full path where all data and related information of the search collection is stored.

  Collection Description:
  Shows the description of the selected search collection if available.

  Search Collection Language:
  Shows the language for which the search collection and its index are optimized. The index uses this language to analyze the documents when indexing, if no other language is specified for the document. This feature enhances the quality of search results for users, as it allows them to use spelling variants, including plurals and inflections, for the search keyword.

  Summarizer used:
  Shows whether a static summarizer is enabled for this search collection. The static summarizer creates a summary of the page, which is based on the page's full content. The page's full content can include metadata, HTML elements, and Web Content Manager templates. These additional elements might be interpreted as text and thus become a part of the page's summary. Do not use the static summarizer if the page's summary contains a large amount of noise from these additional elements.

  Last update completed:
  Shows the date when a content source defined for the search collection was last updated by a scheduled update.

  Next update scheduled:
  Shows the date when the next update of a content source that is defined for the search collection is scheduled.

  Number of active documents:
  Shows the number of active documents in the search collection, that is, all documents that are available for search by users.

  Notes:

  1. To update the status information, click **Refresh**. Clicking the refresh button of the browser does not update the status information.
  2. If you delete a portlet from the portal after a crawl of the portal site, the deleted portlet is no longer listed in the search results. However, refreshing the view does not update the status information about the **Number of active documents**. This information is not updated until after the next cleanup run of portal resources.

Related concepts:

"Delayed cleanup of deleted portal pages" on page 278
Get an overview of the cleanup service for portal pages and their dependent resources.

Related tasks:

"Managing the content sources of a search collection" on page 701
Search collections consist of one or more content sources. You can administer the content sources.

Related reference:

"Creating the portal site search collection can fail" on page 721
Creating the portal site search collection can fail due to a file path length restriction.

Setting up a JCR search collection

A JCR search collection is a special purpose search collection that is used by WebSphere Portal applications. It is not designed to be used alongside user-defined search collections. A JCR search collection requires a special setup. This setup includes the creation of a new content source for the search collection. Under normal circumstances, you do not need to re-create the JCR search collection. However, in rare cases you might need to re-create it, for example if you deleted the default JCR search collection.

About this task

The portal installation has the JCR search collection that is created by default. It is named JCRCollection1. If this collection is removed or does not exist for other reasons, you can manually re-create the JCR search collection. The portal also re-creates the JCR search collection if you edit Web Content Manager content. Web Content Manager Authoring and its search capability are required to have the JCR search collection available, paired with the respective content source. If the JCR search collection gets deleted, a search is not possible by using the Authoring portlet. The JCR search collection can be used only by a search portlet that knows how to present and deal with the search result in which the returned information is useless in a more generic context of search. This search collection is also flagged so that it does not participate in search by using the All Sources search scope. An administrator cannot manually add it. The JCR search collection is a special purpose search collection that the JCR requires to allow specialized application to do low-level searches in the repository. The JCR search collection is required to be available only once.

Notes:

For Web Content Manager:

If you use Web Content Manager, the `JCRCollection1` collection is created the first time that you create a web content item, if it does not exist. In this case, it might not be necessary to create the collection manually, although it is fine to create it manually first, if required. It is used by the search feature within the Web Content Manager authoring portlet. If you delete this search collection, you might not be able to search for items within the authoring portlet.

For virtual portals:

When you create a virtual portal, the creation of the JCR search collection depends on whether you create the virtual portal with or without content:

- If you create the virtual portal with content, the portal creates the JCR collection for the virtual portal by default.
- If you create only the virtual portal and add no content to it, the portal creates no JCR collection with it. It gets created only when content is added to the virtual portal.

You can view the URL of the JCR search collection in the search administration portlet Manage Search of the virtual portal. The URL looks as follows:
```
 http://host_name:port_number/wps/seedlist/myserver/
hello?Action=GetDocuments&Format=ATOM&Locale=en_US&Range=100
&Source=com.ibm.lotus.search.plugins.seedlist.retriever.jcr.JCRRetrieverFactory&S
```
Where *wsid* is the actual workspace ID of the virtual portal. The workspace ID is the identifier of the workspace in which the content item is created, stored, and maintained. For example, if the workspace ID of the virtual portal is 10, then the URL looks as follows:
```
http://host_name:port_number/wps/seedlist/myserver/
hello?Action=GetDocuments&Format=ATOM&Locale=en_US&Range=100
&Source=com.ibm.lotus.search.plugins.seedlist.retriever.jcr.JCRRetrieverFactory&S
```
If the JCR search collection was deleted, or if you added content to an originally empty virtual portal and the JCR search collection was not automatically created, complete the following steps:

- If you are using a virtual portal, go to the Security tab of the content source to verify that the workspace ID of the virtual portal is correct.
- If the JCR search collection was deleted, run the **ConfigEngine** task `create-textsearch-collections` to re-create the JCR search collection.

If neither of the preceding options succeed in creating the JCR search collection, manually set up the JCR search collection.

To set up a JCR search collection manually, proceed as follows:

Procedure

1. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
2. Click **Search collections.**
3. To create a new search collection, click **New collection**.
4. Specify the following values for the parameters as required:

   Search Service
   Select the required search service the JCR collection uses. If you have a stand-alone environment, select **Default Portal Service**. If you have a clustered environment, select **Remote Search Service**.

   Location of collection
   The directory location for the collection where you intend the search collection to be created. This parameter is to be specified as *index directory location/collection name*. For example, if the index directory is `c:/JCR` and the collection name is JCRCollection1, then the location of the collection must be specified as `c:/JCR/JCRCollection1`.

   Note: Verify that the **jcr.textsearch.indexdirectory** resource value is updated with `c:/JCR`. To view this resource and corresponding value, complete the following steps:

   a. Go to **Resources** > **Resource Environment** > **Resource Environment Providers** and select **JCR ConfigService PortalContent**.
   b. In the Additional Properties section of the Configuration window, select **Custom properties**.
   c. Find **jcr.textsearch.indexdirectory** and update the value if needed.

Name of collection.
The name of the collection must be `JCRCollection1`.

Description of collection
This parameter is optional. Specify `JCR seedlist collection`.

Specify Collection language
Specify the collection language. By default this parameter is set to English (United States).

After you create the new collection, you can see the name of the collection you created in the list.

5. Double-click the collection that you created.
6. To create the content source for the new search collection, click **New Content Source**.
7. Specify the collection parameters as follows:
   - For the type of the content source, select **Seedlist Provider**.
   - Provide the name for the new Content Source in the field **Content Source Name**. For example, you can specify `JCRSource`.
   - Specify the value for the **URL** field **Collect documents linked from this URL:** as follows:

     ```
     http://server_name:port_number/wps/seedlist/myserver?Action=GetDocuments
        &Format=ATOM&Locale=en_US&Range=100
        &Source=com.ibm.lotus.search.plugins.seedlist.retriever.jcr.JCRRetrieverFactory&Start=0
        &SeedlistId=1@OOTB_CRAWLER1
     ```

     In this URL the range parameter specifies 100 documents in one page of a session and the workspace ID of the base portal is 1.

     If you are working in a virtual portal, specify the content source URL for the virtual portal as follows:

     ```
     http://server_name:port_number/wps/seedlist/myserver/
        virtual_portal_context?Action=GetDocuments
        &Format=ATOM&Locale=en_US&Range=100
        &Source=com.ibm.lotus.search.plugins.seedlist.retriever.jcr.JCRRetrieverFactory&Start=0
        &SeedlistId=1@OOTB_CRAWLERwsid
     ```

     Where *wsid* is the workspace ID of the virtual portal. To determine the workspace ID of the virtual portal, complete the following steps:

     a. Click the **Administration menu** icon. Then, click **Portal Analysis** > **Enable Tracing**.
     b. In the **Append these trace settings:** field, add `com.ibm.icm.ts.*=finest` to enable the JCR **`TextSearch`** trace.
     c. Save all Web Content Manager documents in the virtual portal.
     d. In `trace.log`, you can find trace information similar to the following:

        ```
        [6/5/13 18:51:04:337 IDT] 000001c3 BaseDBImpl 3 insertSeedlistEvents:
         Inserted event: Event:
        action='Update_Node(3)', timestamp='2013-06-05 18:51:04.337', document id=,<workspace: 3,
        ```

8. Go to the **Security** tab.
9. Enter the user ID and password of the WebSphere Portal Express administrator.
10. Click **Create** to create the new content source. If the Content Source was created successfully, the following message is displayed on the page:

    `EJPJB0025I: Content source source_name in collection collection_name is OK.`
11. You can start the crawler manually or schedule it to run at regular intervals.
    - To start the crawler manually, go to the content source and click the **Start Crawler** button for the content source.

- To schedule the seedlist crawler, click the **Edit Content Source** button, and click the **Scheduler** tab. Specify the date and time and the frequency for the crawl. The crawler is triggered automatically at the time that you scheduled.

"Re-creating a JCR search collection after it was deleted"
The portal installation has the JCR search collection created by default. It is named JCRCollection1. If this collection is removed or does not exist for other reasons, you can re-create the JCR search collection.

Related reference:

"JCR search service configuration parameters" on page 680
The following search service configuration parameters can be modified to enable and configure searching for content that is stored in the JCR database. These JCR search service configuration parameters can be modified by accessing the **JCR ConfigService PortalContent** resource environment provider.

Re-creating a JCR search collection after it was deleted:

The portal installation has the JCR search collection created by default. It is named JCRCollection1. If this collection is removed or does not exist for other reasons, you can re-create the JCR search collection.

Procedure

- If you are using a virtual portal, navigate to the Security tab of the content source to verify that the workspace ID of the virtual portal is correct.
- If the JCR search collection was deleted, run the **`ConfigEngine`** task `create-textsearch-collections` to re-create the JCR search collection.
- If neither of the preceding options succeed in creating the JCR search collection, manually re-create the JCR search collection. Refer to *Setting up a JCR search collection* to view these steps.

Managing the content sources of a search collection

Search collections consist of one or more content sources. You can administer the content sources.

About this task

To work with content sources of a collection, go to the **Manage Search** portlet. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**. Then, click **Search Collections**. Then, select a search collection by clicking the collection name link. Portal Search displays the Content Sources panel. It shows the status of the selected search collection and lists its content sources and their status. It shows information that is related to the individual content sources, and you can do tasks on these content sources.

You can select the following option icons and do the following tasks in relation to the search collection, which you selected from the Search Collections list:

- **New Content Source**. Use this option to create a new content source for the search collection that you selected from the Search Collections list. For detailed instructions, refer to the portlet help.

  Notes:

  1. WebSphere Portal Express search crawler supports basic authentication, therefore Unicode character set for user and password is not supported.

2. You can configure a search collection to cover multiple content source of different types. For example, you can combine portal sites, websites, and local document collections.
3. The selectable options and data entry fields that are displayed under the different configuration tabs depend on which type of content source you select.
4. If you select Portal site, the appropriate data for your portal site is already complete.
5. If you select WCM site (Web Content Manager site), you need to enter the appropriate data. For information about how to construct the URL for the content source refer to *Seedlist 1.0 REST service API* in the Web Content Manager documentation.
6. For some content sources, you might need to enter sensitive data, such as a user ID and password. For example, this action applies to secured WebSphere Portal Express sites or HTTP sites that require a user ID and password. To ensure encryption of this sensitive data when it is stored, update and run the file `searchsecret.xml` by using the XML configuration interface before you create the content source.
7. If you are using form-based authentication, specify the following fields:

   Note: Each host name within a crawler can have a single form-based security definition, a single basic authentication definition, or multiple basic authentication realm definitions.

   User name and Password
   The user name and password that is associated with the login form.

   URL of login FORM
   Specify the submit URL value of the login form for the site that will be crawled. The crawler issues a POST request to this URL and passes it on to the user name and password, through the Ajax proxy.

   User FORM field name
   Specify the user name field value in the login form, where the user name is defined.

   Password FORM field name
   Specify the password field value in the login form, where the password is defined.
8. When you create a portal site content source in a portal cluster environment that is configured with SSL, you need to provide the cell security information for the web server and the nodes. For example, in a cluster with the cluster URL `https://web_server/wps/portal`, the primary node URL `http://node_1:10039/wps/portal`, and the secondary node URL `http://node_2:10050/wps/portal` you need to provide the user ID and password for the web server and both nodes 1 and 2.
9. Under the **General parameters** tab, you must set the URL for the content source in a field **Collect documents linked from this URL:** . The crawler needs this URL for crawling. For information about how to construct the URL for the content source refer to *Seedlist 1.0 REST service API* in the Web Content Manager documentation.

   Note: A crawler failure can be caused by URL redirection problems. If this occurs, try by editing this field accordingly, for example, by changing the URL to the redirected URL.

10. For crawling a website content source, you can set a timeout under the **General parameters** tab under the option **Stop collecting after (minutes):** . This timeout works as follows:
    a. The timeout works only for website content sources.
    b. The timeout works as an approximate time limit. It might be exceeded by some percentage.
    c. The crawl action is put in a queue. It might therefore take several minutes until it is run and the time counter starts. It might therefore seem that the crawl takes longer than the timeout that you set.

    Therefore, when you start the crawl by clicking **Start Crawler**, allow for some time tolerance and be aware of the time that is required for crawls and imports and availability of documents.
11. In the **Advanced Parameter** tab, the entry field for the Default Character Encoding contains the initial default value `windows-1252`, regardless of the setting for the Default Portal Language. To go to the Default Portal Language, Click the **Administration menu** icon. Then, click **Portal Settings** > **Global Settings**. Enter the required default character encoding, depending on your portal language. Otherwise, documents might be displayed incorrectly under Browse Documents.
12. Before you start the crawl, set the preferred language of the crawler user ID to match the language of the search collection that it crawls.
13. You start the initial crawl on a newly created content source by either of the following options:
    – After you created a new content source, click the **Start Crawler** icon. This starts an immediate crawl.
    – When you create the content source, define a schedule under the **Schedulers** tab. The crawl starts at the next possible time that you specified.

- **Refresh**. Use this option to update the list of content sources and the status that is shown for this collection.
- Select the following option icons and do the following tasks on a content source:
  – **View Content Source Schedulers**. Use this option to view and manage schedulers. This option is only available if you defined schedulers for the content source.
  – **Start Crawler**. Click this icon to start a crawl on the content source. This action updates the contents of the content source by a new run of the crawler. While a crawl on the content source is running, the icon changes to **Stop Crawler**. Click this icon to stop the crawl. Portal Search refreshes different content sources as follows:
    - For website content sources, documents that were indexed before and still exist in the content source are updated. Documents that were indexed before, but no longer exist in the content source are retained in the search collection. Documents that are new in the content source are indexed and added to the collection.
    - For WebSphere Portal Express sites, the crawl adds all pages and portlets of the portal to the content source. It deletes portlets and static pages from the content source that were removed from the portal. The crawl works similarly to the option Regather documents from Content Source.
    - For IBM Web Content Manager sites, Portal Search uses an incremental crawling method. Additionally to added and updated content, the Seedlist explicitly specifies deleted content. In contrast, clicking Regather documents

from Content Source starts a full crawl; it does not continue from the last session, and it is therefore not incremental.

- For content sources created with the seedlist provider option, a crawl on a remote system that supports incremental crawling, such as IBM Connections, behaves like a crawl on a Web Content Manager site.

– **Regather documents from Content Source**. This option deletes all existing documents in the content source from previous crawls and then starts a full crawl on the content source. Documents that were indexed before and still exist in the content source are updated. Documents that were indexed before, but no longer exist in the content source are removed from the collection. Documents that are new in the content source are indexed and added to the collection.

– **Verify Address of Content Source**. Click this icon to verify that the URL of the content source is still live and available. Manage Search returns a message about the status of the content source.

- **For web site content sources:** Documents that were indexed before and still exist in the content source are updated. Documents that were indexed before, but no longer exist in the content source are retained in the search collection. Documents that are new in the content source are indexed and added to the collection.
- **For WebSphere Portal Express sites** the crawl adds all pages and portlets of the portal to the content source. It deletes portlets and static pages from the content source that were removed from the portal. The crawl works similarly to the option Regather documents from Content Source described later.
- **For Web Content Manager sites** Portal Search uses an incremental crawling method. Additionally to added and updated content, the Seedlist explicitly specifies deleted content. In contrast, clicking Regather documents from Content Source starts a full crawl; it does not continue from the last session, and it is therefore not incremental.

–

Notes:

- It is of benefit to define a dedicated crawler user ID. The pre-configured default portal site search uses the default administrator user ID `wpsadmin` with the default password of that user ID for the crawler. If you changed the default administrator user ID during your portal installation, the crawler uses that default user ID. If you changed the user ID or password for the administrative user ID and still want to use that user ID for the Portal Search crawler, you need to adapt the settings .

  To define a crawler user ID, select the **Security** tab, and update the user ID and password. Click **Save** to save your updates.
- If you modify a content source that belongs to a search scope, update the scope manually to make sure that the scope still covers that content source. Especially if you changed the name of the content source, edit the scope and make sure that it is still listed there. If not, add it again.
- If you delete a content source, then the documents that were collected from this content source remains available for search by users under all scopes, which included the content source before it was deleted. These documents are available until their expiration time ends. You can specify this expiration time under **Links expire after (days):** under **General Parameters** when you create the content source.

– **Delete Content Source**. Click this icon to delete the selected content source.

Note: If you delete a content source, then the documents that were collected from this content source remains available for search by users under all scopes, which included the content source before it was deleted. These documents are available until their expiration time ends. You can specify this expiration time under **Links expire after (days):** under **General Parameters** when you create the content source.

– **Start Crawler**. Click this icon to start a crawl on the content source. This updates the contents of the content source by a new run of the crawler. While a crawl on the content source is running, the icon changes to **Stop Crawler**. Click this icon to stop the crawl. Portal Search refreshes different content sources as follows:
  - **For web site content sources:** Documents that were indexed before and still exist in the content source are updated. Documents that were indexed before, but no longer exist in the content source are retained in the search collection. Documents that are new in the content source are indexed and added to the collection.
  - **For WebSphere Portal Express sites** the crawl adds all pages and portlets of the portal to the content source. It deletes portlets and static pages from the content source that were removed from the portal. The crawl works similarly to the option Regather documents from Content Source described later.
  - **For Web Content Manager sites** Portal Search uses an incremental crawling method. Additionally to added and updated content, the Seedlist explicitly specifies deleted content. In contrast, clicking Regather documents from Content Source starts a full crawl; it does not continue from the last session, and it is therefore not incremental.

• **View** information about the status and configuration of the content source.

Note: To update the status information, click the **Refresh** button or the refresh button of the browser.

• **View** the **Collection Status** information of the selected search collection. The status fields show the following data that changes over the lifetime of the search collection:

Search Collection Name:
Shows the name of the selected search collection.

Search Collection Location:
Shows the location of the selected search collection in the file system. This is the full path where all data and related information of the search collection is stored.

Collection Description:
Shows the description of the selected search collection if available.

Search Collection Language:
Shows the language for which the search collection and its index are optimized. The index uses this language to analyze the documents when indexing, if no other language is specified for the document. This feature enhances the quality of search results for users, as it allows them to use spelling variants, including plurals and inflections, for the search keyword.

Summarizer used:
Shows whether a static summarizer is enabled for this search collection. The static summarizer creates a summary of the page, which is based on the page's full content. The page's full content can include metadata,

HTML elements, and Web Content Manager templates. These additional elements might be interpreted as text and thus become a part of the page's summary. Do not use the static summarizer if the page's summary contains a large amount of noise from these additional elements.

Last update completed:
Shows the date when a content source defined for the search collection was last updated by a scheduled update.

Next update scheduled:
Shows the date when the next update of a content source that is defined for the search collection is scheduled.

Number of active documents:
Shows the number of active documents in the search collection, that is, all documents that are available for search by users.

Notes:

1. To update the status information, click **Refresh**. Clicking the refresh button of the browser does not update the status information.
2. If you delete a portlet from the portal after a crawl of the portal site, the deleted portlet is no longer listed in the search results. However, refreshing the view does not update the status information about the **Number of active documents**. This information is not updated until after the next cleanup run of portal resources.

For more details about the available options for content sources, see the **Manage Search** portlet help.

"Applying filter rules"
Portal Search provides a facility for applying filter rules to the crawler process. The crawler filters control the crawler progress and the type of documents that are indexed and cataloged.

Related concepts:

"Delayed cleanup of deleted portal pages" on page 278
Get an overview of the cleanup service for portal pages and their dependent resources.

Related reference:

"Hints and tips for using Portal Search" on page 714
View some useful tips for using Portal Search.

"Creating the portal site search collection can fail" on page 721
Creating the portal site search collection can fail due to a file path length restriction.

Related information:

Web Content Manager - Seedlist 1.0 REST service API

Applying filter rules:

Portal Search provides a facility for applying filter rules to the crawler process. The crawler filters control the crawler progress and the type of documents that are indexed and cataloged.

You can define the filter rules when creating a content source of type **Web site** only. You define the filters under the **Filters** tab. You can combine any combination of the following filtering options for a filtering rule:

Table 81. Filter rule options

Filter option	Possible settings	
Apply rule while:	Collecting documents	Indexing documents
Rule type:	Exclude	Include
Rule basis:	URL text	File type

Depending on your choices on the options, the filter rules result in the following behavior for selection of documents or pages:

Table 82. Document selection behavior by filter rules

Option for applying the rule	Selected rule type option: Exclude	Selected rule type option: Include
Apply rule while **Collecting documents**	The page or document is excluded, and links on the page are **not** explored.	Only pages or documents that meet the criteria and that have a link on a parent page that meets the criteria, starting with the initial site.
Apply rule while **Adding documents to index**	The page or document is excluded, and links on the page are explored.	The entire site is searched, and pages or documents that meet the filtering criteria will be included.

Note: When you use the option **Apply rule while Collecting documents** with **Rule type: Include**, make sure that the URL in the field **Collect documents linked from this URL:** fits the specified rule; otherwise no documents will be collected. For instance, crawling the URL `http://www.ibm.com/products` with the URL filter `*/products/*` will not give any results, because the rule has a trailing slash, but the URL does not. But either crawling `http://www.ibm.com/products/` with the URL filter `*/products/*` (both with trailing slash) or crawling `http://www.ibm.com/products` with the URL filter `*/products*` (no trailing slash) will work.

For more details about filter rules and how to apply them refer to the **Manage Search** portlet and its help.

Exporting and importing search collections

View the steps to export search collections from a source portal and import them into a target portal.

About this task

The following are possible use cases for exporting and importing search collections:

- You verified your search collections on a test portal, and you want to move these collections to your production portal.
- You verified your search collections locally on a portal, and you want to move these collections to a configuration with remote search.
- You verified your portal search configuration and search collections on a single portal, and you want to move these collections to a portal cluster environment.
- You are staging your portal to production by using the ReleaseBuilder.

To export and import your search collections, use the **Import or Export Collection** option of the Manage Search portlet. You can use that option for both exporting and importing. For more details about these tasks and the Manage Search portlet,

refer to the portlet help. To export and import your search collections, proceed by the following steps:

Procedure

1. To include the security information when you export the search collection, add the **WS_KEY** parameter to the search service that contains the source search collection that you want to export. Complete the following steps:
   a. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
   b. Click **Search Services**.
   c. Click the **Edit** icon for the search service that contains the search collection that you want to export.
   d. In the **Parameter key** field, enter `WS_KEY`.
   e. In the **New parameter value** field, enter `secret`.
   f. Click **Add Parameter**.
   g. Click **OK**.

   Note: If you do not export the security information when you export a search collection, you must manually add the user name and password to each content source after you import the search collection into the target portal.
2. On the source portal, export your search collections. This exports the configuration data and all document URLs of your search collections.

   Notes:
   a. Before you export a collection, make sure that the user who is running the portal application process has write access to the target directory location.Otherwise, you might get an error message, such as `File not found`.
   b. When you specify the target directory location for the export, be aware that the export can overwrite files in that directory.
3. Document all of the following data:
   - The target file names and directory locations to which you export the collections. For example, `C:\ibm\wp_profile\PortalServer\collections`.
   - The following configuration data of the collections: The **location, name, description**, and **language** for each collection.
4. Create the search collections on the target portal. This task creates the empty shell for the search collection. Complete the following data entry fields and select the following options according to the data that you documented:
   - **Location of Collection:** Specify the new collection location.
   - **Name of Collection:** Specify the collection name. The name can match the old setting, but does not have to match it.
   - **Description of Collection:** Specify a collection description. The description can match the old setting, but does not have to match it.
   - **Specify Collection Language:** Select this to match the old setting.
   - **Select Summarizer:** You do not need to select this option. The value is overwritten by the import.

   You do not have to add content sources or documents; that is completed by the import task.

5. To include the security information when you import the search collection, add the **WS_KEY** parameter to the search service that contains the target search collection. Complete the following steps:
   a. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
   b. Click **Search Services**.
   c. Click the **Edit** icon for the search service that contains the target search collection.
   d. In the **Parameter key** field, enter `WS_KEY`.
   e. In the **New parameter value** field, enter `secret`.
   f. Click **Add Parameter**.

   Note: If you do not import the security information when you import a search collection, you must manually add the user name and password to each content source after you import the search collection into the target portal.
6. Import the data of your search collections into the target portal. For the import source information, use your documented file names and directory locations to which you exported the collections before the portal upgrade.

What to do next

Notes:

1. Additionally to exporting and importing your search collections, you must configure Portal Search on the target portal. This depends on the requirements of your target portal environment and configuration. For details about how to do this refer to the appropriate topics of the Portal Search documentation, such as Planning and preparing for Portal Search and Administering Portal Search.
2. Before you export a collection, make sure that the portal application process has write access to the target directory location. Otherwise, you might get an error message, such as `File not found`.
3. Import collection data only into an empty collection. Do not import collection data into a target collection that has content sources or documents already.
4. When you import search collection data into a collection, most of the collection configuration data are also imported. For example, this includes the content sources, schedulers, filters, and language settings. If you configured such settings when creating the new collection, they are overwritten by the imported settings.
5. If you want to migrate from one portal version to a higher version, you need to delete the search collections between the export and the reimport. Follow the steps that are described in the topic about Migrating web search collections.
6. When you import a portal site collection from a Version 5.1 portal to a Version 6 portal, the collection configuration data are imported, but not the documents. Therefore, to enable users to search the portal site collection on the target portal, you can either import the portal site collection and then start a crawl, or re-create the portal site collection. For details about how to do this see the topic abut Resetting the default search collection. This restriction does not apply if you migrate your portal site search collections between Version 6 portals.
7. When you import a collection, a background process fetches, crawls, and indexes all documents that are listed by URL in the previously exported file. Therefore, be aware that the crawling process can require extended memory and time, depending on your Portal Search configuration. For more information, see the topic about Hints and tips for Portal Search crawls.

Related concepts:

"Planning and preparing for Portal Search" on page 611
Learn about some planning considerations and first steps that you need to apply before working with Portal Search.

Related tasks:

"Administering Portal Search" on page 671
You can administer and configure many details for Portal Search.

"Resetting the default search collection" on page 711
Under certain circumstances, you might want to change the configuration of the portal site search collection. In this case, you must re-create the collection, as search collections cannot be modified.

"Migrating web search collections" on page 823
You migrate web search collections to IBM WebSphere Portal Express Version 8.5 by exporting each web search collection from the earlier portal and then importing the web search collection into the new portal. You can also use the same functionality to move web search collections to a production portal after verifying them on a test portal, or to move web search collections to a configuration with remote search after verifying them locally on a portal.

Related reference:

"Hints and tips for Portal Search crawls" on page 717
View some useful tips about crawls that Portal Search performs. For example, crawling can require extended memory and time, depending on your Portal Search environment and configuration.

Related information:

"" on page 667
Use Portal Search to facilitate indexing content sources and searching for information. You can administer search services, search collections, and search scopes, as well as enhance the search experience of your portal site with the portal search portlets.

Searching and crawling portal and other sites

Configure your local portal site, and crawl remote portal sites, so that they are searchable by users. Run crawlers against other, external Web sites to make them searchable by local portal users.

Users of your portal can search across various types of sites. In addition to searching the local portal site, you can crawl remote portal sites, and external Web sites, to make search results from those sites available to your local portal users. Examples of search scenarios include:

- Users of your portal search your own local portal site. This can include public and secure pages of your portal.
- Users of your portal search the Web Content Manager collection provided with the portal. This includes all Web Content Manager sites and libraries,
- Users of your portal site search other portal sites. This works only for public pages of the other portals.
- Users of your portal search external Web sites such as yahoo.com or google.com or cnn.com. When you run a crawler against external Web sites, you can collect and display external search results next to results from your local portal site.
- External users search your portal site. This works only for public pages of your portal.

"Resetting the default search collection"
Under certain circumstances, you might want to change the configuration of the portal site search collection. In this case, you must re-create the collection, as search collections cannot be modified.

"Crawling a remote portal site" on page 712
Configure Portal Search to crawl and index a remote, public portal site.

"Crawling an external site using a seedlist provider" on page 713
The seedlist crawler is a special HTTP crawler that can be used to crawl external sites which publish their content using the seedlist format. The seedlist format is an ATOM/XML-based format specifically for publishing application content, including all its metadata. The format supports publishing only updated content between crawling sessions for more effective crawling. You can configure the seedlist crawler with general parameters, filters and schedulers, then run the crawler.

Resetting the default search collection

Under certain circumstances, you might want to change the configuration of the portal site search collection. In this case, you must re-create the collection, as search collections cannot be modified.

About this task

The portal site default search collection is created at the first time when an administrator navigates to the search administration portlet Manage Search. This process requires considerations about the configuration tasks that are related to the portal and Portal Search. Consideration must also be given about the sequence by which you complete these tasks. An example scenario might be that you want to run a portal database transfer, for example, from the default database to a different database. In this case, you must create the portal site collection by navigating to the Manage Search portlet *before* you transfer the database. Otherwise, your portal site collection is not available after the database transfer.

If you created the portal site collection by navigating to the Manage Search portlet *before* you configured your portal and Portal Search, you might need to re-create the search collection. Example scenarios are as follows:

- If the preferred language for the portal site crawler user ID did not match the language of the portal site search collection.
- If you decide to change the default directory location for search collections in your portal installation. For information about how to do this configuration, see the topic about Configuring the default location for search collections.
- If the file path length for search collections exceeds its limit of 118 characters, the collection cannot be created. In this case specify a shorter value for the parameter **`DefaultCollectionsDirectory`**. For details about how to configure this parameter see the topic about Configuring the default location for search collections.

  This file path length problem can occur particularly when the portal site collection is created on one of the following operating systems:

  – Linux

  For details about this length limitation, see the topic about what to do if Creating the portal site search collection fails.
- If you want to turn the summarizer off so summary information is not generated for your portal and web content.
- If you want to change the name of the search collection.

Procedure

1. Complete the required configuration tasks, such as language or path settings
2. Create a search collection with the appropriate configuration settings.
3. Export the content sources from the default search collection. In a default portal installation, these sources are the Portal Content Source, which contains portal pages and portlets, and the Web Content Manager Content Source, which contains web content. For more information about exporting a search collection, see the section about *Exporting and importing search collections* in the related links or the Manage Search portlet help.
4. Import these exported content sources into your new search collection. For more information about importing a search collection, see the section about *Exporting and importing search collections* in the related links.
5. You can now delete the default search collection.

Results

Portal Search starts a new crawl on the portal site search collection.

Notes:

1. On a multilingual portal site, you can create multiple collections in different languages. For details, see the topic about Crawling a multilingual portal site.
2. When you start the crawl for the first time, a warning message might display. You can ignore this message. For more information, see the topic about Hints and tips for Portal Search crawls.

Related tasks:

"Configuring the default location for search collections" on page 682
You can modify the default directory location under which search collections are created on a per search service basis. View some related information.

"Crawling a multilingual portal site" on page 631
View the steps to set up search on a multilingual portal for users with different language preferences.

"Exporting and importing search collections" on page 707
View the steps to export search collections from a source portal and import them into a target portal.

Related reference:

"Hints and tips for Portal Search crawls" on page 717
View some useful tips about crawls that Portal Search performs. For example, crawling can require extended memory and time, depending on your Portal Search environment and configuration.

"Creating the portal site search collection can fail" on page 721
Creating the portal site search collection can fail due to a file path length restriction.

Related information:

Manage Search

Crawling a remote portal site

Configure Portal Search to crawl and index a remote, public portal site.

About this task

You can enable search on other portal sites. However, only the public pages of other portals can be searched.

To have Portal Search crawl and index a public portal site, proceed as follows:

1. Create a new content source using the Manage Search portlet.
2. Select Web site from the pull-down menu.
3. Enter the URL of the portal site that you want to make available for search by your users.

When you start the crawl, the public portion of the portal site is crawled. The search collection will only contain public pages.

Crawling an external site using a seedlist provider

The seedlist crawler is a special HTTP crawler that can be used to crawl external sites which publish their content using the seedlist format. The seedlist format is an ATOM/XML-based format specifically for publishing application content, including all its metadata. The format supports publishing only updated content between crawling sessions for more effective crawling. You can configure the seedlist crawler with general parameters, filters and schedulers, then run the crawler.

Before you begin

Before configuring the seedlist crawler, collect the following information:

- Root URL, which is the URL of the seedlist page.

  The seedlist page is a special ATOM/XML page containing metadata that directs the crawler to the actual links that should be fetched and indexed to become searchable later. The seedlist page also contains document level metadata that is stored along with the document in the search index. In order to make seedlist crawler results searchable, you must provide the crawler with a URL to a page containing a seedlist. The crawler retrieves the seedlist and crawls the pages indicated by the seedlist.
- User ID and Password, which are used by the crawler to authenticate the seedlist page.

About this task

To configure and create the seedlist crawler:

Procedure

1. Click **Manage Search** > **Search Services**.
2. Click the relevant Portal Search Service.
3. Click the name of an existing search collection, or create a new search collection.
4. Click **New Content Source**.
5. Click the drop-down menu icon next to **Content source type** and click **Seedlist provider** to indicate that the content source is a seedlist.
6. Under the tabs **General Parameters**, **Advanced parameters**, **Schedulers** and **Security**, provide the information in the fields and select options as required. For details refer to the topic *Managing and administering Portal Search*.
7. Click **Create**. This creates the new content source.

8. To run the crawler, click the start crawler icon for the content source on the Content Sources page. If you have defined a crawler schedule under the **Schedulers** tab, the crawler will start at the next possible time that you specified.

Related reference:

"Applying filter rules" on page 706
Portal Search provides a facility for applying filter rules to the crawler process. The crawler filters control the crawler progress and the type of documents that are indexed and cataloged.

Related information:

 Manage Search

Hints and tips for using Portal Search

View some useful tips for using Portal Search.

Content Model has only one search collection

Currently, the Content Model Search Service has only one search collection. This search collection is provided with the installation by default. You cannot modify this default Content Model search collection or create more search collections under the Content Model Search Service. The content model search service is listed because you can include it in scopes.

Using the Search Center with external search services with different languages

In order to use external search services such as Google and Yahoo! with an English search keyword, a URL such as the sample URL mentioned in the Search Center portlet help for configuring the portlet works fine as is: `http://www.google.com/search?q=` . However, if you search in other languages, consult the documentation of the remote search service that you use to ensure that the web interface is set up and used appropriately for the language that you use for your search. This can avoid problems with the displayed results, depending on the combination of languages set for WebSphere Portal Express, your browser, and the search.

Dynamic portal pages are not added to the search seedlist

Dynamic WebSphere Portal Express pages are not added to the Portal Search seedlist. Dynamic pages can have portlets, and portlets are added to the seedlist; therefore, if such pages are added to the seedlist, users get duplicate result list items. If you want to make HTML on a page searchable, create a static portal page and add the HTML. The static page is then added to the seedlist and listed among search results. For more detailed information about static and dynamic pages, see *Creating and adding static content* and *Dynamic user interfaces*.

Search collections unavailable in cluster if failover occurs

If a cluster member in a cluster fails, users who were using the affected cluster member when the failover occurred can no longer access search collections. This problem can occur with horizontal scaling when a node fails or with vertical scaling when a particular cluster member fails.

Users who are logged in to the cluster member that failed must log out of WebSphere Portal Express and then log back in before they are able to access search collections again.

Search can return documents based on metadata

Search can return documents that are based on metadata of these documents, not just on words that are found in the fields or actual text of the document. It might appear to Portal Search users that their searches return documents that do not appear to match the search criteria. Metadata for documents is also indexed for search. Therefore, if the metadata of documents matched the search criteria, these documents are also returned as results for the search. Metadata works as designed and is usually considered to be of benefit.

Documents from deleted content source can remain available under scope

If you delete a content source, then the documents that were collected from this content source will remain available for search by users under all scopes which included the content source before it was deleted. These documents will be available until their expiration time ends. When you create the content source, you can specify the expiration time by modifying the **Links expire after (days)** field in the General Parameters tab.

Documents from deleted web content library can remain available in the search collection

When you delete a web content library, you must also delete the corresponding entries from the search collection. When a web content library is deleted and crawled, deleting the corresponding crawler deletes the entries from the search index. If you use one Web Content Manager content source, which automatically crawls all web content libraries, then delete and re-create the content source, or you can select **Regather documents from Content Source**. This step deletes all existing documents in the content source from previous crawls and then starts a full crawl on the content source. Documents that were indexed before, but no longer exist in the content source are removed from the collection.

Note: If you plan on deleting web content libraries frequently, then it is suggested to define one content source per web content library. When that library is deleted, only the respective content source needs to be deleted as well.

Portal Search portlets are not compatible with WSRP

The Portal Search portlets cannot be provided as WSRP services, as some additional and more advanced WebSphere Portal Express concepts and features are not reflected by the current WSRP standard yet. These Portal Search portlets include Manage Search and the Search Center.

Default Portal Search Service and its collections show in the portal default language

The search administration portlet Manage Search lists the Default Portal Search Service and its collection Portal Content or other collections in the default portal language and not in the language that the user selected as preferred language for the portal or set in the browser. For example, if the portal default language is set to English and the user selected German as the preferred portal language or set the

browser language to German, the Default Portal Search Service and its collections show in English.

Virtual portals have separate search services and collections

Search services and search collections are separate for individual virtual portals and are not shared between individual virtual portals. Set up separate search services and separate search collections for each individual virtual portal. These collections can be used to crawl and search the same set of documents.

"Hints and tips for improving quality of Portal Search results"
There are three options available to improve the quality of search results and thus the overall search experience for your site visitors. The three options are using the Suggested Links portlet, changing the default query operator from `Or` to `And`, and applying boost factors to specific metadata fields.

"Hints and tips for Portal Search crawls" on page 717
View some useful tips about crawls that Portal Search performs. For example, crawling can require extended memory and time, depending on your Portal Search environment and configuration.

"How Portal Search handles special characters when indexing" on page 719
Portal Search indexes words that are composed of consecutive literals, that is letters, digits, and special characters. Learn how Portal Search handles special characters during indexing.

"Uninstalling WebSphere Portal Express does not delete search collections" on page 720
When you uninstall WebSphere Portal Express, the directories and files for the search collections are not deleted. Therefore, before you uninstall WebSphere Portal Express, delete all search collections by selecting the collections individually and clicking the option **Delete Collection**. If you do not delete, these files and directories remain on the hard disk drive. If you want to delete the search collection data after uninstalling WebSphere Portal Express, you need to delete manually.

"Linux operating systems might require higher limit of open files for Portal Search to work properly" on page 720
The limit for the number of open files in a Linux operating system might be too low for Portal Search to work properly. This might result in a `Portlet Unavailable` error.

"Creating the portal site search collection can fail" on page 721
Creating the portal site search collection can fail due to a file path length restriction.

"On IBM i set USER.REGION variable" on page 721
Under IBM i, Portal Search collections might fail to collect documents.

"Users cannot see portal site search results in their preferred language" on page 722
If the preferred language of the crawler user ID does not match the language of the search collection, users might not see search results in their language.

Hints and tips for improving quality of Portal Search results

There are three options available to improve the quality of search results and thus the overall search experience for your site visitors. The three options are using the Suggested Links portlet, changing the default query operator from `Or` to `And`, and applying boost factors to specific metadata fields.

Using the Suggested Links portlet

Use the Suggested Links portlet to ensure that a specific document is listed at the beginning of a set of search results. When a site visitor runs a search, the Suggested Links portlet displays the documents that are tagged with the same keywords from the search. To edit the keywords for Suggested Links, complete the following steps:

1. Log in to WebSphere Portal Express as an administrator and navigate to the Search Center.
2. Select **Edit keywords for Suggested Links** for the document that you would like to edit.
3. Specify what tags you would like to associate with that specific document.
4. Click **Save**.

The keywords that you specified are now associated with that item. If a site visitor searches for any of those keywords that are associated with that specific document, that document displays as a link in the Suggested Links portlet.

Changing the default search operator from "Or" to "And"

When a site visitor enters more than one search term, the Portal search engine applies a logical `Or` operator as a default. In order for a document to display in the search results list, the document must have only one of those search terms. However, search engines such as Google often use `And` as the default search operator. Using `And` as the default search operator means that all of the terms used in the search query must be found in each of the returned documents. For more details on changing the default search operator, see *Search service configuration parameters* in the related links.

Applying boost factors to specific metadata fields

When a site visitor runs a search, any values from metadata fields such as `title`, `description`, or `keywords` are automatically added to the generic `content` field in the search index. The values in these fields contribute equally to relevance calculation for any qualifying documents that are returned in the search results list. Use the search service configuration parameter **`boostingSettings`** to give extra weight to specified metadata fields, such as `title` or `description`. You can also use **`boostingSettings`** to specify how much extra weight specified metadata fields receive. For more information on the parameter **`boostingSettings`**, see *Search service configuration parameters* in the related links.

Related reference:

"Search service configuration parameters" on page 674
Learn about the portal search service parameters and possible values.

Related information:

Improving search quality in Portal Search

Hints and tips for Portal Search crawls

View some useful tips about crawls that Portal Search performs. For example, crawling can require extended memory and time, depending on your Portal Search environment and configuration.

HTTP crawler does not support JavaScript

The HTTP crawler of the Portal Search Service does not support JavaScript. Therefore some text of web documents might not be accessible for search by users. This depends on how the text is prepared for presentation in the browser. Specifically text that is generated by JavaScript might or might not be available for search.

Crawling a portal site for the first time can result in a message

When you start the crawl on a portal site for the first time, this can result in the following message:

```
EJPJP0009E: Wrong root url for Portal site crawler: https://root_url
```

You can ignore this message. The crawl runs correctly.

To resolve this problem, edit the content source, select the **General Parameters** tab, and the set the parameter **Stop fetching documents after (seconds):** to a value of `90` seconds.

Memory required for crawls

Depending on your Portal Search environment, crawling can require large amounts of memory. Therefore, before you start a crawl, make sure that WebSphere Portal Express has enough free memory. Memory shortage can cause a corrupted search collection and eventually lead to a system freeze.

To resolve this problem, raise the limit to the number of open files by using the `ulimit` command as root administrator.

Due to the resources needed for a crawl and index, it is recommended that you schedule crawls to occur when user activity is relatively low.

Time required for crawls and imports and availability of documents

The following search administration tasks can require extended periods of time:

- Crawling a content source. During the crawl documents might not be immediately available for searching or browsing.
- Indexing the documents fetched by a crawl. When a crawl has been complete and all documents have been collected, building the index takes some more time.
- Importing a search collection. When you import data to a collection, it can take some time until the content sources for the collection are shown in the Content Sources in Collection box and the documents of the imported collection are available for crawling.

These tasks are put in a queue. It might therefore take several minutes until they are executed and the respective time counters start, for example, the crawl **Run time** and the timeout for the crawl set by the option **Stop collecting after (minutes):** . The time required for these tasks is further influenced by the following factors:

- The number of documents in the content source that is being crawled
- The size of the documents in the content source that is being crawled

- Speed and availability of your processors, hard drive storage systems, and network connection.
- The value that you selected from the **Stop collecting after (minutes):** drop-down menu when you created or edited the content source.

Therefore both the time limits that you can specify and the times that are shown for these processes work as approximate time limits. This applies, for example, to the following scenarios:

- When you start a crawl by selecting a content source in the **Content Sources in Collection** box and clicking **Start collecting**.
- When you import a search collection and when you start a crawl on the imported search collection.
- When an installation is complete and you initialize the pre-configured portal site collection by selecting the portal site content source and clicking **Start collecting**.
- The time shown under **Last update completed** in the collection status information is later than you might assume by just adding the crawler time limit specified by **Stop collecting after (minutes):** to the crawling start time. This delay is caused by the additional time required by building the index.

Furthermore, this influences other status indicators given in the Manage Search portlet. For example, the number of documents shown for a content source can show with an unexpectedly low figure or even at zero (**0**) until the crawl on that content source has been completed.

How Portal Search handles special characters when indexing

Portal Search indexes words that are composed of consecutive literals, that is letters, digits, and special characters. Learn how Portal Search handles special characters during indexing.

This includes the following characters:

- The hash or pound sign (**#**).
- The percent sign (**%**).
- The plus sign (**+**).
- The asterisk (*****).

During indexing special characters are handled as follows:

Blank or white space; this includes the tab
Blanks separate words and are not indexed. Example: The string `key board` is indexed as two separate words `key` and `board` .

Line break or new line
Line breaks separate words and are not indexed unless they are preceded by a dash (**-**). Examples:

- The string

  ```
  key
  board
  ```

  is indexed as two separate words `key` and `board` .
- The string

  ```
  key-
  board
  ```

  is indexed as one word `keyboard`.

Dot or sentence end period (.) and comma (,)
Dots and commas separate words and are not indexed, unless they are both preceded *and* followed by a letter or digit. Example: The string `www.ibm.com` is indexed as `www.ibm.com` and not as three separate words.

Question mark (?) and exclamation mark (!)
Question marks and exclamation marks separate words and are not indexed unless they are followed by a letter.

Other punctuation: ( ) { } [ ] < > ; : / \ | " _ -
These characters separate words and are not indexed.

Other characters
All other characters are removed from the strings in which they appear but do not separate words.

Notes:

1. All characters that split words are discarded during indexing and searching.
2. The previous statements apply to **indexing**. However, in a **search query** all characters that can be part of the search syntax are treated in that capacity and not as part of the search query. These are the plus (+) and minus (-) signs, double quotation marks (`"`), and the asterisk wild card character (*). If users want to include such characters in their search query, they must enclose them in double quotation marks. For example `"+hello"` searches for the string `+hello`; `"*Hello*"` searches for the string `*Hello*`.
3. The less than (<) and greater than (>) symbols are special HTML characters that Search cannot handle.

Uninstalling WebSphere Portal Express does not delete search collections

When you uninstall WebSphere Portal Express, the directories and files for the search collections are not deleted. Therefore, before you uninstall WebSphere Portal Express, delete all search collections by selecting the collections individually and clicking the option **Delete Collection**. If you do not delete, these files and directories remain on the hard disk drive. If you want to delete the search collection data after uninstalling WebSphere Portal Express, you need to delete manually.

The directory path of a search collection is determined by what you typed in the field Location of Collection when you created the search collection. You can look up the collection location by doing the following steps:

1. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
2. Click **Search Collections**.
3. Select the collection from the **Search Collections** box.
4. The collection location is shown in the list under **Search collection status information**.

Linux operating systems might require higher limit of open files for Portal Search to work properly

The limit for the number of open files in a Linux operating system might be too low for Portal Search to work properly. This might result in a `Portlet Unavailable` error.

To resolve this problem and allow a higher number of files to be handled, raise the limit to the number of open files by issuing the following command as root administrator:

```
ulimit -n 4096
```

Creating the portal site search collection can fail

Creating the portal site search collection can fail due to a file path length restriction.

Problem: If the file path length for the location of search collections exceeds its limit, the collection cannot be created.

Cause: The file path length for the portal search collection is limited to 118 characters. If this limit is exceeded, the default collection cannot be created. The following items contribute to the length of the file path:

- The installation directory path.
- By default, the search collection for the portal site content is created under the path *your_portal_install_directory*/PortalServer/collections.
- The name of the virtual portal.
- The name of the search collection.

Solution: Complete the following steps to resolve the issue:

1. Change the default directory location for the portal site search collection to a shorter path. The complete path and file name must not exceed a length of 118 characters. For information, go to *Configuring the default location for search collections*.
2. Re-create the portal site search collection. For information, go to *Resetting the default search collection*.

Related tasks:

"Resetting the default search collection" on page 711
Under certain circumstances, you might want to change the configuration of the portal site search collection. In this case, you must re-create the collection, as search collections cannot be modified.

"Configuring the default location for search collections" on page 682
You can modify the default directory location under which search collections are created on a per search service basis. View some related information.

On IBM i set USER.REGION variable

Under IBM i, Portal Search collections might fail to collect documents.

In this case the logs will provide the following or similar information:

```
[8/24/08 23:19:47:164 EDT] 000000cd ServletWrappe E
         Uncaught init() exception thrown by servlet SearchSeedlistServletSecured
[8/24/08 23:19:47:175 EDT] 000000cd ServletWrappe E
         Deregister the mbean because of uncaught init() exception thrown by
         servlet SearchSeedlistServletSecured: javax.servlet.ServletException:
         Could not load resource bundle nls.SeedlistServletMessages using locale
         en_${USER.REGION} - Java Exception Message: java.util.MissingResourceException:
         Can't find bundle for base name nls.SeedlistServletMessages,
         locale en_${USER.REGION}
```

Solution: In order for portal collections to work on a IBM i system, set the system variable USER.REGION.

Users cannot see portal site search results in their preferred language

If the preferred language of the crawler user ID does not match the language of the search collection, users might not see search results in their language.

Therefore, set the preferred language of the portal site crawler user ID to match the language of the portal site search collection that it crawls. If you do this after you started a crawl on the portal site search collection, you need to reset the portal site collection. Refer to the topic about Resetting the default search collection.

If your portal site is multilingual and your users use different languages to search WebSphere Portal Express, set the portal site collections up as described under the topic about Crawling a multilingual portal site.

Related tasks:

"Resetting the default search collection" on page 711
Under certain circumstances, you might want to change the configuration of the portal site search collection. In this case, you must re-create the collection, as search collections cannot be modified.

"Crawling a multilingual portal site" on page 631
View the steps to set up search on a multilingual portal for users with different language preferences.

Setting limits on searches for users and groups

Searching for users or groups is a time consuming task. A search may time out or return more results than the system can handle or the user may expect. To prevent this behavior, you can set limits on searches for users or groups.

You can limit searches for users or groups in two ways:

- Setting a number of search results
- Setting a timeout for searches in the user repository

Set the **maxSearchResults** and **searchTimeOut** parameters in the following file:

Table 83. Location of the `wimconfig.xml` file

Operating system	Directory path
Windows:	*wp_profile_root*\config\cells*cell_name*\wim\config\wimconfig.xml
Linux:	*wp_profile_root*/config/cells/*cell_name*/wim/config/wimconfig.xml
IBM i:	*wp_profile_root*/profiles/config/cells/*cell_name*/wim/config/wimconfig.xml

The **maxSearchResults** parameter specifies the number of search results. The **searchTimeOut** parameter specifies the time out in milliseconds.

If you set **maxSearchResults**=200 and **searchTimeOut**=120000, it returns a number of 200 users or groups and terminates the search if the back end does not respond within two minutes. These settings affect the user or groups shown in portlets (for example the **User Manager** portlet) and XML export scripts.

If not all search results are returned, and you are using an LDAP server as the user repository, a `SizeLimitException` can be thrown instead of returning the items. The search result can exceed the limit defined in LDAP. Perform one of the following actions:

- Refine your search conditions to return fewer results.
- Change the setting in LDAP to allow a larger search size limit.

Important: If your user repository contains more users or groups than the value configured in **maxSearchResults**, a complete export of users and groups with XML access is not possible.

LDAP search filter expressions

The rules for rule-based user groups are based on the LDAP search filter syntax.

For information about the LDAP search filter syntax, see RFC2254 - The String Representation of LDAP Search Filters in the related links section.

You can use this subset of the LDAP search filter syntax:

- The `AND` operator represented by an ampersand (`&`).
- The `OR` operator represented by a vertical slash (`|`).
- The `NOT` operator represented by an exclamation mark (`!`).
- Equality comparison represented by an equal sign (`=`) for name and value expressions.
- Wildcards represented by an asterisk (`*`) at the beginning or end of values in name and value expressions.

Note: Attributes must not start with one of the operator symbols `AND`, `OR`, or `NOT` (`&`, `|`, or `!`), and they must not contain a comparison equal sign (`=`), or parentheses.

For example:

`(uid=testuser)`
: Matches to all users that have exactly the value `testuser` for the attribute `uid`.

`(uid=test*)`
: Matches to all users that have values for the attribute `uid` that start with `test`.

`(!(uid=test*))`
: Matches to all users that have values for the attribute `uid` that do not start with `test`.

`(&(department=1234)(city=Paris))`
: Matches to all users that have exactly the value `1234` for the attribute `department` and exactly the value `Paris` for the attribute `city` .

`(|(department=1234)(department=56*))`
: Matches to all users that have exactly the value `1234` or a value that starts with `56` for the attribute `department`.

`(&(department=12*)(!(department=123*)))`
: Matches to all users that have a value starting with `12`, but not starting with `123` for the attribute `department`.

Syntax validation

When you define or modify a rule base user group, the rule-based user groups adapter validates the syntax for the LDAP search filter expression. For example:

Invalid rule specified:
If you provide a rule that is not valid, rule-based user groups return the appropriate error message. However, it does not check whether the attribute names that you use exist in the user configuration. You can verify the configuration by using the code that calls the search filter.

Invalid attribute specified:
If an invalid attribute name is contained in a rule, the group membership determination for rule-based user groups does not work and logs an error. Existing rules might break if your attribute configuration in the system changes, for example, when an attribute is removed or renamed.

Related information:

 RFC2254 - The String Representation of LDAP Search Filters - http://www.faqs.org/rfcs/rfc2254.html

Creating the portal site search collection can fail

Creating the portal site search collection can fail due to a file path length restriction.

Problem: If the file path length for the location of search collections exceeds its limit, the collection cannot be created.

Cause: The file path length for the portal search collection is limited to 118 characters. If this limit is exceeded, the default collection cannot be created. The following items contribute to the length of the file path:

- The installation directory path.
- By default, the search collection for the portal site content is created under the path *your_portal_install_directory*/PortalServer/collections.
- The name of the virtual portal.
- The name of the search collection.

Solution: Complete the following steps to resolve the issue:

1. Change the default directory location for the portal site search collection to a shorter path. The complete path and file name must not exceed a length of 118 characters. For information, go to *Configuring the default location for search collections*.
2. Re-create the portal site search collection. For information, go to *Resetting the default search collection*.

Related tasks:

"Resetting the default search collection" on page 711
Under certain circumstances, you might want to change the configuration of the portal site search collection. In this case, you must re-create the collection, as search collections cannot be modified.

"Configuring the default location for search collections" on page 682
You can modify the default directory location under which search collections are created on a per search service basis. View some related information.

Portal Search trace and log files

Portal Search provides logging and tracing so that you can get additional information for resolving possible problems.

Portal Search has the following trace strings:

com.ibm.portal.search
Use to turn on **all** Portal Search messages.

com.ibm.portal.search.notIndexed
Enable to obtain messages about URLs that are discovered by the crawler but could not be fetched and indexed for different reasons.

com.ibm.portal.search.crawler
Enable to obtain messages about the crawling process.

com.ibm.portal.search.crawler.failure
Enable to obtain messages about failures that happen during a crawl.

com.ibm.portal.search.index
Enable to obtain messages about the indexing process.

com.ibm.portal.search.index.failure
Enable to obtain messages about failures that happen during the indexing process.

Enabling logging for a remote search configuration

If you have set up Portal Search in a remote configuration on a WebSphere Application Server server, the log messages are filtered and printed out according to the WebSphere Application Server logging and tracing configuration. In this case refer to the WebSphere Application Server information center.

Related concepts:

"Logging and tracing" on page 3541
If you are experiencing a problem, you might want to enable tracing and then re-create the problem to capture more log information. You can enable logging and tracing for software that is included with WebSphere Portal Express. Enabling tracing makes log output more verbose. For example, you can enable tracing within WebSphere Application Server to obtain information about application servers and other processes.

Chapter 6. Document Conversion Services

Document Conversion Services are used when you work with the Common Mail Portlet, IBM Web Content Manager authoring and previewing, and search.

About this task

Documents that are produced by many standard applications (such as word processors or spreadsheet editors) can be viewed as HTML pages with Document Conversion Services. Documents that are received as attachments to email can be viewed in the browser even if the application that created the document is not installed. Document Conversion Services also allows documents to be searched by content. Document pages can also be converted into an image format such as GIF, BMP, JPG, or PNG.

"Configure Document Conversion Services"
Learn how to configure the Document Conversion Services in WebSphere Portal. A knowledge of these prerequisites steps can assist you in preventing, identifying, and correcting problems that are related to Document Conversion Services.

"Supported operating systems for document conversion services" on page 731
Some operating systems support running the DCS locally while others do not. The following table lists the operating systems and versions that support running the document conversion services. On operating systems that do not support running the DCS locally, document conversion must be done on a remote IBM WebSphere Application Server that supports document conversion services.

"Setting up file type definitions to enable Document Conversion Services" on page 731
Set up file definition types to ensure that document conversions works for Microsoft Office, Lotus® SmartSuite®, and OpenOffice file types.

"Using Document Conversion Services with Stellent" on page 734
For Linux, when using Stellent version 8.1.0 and later, the X-server dependency has been removed.

"Configuring a remote Document Conversion Service" on page 734
To balance processing power, you can run document conversion services on a remote IBM WebSphere Application Server that supports document conversion services. Enabling the delegation of document conversion services to a remote server requires steps on both IBM WebSphere Portal Express and the remote server.

"Files conversion supported by DCS" on page 737
View a list of file types that are supported by DCS for conversion.

Configure Document Conversion Services

Learn how to configure the Document Conversion Services in WebSphere Portal. A knowledge of these prerequisites steps can assist you in preventing, identifying, and correcting problems that are related to Document Conversion Services.

Visual C++ libraries are required for Windows

To run Document Conversion Services on Windows, download Visual C++ libraries included in the latest Visual C++ Redistributable Package available from the Microsoft download site. There are three versions of this package, x86, x64, and IA64 for the 86-bit, 64-bit, and 64-bit Itanium systems.

You must search the Microsoft download site for the `vcredist_x86.exe`, `vcredist_x64.exe`, or `vcredist_IA64.exe` files.

Exporter task

Run the **exporter** task from the *wp_profile_root*`/PortalServer/config/oiexport/exporter` directory to ensure that you installed all the libraries.

A successful task displays the following message:

```
Error: no input file was specified
Error: no output file was specified
Error: No output id was specified
```

"Configuring Document Conversion Services for systems other than Windows" Configure the Document Conversion Services to complete document conversions in an operating system other than Windows. You must complete the following steps whenever you start the Portal server from a new terminal window.

"Configuring Document Conversion Services for IBM i" on page 729 Configure the Document Conversion Services to perform document conversions on IBM i systems. You need to complete the following steps whenever you start the Portal server from a new terminal window.

"Configuring images for Document Conversion Services" on page 729 Configure the Document Conversion Services to view images in an operating system other than Windows.

Configuring Document Conversion Services for systems other than Windows

Configure the Document Conversion Services to complete document conversions in an operating system other than Windows. You must complete the following steps whenever you start the Portal server from a new terminal window.

Procedure

1. Ensure that the `oiexport` directory is in the PATH as *wp_profile_root*`/PortalServer/config/oiexport:$PATH`.

   For example, `export PATH=`*wp_profile_root*`/PortalServer/config/oiexport:`*AppServer_root*`/java/jre/bin:$PATH`
2. Export `LD_LIBRARY_PATH/LIBPATH/SHLIB_PATH` to point to the `oiexport` directory and the graphics library directory. This environment variable is different for different systems, as detailed in the following table:

Table 84. Library path variables by operating system

Platform	Library path variable
Linux	LD_LIBRARY_PATH

Required library: On Linux, you must install the following library to run document conversion services: `libstdc++.so.5`. This library is included with GNU Compiler Collection (GCC) 3.2 to 3.3.6.

You must install `libstdc++.so.5` and include the folder that contains the library in the library path variable. The command might be `export LD_LIBRARY_PATH=`*`wp_profile_root`*`/PortalServer/config/oiexport:/usr/X11R6/lib:/user/lib`

3. Run the **exporter** task from the *wp_profile_root*`/PortalServer/config` directory to ensure that you installed all the libraries.

   A successful task displays the following message:

   ```
   Error: no input file was specified
   Error: no output file was specified
   Error: No output id was specified
   ```

Configuring Document Conversion Services for IBM i

Configure the Document Conversion Services to perform document conversions on IBM i systems. You need to complete the following steps whenever you start the Portal server from a new terminal window.

Procedure

1. Log in to the WebSphere Integrated Solutions Console.
2. Select **Servers** > **Server Types** > **WebSphere application servers**
3. Click the **WebSphere_Portal** server to open the configuration window.
4. Select **Server Infrastructure** > **Java and Process Management** > **Process definition**.
5. Select **Additional Properties** > **Environment Entries**.
6. Click **LIBPATH** to open the configuration window.
7. In the **Value** field, add the following information: `:/QOpenSys/usr/bin:/QOpenSys/usr/bin/X11:/QOpenSys/usr/sbin`.
8. Click **Apply** and save your changes to the master configuration. The value for the **LIBPATH** environment entry displays as follows: `${WPS_HOME}/lwo/prereq.odc/shared/app/oiexport:/QOpenSys/usr/bin:/QOpenSys/usr/bin/X11:/QOpenSys/usr/sbin`
9. Restart WebSphere Application Server.

Configuring images for Document Conversion Services

Configure the Document Conversion Services to view images in an operating system other than Windows.

About this task

The following steps do not apply to IBM i. For information about viewing images on IBM i, read *Rendering documents on IBM i*.

Procedure

1. Connect to one of the following workstations from your Windows workstation with PuTTY or any other Telnet client.
   - Linux
2. The `oiexport` directory should be in the PATH as *wp_profile_root*`/PortalServer/config/oiexport:$PATH`.

For example, export PATH = *wp_profile_root*/PortalServer/config/oiexport:*AppServer_root*/java/jre/bin:$PATH

3. Export LD_LIBRARY_PATH /LIBPATH/ SHLIB_PATH to point to the oiexport directory and the graphics library directory. This environment variable is different for different operating systems, as detailed in the following table:

Table 85. Library path variables by operating system.

Platform	Library path variable
Linux	LD_LIBRARY_PATH

For example, in case of Linux, the command might be export LD_LIBRARY_PATH = *wp_profile_root*/PortalServer/config/oiexport:/usr/X11R6/lib

4. Now connect to another workstation through XBrowser.
5. Start the XBrowser and enter the host name. When you log in successfully, type echo $DISPLAY at the prompt. Which gives you <machine-ip>:0.0,, where <machine-ip> refers to the ip of the workstation from where XBrowser was started.
6. In the same Telnet/ssh session where you exported PATH and LD_LIBRARY_PATH/LIBPATH/SHLIB_PATH, type the command export DISPLAY=<machine-ip>:0.0. This value is the same as the value you get in the previous step.
7. Change directory to the location where the xclock and xhost files are present, such as, /usr/X/bin.
8. Type the command ./xclock &.
9. Type ./xhost + on the workstation through which you are connected through XBrowser. Which gives you the message access control disabled, clients can connect from any host.

   Note: Changing directory is required because by entering the commands xclock& or xhost+, the error message command not found is displayed.
10. The Xclock menu comes up on the desktop of the workstation to which you are connected through X-Browser.
11. Change directory to *AppServer_root*/bin.
12. Restart the server.
13. Open a web browser and try viewing the file that contains the images.

Starting the X server by using Exceed or X virtual frame buffer (Xvfb)

The previous steps are given to start the X server by using XBrowser. The following steps are used to start the X server by using Exceed or X virtual frame buffer (Xvfb).

Procedure

1. Start Exceed.
2. Connect to one of the following workstations with PuTTY and then type export PATH.
   - Linux
3. Type export LD_LIBRARY_PATH/LIBPATH/SHLIB_PATH.
4. Type export DISPLAY.

5. Type `echo $DISPLAY` on the workstation to which you connected by using Exceed or Xvbf.
6. Enter `xclock &` (if `xclock` displays on the desktop of the workstation to which you are connected through Exceed or Xvfb, then X server is running).
7. If `xhost + {xhost + <IP>}` displays, this means conversion works for this particular IP and only + means for all workstations.

   You must start WebSphere Portal Express from the console or an X Server-enabled client with the same privileges (xhost+) as the console. The graphical conversions require access to an X Window System server, since they require access to the Xm, Xt and X11 libraries. Also, the DISPLAY environment variable must be set to the account that Portal is running under. The DISPLAY must be valid at the point that Portal is started. Otherwise, telnetting to a workstation and starting Portal from there does not work. You must start Portal from an X terminal.

Supported operating systems for document conversion services

Some operating systems support running the DCS locally while others do not. The following table lists the operating systems and versions that support running the document conversion services. On operating systems that do not support running the DCS locally, document conversion must be done on a remote IBM WebSphere Application Server that supports document conversion services.

The following operating systems are supported by DCS.

Table 86. Supported Platforms

Platform	Version
IBM AIX® 64 bit	6.1, 7.1
Linux X86-64	RHLE 5, SLES 10, 11
Solaris SPARC	v10

Setting up file type definitions to enable Document Conversion Services

Set up file definition types to ensure that document conversions works for Microsoft Office, Lotus SmartSuite, and OpenOffice file types.

Procedure

1. Locate the `content-types.properties` file in the following directory:
   - , , Linux, , Windows: *AppServer_root*`/java/jre/lib`
   - IBM i:
     - JDK 7: `/QOpenSys/QIBM/ProdData/JavaVM/jdk70/64bit/jre/lib`
     - JDK 7.1: `/QOpenSys/QIBM/ProdData/JavaVM/jdk71/64bit/jre/lib`
2. Update the `content-types.properties` file as follows. These extensions are case-sensitive.

   Microsoft Office 2007

```
application/vnd.openxmlformats-officedocument.wordprocessingml.document: \
   description=Microsoft Word 2007 file;\
   file_extensions=.docx

application/vnd.openxmlformats-officedocument.presentationml.presentation: \
   description=Microsoft PowerPoint 2007 file;\
```

```
   file_extensions=.pptx

application/vnd.openxmlformats-officedocument.spreadsheetml.sheet: \
   description=Microsoft Excel 2007 file;\
   file_extensions=.xlsx
```

Microsoft Office (before 2007)

```
application/msword: \
 description=Microsoft Word;\
 file_extensions=.doc

application/vnd.ms-excel: \
 description=Microsoft Excel;\
 file_extensions=.xls

application/vnd.ms-powerpoint: \
 description=Microsoft PowerPoint;\
 file_extensions=.ppt
```

Lotus SmartSuite

```
application/vnd.lotus-freelance: \
        description=Lotus Freelance;\
        file_extensions=.prz

application/vnd.lotus-1-2-3: \
        description=Lotus 1-2-3;\
        file_extensions=.123

application/vnd.lotus-wordpro: \
        description=Lotus WordPro;\
        file_extensions=.lwp
```

OpenOffice

```
application/vnd.sun.xml.writer: \
 description=Open Office;\
 file_extensions=.sxw

application/vnd.sun.xml.writer.template: \
 description=Open Office;\
 file_extensions=.stw

application/vnd.sun.xml.writer.global: \
 description=Open Office;\
 file_extensions=.sxg

application/vnd.sun.xml.calc: \
 description=Open Office;\
 file_extensions=.sxc

application/vnd.sun.xml.calc.template: \
 description=Open Office;\
 file_extensions=.stc

application/vnd.sun.xml.impress: \
 description=Open Office;\
 file_extensions=.sxi

application/vnd.sun.xml.impress.template: \
 description=Open Office;\
 file_extensions=.sti

application/vnd.sun.xml.draw: \
 description=Open Office;\
 file_extensions=.sxd

application/vnd.sun.xml.draw.template: \
 description=Open Office;\
```

```
 file_extensions=.std

application/vnd.sun.xml.math: \
 description=Open Office;\
 file_extensions=.sxm
```

3. Install the appropriate package for your system.

Table 87. Systems and packages

Systems	Extra Packages
IBM i (v7r1)	X11R6
Linux	X11R6 LessTif or Motif with libXm.so.1 (Stellent 8.0.1) LessTif or Motif with libXm.so.2 (Stellent 7.75 or earlier)
Windows	n/a

If OpenMotif is included on the Linux server, do not need to install the LessTif Rational Portfolio Manager package. The binary files for the LessTif package are already included.

Example

A temporary directory is used for document conversion.

- Windows: `c:\temp`
- , , Linux, : `/tmp`

  Note: The default `/tmp` directory is in the root system. If it does not exist, it must be created.
- IBM i: `/temp`

To create a temporary directory other than the default, edit the `convertors.xml` file in *wp_profile_root*`/PortalServer/dcs`. Add the property **tempDir** to the `<global>` tag as follows:

```
<global>
<property name="tempDir" value="yourtempDirectory"/>
</global>
```

What to do next

To start the native application when you click the attachment, the following MIME types must be registered in WebSphere Application Server to enable browsers to properly display documents of that type. The following extra types must be registered:

- `application/msword doc`
- `application/vnd.ms-excel xls`

For , , Linux, , you must start IBM Workplace Services Express from the console or an X-Server-enabled client with the same privileges (**xhost+**) as the console.

Note: Graphical conversions require access to an X Windows server. They require access to the **Xm**, **Xt**, and **X11** libraries. Also, the DISPLAY environment variable must be set to the account that Portal is running under. The DISPLAY must be valid at the point that Portal is started. Therefore, telnetting to a server and starting Portal from there does not work. You must start the portal from an X terminal.

For IBM i, refer to the prerequisites as described in "Configuring images for Document Conversion Services" on page 729.

Using Document Conversion Services with Stellent

For Linux, when using Stellent version 8.1.0 and later, the X-server dependency has been removed.

About this task

Procedure

1. Prepare the system using the steps outlined in "Configuring images for Document Conversion Services" on page 729.
2. Install **libXm.so.3** and **ln -sf /usr/X11R6/lib/libXm.so.3.0.2 /usr/X11R6/lib/libXm.so.2.0.1**.
3. Use the command **find / -name *.ttf** to locate the fonts folder on your machine. Set **GDFONTPATH** to point to the fonts folder. For example:

   ```
   export GDFONTPATH=/opt/IBM/WebSphere/AppServer/java/jre/lib/fonts
   ```

4. Make sure that the environment variable **GDFONTPATH** includes one or more paths to the fonts folder.

   Only TrueType fonts (*.ttf or *.ttc files) are supported. If the variable GDFONTPATH is not found, the current directory is used. If fonts are called for and cannot be found, HTML Export will exit with an error. When copying Windows fonts to a Unix system, the font extension for the files (*.ttf or *.ttc) must be lowercase, or the fonts will not be detected during the search for available fonts. Stellent does not provide fonts with any Outside In product.

What to do next

On platforms that support local document conversion, document conversion services can be delegated to a remote server to better balance processing power and manage server performance.

IBM i: You can choose to run the X Server on the portal server or a remote server. You can also use a frame-buffer device or XVFB to emulate X Windows servers on a headless server. The conversion filters are installed with WebSphere Portal Express. There is no need for configuration. However, you can configure the `export.cfg` file, located under the *wp_profile_root*`/PortalServer/config` directory, if you want to fine tune conversion properties such as graphics size and resolution.

Configuring a remote Document Conversion Service

To balance processing power, you can run document conversion services on a remote IBM WebSphere Application Server that supports document conversion services. Enabling the delegation of document conversion services to a remote server requires steps on both IBM WebSphere Portal Express and the remote server.

Before you begin

- If the remote Document Conversion Service (DCS) was set up on a remote machine with the remote search installer, then skip steps 1 - 7.
- Update the `content-types.properties` file to ensure that document conversions are configured for different file types. For instructions, read *Enabling Document Conversion Services*.

- Copy the RemoteDCS.zip file to your file system. Locate the RemoteDCS.zip file in your WebSphere Portal Express installation directory under *PortalServer_root*/lwo/prereq.odc/RemoteSearchInstaller.

  Note: Updated versions of RemoteDCS.zip might be available online. Check the *Recommended fixes and updates for WebSphere Portal and Web Content Management* site.

About this task

Procedure

1. Install WebSphere Application Server on the remote system.

   Restriction: Stellent version 8.01 must support the remote system.
2. Copy RemoteDCS.zip to the remote server and extract it in any directory with an appropriate archiving tool. A directory that is named dcs becomes available. This directory contains several files and subdirectories.
3. Deploy dcs.war to WebSphere Application Server.
   a. Log in to the WebSphere Integrated Solutions Console.
   b. Select **Applications** > **Install New Application**.
   c. Browse to the directory where you extracted RemoteDCS.zip then locate and select dcs.war.
   d. Specify dcs as the context root and click **Next**.
   e. Leave all other values as default.

      Restriction: The application name must be dcs_war. Do not change the default name.
   f. Map the application to the appropriate servers and specify the installation options as required.
   g. Click **Finish** and save your changes to the master configuration.
4. Ensure dcs.war is running.
   a. Log in to the WebSphere Integrated Solutions Console.
   b. Select **Applications** > **Enterprise Applications**.
   c. Locate **dcs_war** in the list of installed applications.
   d. Select **dcs_war** and click **Start** if the application is not started.
5. Run the configuration script.
   a. Open a command prompt.
   b. Change to the dcs directory. The configuration script is in the root of the dcs directory that is in the directory where you extracted RemoteDCS.zip.
   c. Run the following command:

      Linux : ./setupremotedcs.sh

      IBM i: setupremotedcs.sh

      Windows: setupdcs.bat
   d. When prompted, specify the WebSphere Application Server installation directory. For example, *AppServer_root*.
   e. When prompted for the profile, specify the directory where dcs.war is installed. For example, *AppServer_root*/profiles/*profile_name*/installedApps/*node_name*/dcs_war.ear

The configuration script copies all the files to the oiexport directory where you installed dcs.war; for example, *AppServer_root*/profiles/*profile_name*/installedApps/*node_name*/dcs_war.ear/dcs.war/WEB-INF/lib/oiexport

6. Complete the following steps to add environment variables to your remote search server:
   a. Log on to the WebSphere Integrated Solutions Console.
   b. Go to **Servers** > **Server Types** > **WebSphere application servers**.
   c. Select your remote DCS server.
   d. Go to **Java and Process Management** > **Process definition**.
   e. Click **Environment Entries**.
   f. Click **New** to create the following entries:

      Note: If the entries exist, click the entry and update the value.

      LD_LIBRARY_PATH
      AppServer_root/profiles/*profile_name*/installedApps/*node_name*/dcs_war.ear/dcs.war/WEB-INF/lib/oiexport

      LIBPATH
      AppServer_root/profiles/*profile_name*/installedApps/*node_name*/dcs_war.ear/dcs.war/WEB-INF/lib/oiexport

      PATH
      AppServer_root/profiles/*profile_name*/installedApps/*node_name*/dcs_war.ear/dcs.war/WEB-INF/lib/oiexport

      SHLIB_PATH
      AppServer_root/profiles/*profile_name*/installedApps/*node_name*/dcs_war.ear/dcs.war/WEB-INF/lib/oiexport

   g. Save your changes.
7. Restart the server. Document Conversion Services is now installed on the remote server.
8. Complete the following steps:
   a. Open wkplc.properties with any text editor from the *wp_profile_root*/ConfigEngine/properties directory.
   b. Add the following property and value to wkplc.properties: DcsRemoteHost=*URL of the remote host*.

      Where *URL of the remote host* is the URL of the remote server to which you plan to delegate document conversion services. For example, DcsRemoteHost=http://*server_name*:*port_number*/dcs/dcs. The *server_name* is the fully qualified host name of the remote DCS server. The *port_number* is the port number of the default host where DCS is run. For example, the URL might be **DcsRemoteHost=http://example.mycompany.com:9080/dcs/dcs**.

      Important: This property does not exist in the wkplc.properties file by default. You must add this property and specify a value before you proceed to the next step.
   c. Run the appropriate task to delegate some or all of the document conversion functions to the remote server.
      1) Open a command prompt.
      2) Change to the following directory:

         Linux : *wp_profile_root*/ConfigEngine

         IBM i: *wp_profile_root*/ConfigEngine

Windows: *wp_profile_root*\ConfigEngine

3) Run the following task:

Note: Run the following command on all nodes in a cluster, if applicable.

Linux : ./ConfigEngine.sh *task_name*

IBM i: ConfigEngine.sh *task_name*

Windows: ConfigEngine.bat *task_name*

Where *task_name* is one of the following configuration tasks:

Table 88. Tasks to delegate document conversion functions to a remote server.

Task	Description
delegate-all-conversions	This task delegates all file conversion services to a remote server.
delegate-text-conversions	This task delegates text document conversions to a remote server.
remove-conversion-delegation	This task ends the delegation of all file conversion services to a remote server.

d. Restart WebSphere Portal Express.

Files conversion supported by DCS

View a list of file types that are supported by DCS for conversion.

IBM WebSphere Portal Express has a feature that allows you to view documents as HTML files. This feature is useful if you do not have access to the application used to create the file. The following table lists file types that are supported by DCS for file conversion.

Table 89. Archive file types that are supported by DCS

Archive	Version
7z (BZIP2 and split archives not supported)	
7z Self Extracting exe (BZIP2 and split archives not supported)	
LZA Self Extracting Compress	
LZH Compress	
Microsoft Office Binder	95 – 97
Microsoft Cabinet (CAB)	
RAR	1.5, 2.0, 2.9
Self-extracting.exe	
UNIX Compress	
UNIX GZip	
UNIX tar	
Uuencode	
Zip	PKZip
Zip	WinZip

Table 90. Database file types that are supported by DCS

Database	Version
DataEase	4.x
DBase	III, IV, V
First Choice DB	Through 3.0
Framework DB	3.0
Microsoft Access (text only)	1.0, 2.0, 95 – 2010
Microsoft Access Report Snapshot (File ID only)	2000 – 2003
Microsoft Works DB for DOS	2.0
Microsoft Works DB for Macintosh	2.0
Microsoft Works DB for Windows	3.0, 4.0
Microsoft Works DB for DOS	1.0
Paradox for DOS	2.0 – 4.0
Paradox for Windows	1.0
Q&A Database	Through 2.0
R:Base	R:Base 5000
R:Base	R:Base System V
Reflex	2.0
SmartWare II DB	1.02

Table 91. Email file types that are supported by DCS

Email	Version
Apple Mail Message (EMLX)	2.0
Encoded mail messages	MHT
Encoded mail messages	Multi Part Alternative
Encoded mail messages	Multi Part Digest
Encoded mail messages	Multi Part Mixed
Encoded mail messages	Multi Part News Group
Encoded mail messages	Multi Part Signed
Encoded mail messages	TNEF
EML with Digital Signature	SMIME
IBM Lotus Notes Domino XML Language DXL	8.5
IBM Lotus Notes NSF (File ID)	7.x, 8.x
IBM Lotus Notes NSF (Win32, Win64, Linux x86-32 and Oracle Solaris 32-bit only with Notes® Client or Domino Server)	8.x
MBOX Mailbox	RFC 822
Microsoft Outlook (MSG)	97 – 2013

Table 91. Email file types that are supported by DCS (continued)

Email	Version
Microsoft Outlook Express (EML)	
Microsoft Outlook Forms Template (OFT)	97 – 2013
Microsoft Outlook OST	97 – 2013
Microsoft Outlook PST	97 – 2013
Microsoft Outlook PST (Mac)	2001
MSG with Digital Signature	SMIME

Table 92. Multimedia file types that are supported by DCS

Multimedia	Version
AVI (Metadata only	
DICOM (File ID only)	
Flash (text extraction only)	6.x, 7.x, Lite
Flash (File ID only)	9, 10
MP3 (ID3 metadata only)	
MPEG-1 Audio layer 3 V ID3 v1 (Metadata only)	
MPEG-1 Audio layer 3 V ID3 v2 (Metadata only)	
MPEG-1 Video V 2 (File ID only)	
MPEG-1 Video V 3 (File ID only)	
MPEG-2 Audio (File ID only)	
MPEG-4 (Metadata only)	
MPEG-7 (Metadata only)	
QuickTime (Metadata only)	
Windows Media ASF (Metadata only)	
Windows Media DVR-MS (Metadata only)	
Windows Media Audio WMA (Metadata only)	
Windows Media Playlist (File ID only)	
Windows Media Video WMV (Metadata only)	
WAV (Metadata only)	

Table 93. Other file types that are supported by DCS

Other	Version
AOL Messenger (File ID only)	7.3
Microsoft InfoPath (File ID only)	2007
Microsoft Live Messenger (via XML filter)	10.0
Microsoft Office Theme files (File ID only)	2007 – 2013
Microsoft OneNote (text only)	2007, 2010
Microsoft Project (table view only)	98 – 2003
Microsoft Project (table view only)	2007, 2010
Microsoft Windows Compiled Help (File ID only)	.chm
Microsoft Windows DLL	

Table 93. Other file types that are supported by DCS (continued)

Other	Version
Microsoft Windows Executable	
Microsoft Windows Explorer Command (File ID only)	.scf
Microsoft Windows Help (File ID only)	.hlp
Microsoft Windows Shortcut (File ID only)	.lnk
Trillian Text Log File (via text filter)	4.2
Trillian XML Log File (File ID only)	4.2
TrueType Font (File ID only)	ttf, ttc
vCalendar	2.1
vCard	2.1
Yahoo! Messenger	6.x – 8

Table 94. Presentation file types that are supported by DCS

Presentation	Version
Apple iWork Keynote (MacOS, text, and PDF preview)	09
Harvard Graphics Presentation DOS	3.0
IBM Lotus Symphony® Presentations	1.x
Kingsoft WPS Presentation	2010
Lotus Freelance	1.0 – Millennium 9.8
Lotus Freelance for OS/3	2
Lotus Freelance for Windows	95, 97, SmartSuite 9.8
Microsoft PowerPoint for Macintosh	4.0 – 2011
Microsoft PowerPoint for Windows	3.0 – 2013
Microsoft PowerPoint for Windows Slideshow	2007 – 2013
Microsoft PowerPoint for Windows Template	2007 – 2013
Novell Presentations	3.0, 7.0
OpenOffice Impress	1.1, 3.0
Oracle Open Office Impress	3.x
StarOffice Impress	5.2 – 9.0
Wordperfect Presentations	5.1 – X5

Table 95. Raster Image file types that are supported by DCS

Raster Image	Version
Adobe Photoshop	4.0
Adobe Photoshop PSD (File ID only)	
Adobe Photoshop	CS1 – 6
CALS Raster (GP4)	Type I
CALS Raster (GP4)	Type II

Table 95. Raster Image file types that are supported by DCS (continued)

Raster Image	Version
Computter Graphics Metafile	ANSI
Computer Graphics Metafile	CALS
Computer Graphics Metafile	NIST
Encapsulated PostScript (EPS)	TIFF header Only
GEM Image (Bitmap)	
Graphics Interchange Format (GIF)	
IBM Graphics Data Format (GDF)	1.0
IBM Picture Interchange Format	1.0
JBIG2	Graphic Embeddings in PD
JFIF (JPEG not in TIFF format)	
JPEG	
JPEG 2000	JP2
Kodak Flash Pix	
Kodak Photo CD	1.0
Lotus PIC	
Lotus Snapshot	
Macintosh PICT	BMP only
Macintosh PICT 2	BMP only
MacPaint	
Microsoft Windows Bitmap	
Microsoft Windows Cursor	
Microsoft Windows Icon	
OS/2 Bitmap	
OS/2 Warp Bitmap	
Paint Shop Pro (Win32 only)	5.0, 6.0
PC Paintbrush (PCX)	
Portable Bitmap (PBM)	
Portable Graymap PGM	
Portable Network Graphics (PNG)	
Portable Pixmap (PPM)	
Progressive JPEG	
StarOffice Draw	6.x – 9.0
Sun Raster	
TIFF	Group 5 & 6
TIFF CCITT	Group 3 & 4
TruVision TGA (Targa)	2.0
Word Perfect Graphics	1.0
WBMP wireless graphics format	

Table 95. Raster Image file types that are supported by DCS (continued)

Raster Image	Version
X-Windows Bitmap	x10 compatible
X-Windows Dump	x10 compatible
X-Windows Pixmap	x10 compatible
WordPerfect Graphics	2.0 – 10.0

Table 96. Spreadsheet file types that are supported by DCS

Spreadsheet	Version
Apple iWork Numbers (MacOS, text, and PDF preview)	09
Enable Spreadsheet	3.0 – 4.5
First Choice SS	Through 3.0
Framework SS	3.0
IBM Lotus Symphony Spreadsheets	1.x
Kingsoft WPS Spreadsheets	2010
Lotus 1-2-3®	Through Millennium 9.8
Lotus 1-2-3 Charts (DOS and Windows)	Through 5.0
Lotus 1-2-3 for OS/2	2.0
Microsoft Excel Charts	2.x – 2007
Microsoft Excel for Macintosh	98 – 2011
Microsoft Excel for Windows	3.0 – 2013
Microsoft Excel for Windows (text only)	2003 XML
Microsoft Excel for Windows (.xlsb)	2007 – 2013 (Binary)
Microsoft Works SS for DOS	2.0
Microsoft Works SS for Macintosh	2.0
Microsoft Works SS for Windows	3.0, 4.0
Multiplan	4.0
Novell PerfectWorks Spreadsheet	2.0
OpenOffice Calc	1.1 –3.0
Oracle Open Office Calc	3.x
PFS: Plan	1.0
QuattroPro for DOS	Through 5.0
QuattroPro for Windows	Through X5
SmartWare Spreadsheet	
SmartWare II SS	1.02
StarOffice Calc	5.2 – 9.0
SuperCalc	5.0
Symphony	Through 2.0

Table 96. Spreadsheet file types that are supported by DCS (continued)

Spreadsheet	Version
VP -Planner	1.0

Table 97. Text & Markup file types that are supported by DCS

Text & Markup	Version
ANSI Text	7 & 8 bit
ASCII Text	7 & 8 bit
DOS character set	
EBCDIC	
HTML (CSS rendering not supported)	1.0 – 4.0
IBM DCA/RFT	
Macintosh character set	
Rich Text Format (RTF)	
Unicode Text	3.0, 4.0
UTF-8	
Wireless Markup Language	
XML (text only)	
XHTML (file ID only)	1.0

Table 98. Vector Image file types that are supported by DCS

Vector Image	Version
Adobe Illustrator	4.0 – 7.0
Adobe Illustrator (PDF Preview only)	9.0, CSI – 6
Adobe Illustrator XMP	CSI – 6
Adobe InDesign XMP	CSI – 6
Adobe InDesign Interchange XMP only	
Adobe PDF	1.0 – 1.7 (Acrobat 1–10)
Adobe PDF Package	1.7 (Acrobat 8 – 10)
Adobe PDF Portfolio	1.7 (Acrobat 8 – 10)
Ami Draw	SDW
AutoCAD Drawing	2.5, 2.6
AutoCAD Drawing	9.0 – 14.0
AutoCAD Drawing	2000i – 2012
AutoShade Rendering	2
Corel Draw	2.0 – 9.0
Corel Draw Clipart	5.0, 7.0
Enhanced Metafile (EMF)	
Escher graphics	

Table 98. Vector Image file types that are supported by DCS (continued)

Vector Image	Version
FrameMaker Graphics (FMV)	3.0 – 5.0
Gem File (Vector)	
Harvard Graphics Chart DOS	2.0 – 3.0
Harvard Graphics for Windows	
HP Graphics Language	2.0
IGES Drawing	5.1 – 5.3
Micrografx Designer	Through 3.1
Micrografx Designer	6.0
Micrografx Draw	Through 4.0
Microsoft XPS (Text only)	
Novell PerfectWorks Draw	2
OpenOffice Draw	1.1 – 3.0
Oracle Open Office Draw	3.x
SVG (processed as XML, not rendered)	
Visio (Page Preview mode WMF/EMF)	4.0
Visio	5.0 – 2007
Visio XML VSX (File ID only)	2007
Windows Metafile	

Table 99. Word-Processing file types that are supported by DCS

Word Processing	Version
Adobe FrameMaker (MIF only)	3.0 – 6.0
Adobe Illustrator Postscript	Level 2
Ami	
Ami Pro for OS2	
Ami Pro for Windows	2.0, 3.0
Apple iWork pages (MacOS, text, and PDF preview)	09
DEC DX	Through 4.0
DEC DX Plus	4.0, 4.1
Enable Word Processor	3.0 – 4.5
First Choice WP	1.0, 3.0
Framework WP	3.0
Hangul	97 – 2010
IBM DCA/FFT	2.0 – 5.0
IBM DisplayWrite®	1.01
IBM Writing Assistant	
Ichitaro	5.0, 6.0, 8.0 – 13.0, 2004, 2010
JustWrite	Through 3.0
Kingsoft WPS Writer	2010

Table 99. Word-Processing file types that are supported by DCS (continued)

Word Processing	Version
Legacy	1.1
Lotus Manuscript	Through 2.0
Lotus Word Pro (text only)	9.7, 96 – Millennium 9.8
MacWrite II	1.1
Mass 11	Through 8.0
Microsoft Publisher (File ID only)	2003 – 2007
Microsoft Word for DOS	4.0 – 6.0
Microsoft Word for Macintosh	4.0 – 6.0, 98 – 2011
Microsoft Word for Windows	1.0 – 2013
Microsoft Word for Windows (text only)	2003 XML
Microsoft Word for Windows	98-J
Microsoft WordPad	
Microsoft Works WP for DOS	2.0
Microsoft Works WP for Macintosh	2.0
Microsoft Works WP for Windows	3.0, 4.0
Microsoft Write for Windows	1.0 – 3.0
MultiMate	Through 4.0
MultiMate Advantage	2.0
Navy DIF	
Nota Bene	3.0
Novell PerfectWorks Word Processor	2.0
OfficeWriter	4.0 – 6.0
OpenOffice Writer	1.1 – 3.0
Oracle Open Office Writer	3.x
PC File Doc	5.0
PFS: Write	A, B
Professional Write for DOS	1.0, 2.0
Professional Write Plus for Windows	1.0
Q&A Write	2.0, 3.0
Samna Word IV	1.0 – 3.0
Samna Word IV+	
Samsung jungUm Global (File ID only)	
Signature	1.0
SmartWare II WP	1.02
Sprint	1.0
StarOffice Writer	5.2 – 9.0
Total Word	1.2
Wang IWP	Through 2.6

Table 99. Word-Processing file types that are supported by DCS (continued)

Word Processing	**Version**
WordMarc Composer	
WordMarc Composer+	
WordMarc Word Processor	
WordPerfect for DOS	4.2
WordPerfect for Macintosh	1.02 – 3.1
WordPerfect for Windows	5.1 – X5
Wordstar 2000 for DOS	1.0- 3.0
Wordstar 2000 for DOS	2.0, 3.0
Wordstar for DOS	3.0 – 7.0
Wordstar for Windows	1.0
XyWrite	Through III+

Chapter 7. IBM Connections

IBM Connections portlets give the IBM WebSphere Portal Express users access to more collaboration and social networking features such as activities, blogs, and bookmarks.

"Configuring Portal to work with IBM Connections"
For WebSphere Portal Express to work with IBM Connections you must configure the portal Ajax proxy, and set up single sign-on.

"Configuring IBM Connections to work with your portal" on page 748
Configure IBM Connections to work with your WebSphere Portal Express by Importing the SSL certificate.

"Configuring the IBM Connections portlets" on page 750
Install and configure the IBM Connections Portlets for WebSphere Portal Express.

"Configuring IBM Connections features" on page 750
Configure the features that site visitors need to use from your portal site.

"Configuring community pages" on page 758
Configure how you want community pages access to work in your portal site. For example you can configure community pages to automatically grant access to the page, how many child pages affected by community associations, and more.

CF03 "Configuring Portal to work with Connections in SmartCloud for Social Business" on page 762
The Connections integration assets that are used to integrate WebSphere Portal Express with on-premise Connections can be used to integrate WebSphere Portal Express with IBM Connections in SmartCloud for Social Business as well.

Related information:

IBM Connections: Installing IBM Connections

IBM Connections Portlets for WebSphere Portal

IBM Connections: Configuring search integration

Configuring Portal to work with IBM Connections

For WebSphere Portal Express to work with IBM Connections you must configure the portal Ajax proxy, and set up single sign-on.

"Set up Ajax proxy" on page 748
The support for community pages uses the Ajax proxy to access the remote server. The Ajax proxy is updated for the base Connections URLS during the Connections Portlets installation. If FileNet is used, you must still configure the Ajax proxy manually to update for FileNet. Configure the Ajax proxy so that direct requests that the CCM portlet makes to the FileNet server are allowed to pass through the proxy Server.

"Set up single sign-on" on page 748
Site visitors want to sign in one time. Set up single sign-on to ensure that visitors receive only one authentication challenge. The instructions for deploying the IBM Connections portlets include multiple options for configuring authentication for the portlets. If you did not complete the

procedures in the IBM Connections documentation, you must configure single sign-on using the applications server. For best results, follow the procedures that are provided in the IBM Connections documentation.

Set up Ajax proxy

The support for community pages uses the Ajax proxy to access the remote server. The Ajax proxy is updated for the base Connections URLS during the Connections Portlets installation. If FileNet is used, you must still configure the Ajax proxy manually to update for FileNet. Configure the Ajax proxy so that direct requests that the CCM portlet makes to the FileNet server are allowed to pass through the proxy Server.

Related information:

IBM Connections: Deploying the IBM Connections portlets

Configuring the Ajax proxy to allow FileNet requests

Set up single sign-on

Site visitors want to sign in one time. Set up single sign-on to ensure that visitors receive only one authentication challenge. The instructions for deploying the IBM Connections portlets include multiple options for configuring authentication for the portlets. If you did not complete the procedures in the IBM Connections documentation, you must configure single sign-on using the applications server. For best results, follow the procedures that are provided in the IBM Connections documentation.

Before you begin

About this task

You can configure single sign-on using IBM WebSphere Application Server or by using IBM Security Access Manager. This procedure uses WebSphere Application Server. Instructions for using Security Access Manager are included in the IBM Connections documentation.

Procedure

1. Retrieve the LTPA token from the WebSphere Application Server where your portal is installed.
2. Import the LTPA token into the IBM Connections server.
3. Enable single sign-on on the Connections server.

Related information:

IBM Connections: Deploying the IBM Connections portlets

IBM Connections: Configuring authentication for the portlets

WebSphere Application Server:Single sign-on for authentication

Configuring IBM Connections to work with your portal

Configure IBM Connections to work with your WebSphere Portal Express by Importing the SSL certificate.

"Import SSL certificate to set up trust association" on page 749
To prevent security alert pop-up windows, add Connections certificates to your

portal server. The signer certificate that is used by the Connections server must be trusted by the portal server. The instructions for deploying the IBM Connections portlets include a procedure to import the SSL certificate. If you did not complete that procedure, you must import the certificate to use community pages. For best results, follow the procedures that are provided in the IBM Connections documentation.

Import SSL certificate to set up trust association

To prevent security alert pop-up windows, add Connections certificates to your portal server. The signer certificate that is used by the Connections server must be trusted by the portal server. The instructions for deploying the IBM Connections portlets include a procedure to import the SSL certificate. If you did not complete that procedure, you must import the certificate to use community pages. For best results, follow the procedures that are provided in the IBM Connections documentation.

Before you begin

About this task

Retrieve the certificate for the Connections server. Then, import the SSL certificate to the portal server.

Procedure

1. Enter the URL for the Connections server into a web browser. For example: `https://your_Lotus_Connections_server.com/activities`
2. Save the certificate as a PEM certificate. Save it to the portal server in the directory where IBM WebSphere Application Server was installed. For example: *AppServer_root*
3. Import the saved certificate to the portal server.
   a. Open the WebSphere Integrated Solutions Console.
   b. Select **Security** > **SSL certificate** > **key management** > **Key stores** > **certificates**.
   c. Click **NodeDefaultTrustStore**.
   d. Select **Signer certificates** and click **Add**.
   e. On the Add signer certificate page, provide a name for the certificate that you added, and enter the full qualified name of the saved certificate.
4. Import the saved certificate to the portal server.

Related tasks:

CF03 "Configuring single sign-on (SSO) for backend calls to IBM Connections in SmartCloud for Social Business" on page 765
The Connections integration assets use a common infrastructure that is called HTTP Outbound to complete calls to the Connections backend server. Learn about the steps that are required to configure the HTTP outbound component to complete calls to IBM Connections in SmartCloud for Social Business.

Related information:

IBM Connections: Deploying the IBM Connections portlets

IBM Connections: Importing a certificate to support SSL

Configuring the IBM Connections portlets

Install and configure the IBM Connections Portlets for WebSphere Portal Express.

About this task

To install the IBM Connections by using the Portal Solutions Installer follow the installation procedure provided in the IBM Connections Documentation.

Documentation resource:

Installing the IBM Connections Portlets for WebSphere Portal

"Configuring common directory service"
IBM Connections use the common Directory Services to enable directory lookup from Connections in the WebSphere Portal Express environment

"Configuring authentication for the portlets"
Configure authentication to enable access to the portlets.

Configuring common directory service

IBM Connections use the common Directory Services to enable directory lookup from Connections in the WebSphere Portal Express environment

About this task

To configure the common Directory Services to work with your security configuration, follow the steps that are provided in the Connections Document.

Documentation resource:

Configuring common directory services for your security configuration

Configuring authentication for the portlets

Configure authentication to enable access to the portlets.

About this task

To configure authentication for the portlets, follow the steps that are provided in the connections documentation.

Documentation resource:

Configuring authentication for the portlets

Configuring IBM Connections features

Configure the features that site visitors need to use from your portal site.

"Integrating Connections profile " on page 751
You can enable the Connections profile in WebSphere Portal Express so that users can view Connections business card information and link to features such as communities, blogs, and activities.

"Integrating IBM Connections tags" on page 752
You can integrate tags from IBM Connections into the IBM WebSphere Portal Express tag cloud. The Connections portlets are required to get full function.

"Integrating IBM Connections files" on page 755
To integrate files with your portal, register the Content Management Interoperability Services (CMIS) entry points with the WP FederatedDocumentsService resource environment provider. After this

configuration is complete, content authors insert links to documents hosted on the Connections server. Using personalization, authors can also render inventories of folders that are hosted in files.

Integrating Connections profile

You can enable the Connections profile in WebSphere Portal Express so that users can view Connections business card information and link to features such as communities, blogs, and activities.

Before you begin

About this task

Note: You must enable **Show Email** on the Connections server to ensure that portal users can view profile information.

Procedure

1. Search the Connections LDAP directory for the user whose profile you want to enable in WebSphere Portal Express and then add that user to the portal LDAP directory.
2. Ensure that `ibm-primaryEmail` is mapped to the appropriate attribute in the LDAP and that the email address in the Connections LDAP matches the email address in the portal LDAP. For example, `mail` or `email`
3. Register the Connections server. For stand-alone portal, complete the steps on the node. For a clustered environment, complete the steps on the WebSphere Application Server Network Deployment.
   a. Open the WebSphere Integrated Solutions Console.
   b. Click **Resources** > **URL** > **URL Providers**.
   c. Select the **Default URL Provider** at the beginning of the hierarchy.
   d. Under **Additional Properties**, select **URLs**.
   e. Click **New** and then specify these settings under General Properties:

      Name `CONNECTIONS_PEOPLE_CARD`

      JNDI name
      `JNDI_CONNECTIONS_PEOPLE_CARD`

      Specification
      Card for Connections version 2.5, or earlier:`https://`*`yourConnectionsServer.domain.com`*`/profiles/html/businessCard`

      For example: `https://w3.ibm.com/`*`connections`*`/profiles/html/`*`businessCard`*

      Card for later versions of Connections (post version 2.5):
      `https://`*`yourConnectionsServer.domain.com:portno`*
      For example: `https://w3.ibm.com/`*`connections`*
      The url you type fetches the JavaScript file `file/profiles/portalJS/portalBizCard.js`.. You do not have to type in the entire url to the JavaScript file, only type in the Connections https address. If you are using an HTTP server in front of the Connections environment, the port number can be omitted.

      Category
      `CATEGORY_CONNECTIONS_PEOPLE_CARD`
   f. Click **Save** to save the changes to the master configuration.

4. Restart the WebSphere_Portal server.

Results

After you enable the Connections profile for a particular user, when you move the cursor over that person's active (underlined) name in WebSphere Portal Express, and then click **Click for Person Card**, you see the Connections profile for that person. The profile is displayed in the business card section of the Person card, with links to more Connections features. Click **Profile** to see the user's full Connections profile. To return to the WebSphere Portal Express page, click **Back** in the browser.

Note: If you integrate Connections and then select a user who does not have a Connections profile, portal displays the message `profile does not exist`. Click **Back** to return to WebSphere Portal Express.

Related tasks:

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

Related information:

IBM Connections: Deploying the IBM Connections portlets

Integrating IBM Connections tags

You can integrate tags from IBM Connections into the IBM WebSphere Portal Express tag cloud. The Connections portlets are required to get full function.

"Configure authentication" on page 753
Portal requests data from the IBM Connections server through the Connections search API. By default, access to public data is not secured. In case security is enabled for that application, authentication is required. You can use on of the following authentication methods: basic authentication and LTPA forwarding.

"Configuring task to retrieve tags" on page 753
When you integrate Connections with your portal server, the portal uses a task to retrieve tags and related resources from the various Connections features (such as activities, blogs, bookmarks, communities, files, forums, profiles, and wikis). Then, the portal integrates the tags in the portal tag cloud. You can schedule the task to retrieve the tags to run periodically.

Results

Connections tags are available in theWebSphere Portal Express tag cloud after the scheduled task runs to retrieve tags from the Connections server.

Related information:

IBM Connections: Installing the IBM Connections Portlets for IBM WebSphere Portal

IBM Connections: Configuring and wiring the IBM Connections Tags portlet

IBM Connections: Deploying the IBM Connections Portlets for a Connections server

Configure authentication

Portal requests data from the IBM Connections server through the Connections search API. By default, access to public data is not secured. In case security is enabled for that application, authentication is required. You can use on of the following authentication methods: basic authentication and LTPA forwarding.

About this task

This procedure helps you set up authentication by using a shared slot in the portal credential vault. You need an Connections user ID to configure the credential vault. The user ID does not need to belong to a real user. It can be an ID that is only used for integration. This user ID needs to be able to authenticate to the Connections server. No other access rights are required.

Configure authentication for either basic authentication or LTPA forwarding.

Procedure

Option 1: Basic authentication

1. If basic authentication is used, specify the credentials for the user ID in the portal credential vault.
   a. Click the **Administration menu** icon. Then, click **Access** > **Credential Vault**.
   b. Add a vault slot for the following name:
   `com.ibm.wps.cp.tagging.federation.credentialSlot.search.basic_auth`
   c. Select a shared vault slot, and enter the user ID and password of the Connections user.
   d. Specify an existing resource or specify a new one.

Option 2: LTPA forwarding

2. If LTPA forwarding is used, specify the full distinguished name (DN) of a user ID in the portal credential vault.
   a. Click the **Administration menu** icon. Then, click **Access** > **Credential Vault**.
   b. Add a vault slot for the following name:
   `com.ibm.wps.cp.tagging.federation.credentialSlot.search.ltpa`
   c. Select a shared vault slot, and enter the DN of the Connections user ID.
   d. Enter an arbitrary password. The password is not necessary to create the LTPA token, but the credential vault portlet requires a password for a vault slot.
   e. Specify an existing resource or specify a new one.
3. Restart the WebSphere_Portal server.

Related tasks:

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

Configuring task to retrieve tags

When you integrate Connections with your portal server, the portal uses a task to retrieve tags and related resources from the various Connections features (such as activities, blogs, bookmarks, communities, files, forums, profiles, and wikis). Then, the portal integrates the tags in the portal tag cloud. You can schedule the task to retrieve the tags to run periodically.

Before you begin

The piece of content (POC) resolver is required to render tagged Connections resources within Connections portlets. The instructions for installing the Connections portlets included a step to copy the POC resolver to the portal.

About this task

Depending on your database system, it might be beneficial to reorganize some tables in the community domain. You can also gather current catalog statistics for the tables after an import of Connections data is done. This action helps the SQL optimizer determine an efficient access path. Review your database system documentation for instructions on how to accomplish this task. The database tables that are most affected by an import include:

CP_CUSTOMRESOURCE

CP_CATEGORY

CP_RESOURCE_TAG

LOC_DATA

LOC_DATA_LOD

You can schedule the FederationTaskHandler to periodically retrieve (import) Connections data. For more information about scheduling the FederationTaskHandler, see *Administering tag federation*.

Procedure

Using XML access, run
com.ibm.wps.cp.tagging.federation.taskhandler.FederationTaskHandler

Results

During the import of the tags if you exceed the number of available connections, increase the value for the configuration parameter `max-connections-per-host` for the corresponding policy in the Ajax Proxy configuration file `proxy-config.xml`

Related tasks:

"Administering tag federation" on page 1326
When tags from remote systems, such as IBM Connections are integrated into your WebSphere Portal Express site, you need to schedule tasks to retrieve tags and related data from the remote system, and later to clean them up from the portal. You can also redirect the rendering of federated resources to IBM Connections and add icons to federated resources.

Related reference:

"CP Configuration Service for tagging and rating" on page 312
The CP Configuration Service provides the properties for tagging and rating.

"Using the XML configuration interface to administer tags and ratings" on page 1343
You can use the XML configuration interface to manage tagging and rating in the portal. For example, you can move tagspaces and ratings between portal versions or for staging purposes.

Related information:

IBM Connections: Installing the IBM Connections Portlets for IBM WebSphere Portal

Integrating IBM Connections files

To integrate files with your portal, register the Content Management Interoperability Services (CMIS) entry points with the WP FederatedDocumentsService resource environment provider. After this configuration is complete, content authors insert links to documents hosted on the Connections server. Using personalization, authors can also render inventories of folders that are hosted in files.

About this task

Authors can use the generic CMIS file picker dialog, in the IBM Web Content Manager rich text editor, to insert links to documents hosted on Connections files. Authors can also use the personalization component in Web Content Manager to render inventories of folders on the Connections server

"Determining the URL of the CMIS service document"
Connections Files offers a set of CMIS service documents that provide specific views of the content that is stored in files. Determine which CMIS API service document URL you need before starting the configuration.

"Set up predefined IBM Connection servers" on page 756
Configuring integration with files is similar to configuring other federated documents. You must specify the remote server URL, a display name, and the supported interface type of the remote server.

"Configure federated documents feature for files" on page 757
Complete the configuration of the federated documents feature to communicate with Connections.

Determining the URL of the CMIS service document

Connections Files offers a set of CMIS service documents that provide specific views of the content that is stored in files. Determine which CMIS API service document URL you need before starting the configuration.

About this task

The set of CMIS service documents includes service documents that define the following things:

- User-specific view for the currently logged in user
- Views for individual specific communities

Procedure

Determine the service document URL.
The CMIS service document location can be determined based on the following pattern: *base URL of files service*`/basic/cmis/`*CMIS Service Doc identifier*, where:

The *base URL of files service* denotes the entry URL to your Connections files service. The files service URL is typically the Connections base URL concatenated with the files service context root. The default context root is `/files` .

The *CMIS Service Doc identifier* identifies a specific CMIS service document. Information about individual Connections CMIS service document identifiers is available in the file CMIS API documentation.

For example, for an Connections server with the URL `www.example.com:9444` and that uses the default context root, the CMIS service document location is `https://www.example.com:9444/files/basic/cmis/my/servicedoc`

Related information:

Files CMIS API: Retrieving the Files CMIS API service document

Set up predefined IBM Connection servers

Configuring integration with files is similar to configuring other federated documents. You must specify the remote server URL, a display name, and the supported interface type of the remote server.

About this task

To specify the values, you must add custom properties. Each property includes a *suffix*. The value of the suffix is used to group related properties for each server. Use the same suffix value for properties that are related to the same server. The suffix can be any value if it is unique across the property keys.

Procedure

1. Log on to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Click **WP FederatedDocumentsService**.
4. Under **Additional Properties**, click **Custom Properties**.
5. Register the CMIS service document with the federated documents feature.
   a. Click **New** and add this property and value:
   `wp.federated.documents.ic_personalized_sc.url=`*`https://www.example.com:9444/files/basic/cmis/my/servicedoc`*

   Replace *https://www.example.com:9444/files/basic/cmis/my/servicedoc* with your service document location.
   b. Click **New** and add this property and value:
   `wp.federated.documents.ic_personalized_sc.title.default=`*`Your Documents on IBM Connections`*
6. Specify the interface type.
   a. Click **New** and add this property: `wp.federated.documents.`*`suffix`*`.type`
   b. For the value, specify: `CMIS`

   You can also add more properties, such as:

   wp.federated.documents.ic_personalized_sc.nls.resources
   Value: The name of the resource bundle that contains the translated title and description used to identify this source server in the user interface. If this property is not defined, the default title is used. If no default title and no resource bundle are defined, the value of the **`wp.federated.documents.ic_personalized_sc.url`** property is used in the user interface.

   wp.federated.documents.ic_personalized_sc.vault.slot
   Value: The name of the credential vault slot that stores the credentials used for authentication with the remote server. Credential vault slots are set up and managed by the portal administrator. This property defines the default credential vault slot that is predefined in the user interface, although the user can also select a different slot if one is available. If this property is not defined, the user interface does not display a default credential vault slot, but you can still select a slot from the available list. This property is optional.

Note: The credential vault slot must contain the credentials that are required for authentication with the remote server.

wp.federated.documents.ic_personalized_sc.override.authentication.enabled
Value: `true` or `false`. When set to `true`, the user can change the authentication method for the server in the user interface. When set to `false`, the user interface does not display the field to change the authentication method. The default value is `true`.

Configure federated documents feature for files

Complete the configuration of the federated documents feature to communicate with Connections.

Procedure

1. Specify whether credential vaults slots are used for authentication with remote servers.

   Because you can access federated documents through either the personalization editor or the rich text editor that is provided with Web Content Manager, you can configure credential vault slots for each method independently.

   a. If you are accessing federated documents through the personalization editor, click **wp.federated.documents.pzn.vaultselection.enabled**. To enable credential vaults slots, set the value to `true`, or, to disable credential vault slots, set the value to `false`. By default, the value is `true`.

   b. If you are accessing federated documents through the rich text editor in Web Content Manager, click **wp.federated.documents.wcm.vaultselection.enabled**. To enable credential vaults slots, set the value to `true`, or, to disable credential vault slots, set the value to `false`. By default, the value is `true`.

   If you enable credential vault slots, users can select a credential vault slot in the user interface. You can also use the property **wp.federated.documents.*suffix*.vault.slot** to specify a default credential slot to be used with a given remote server.

2. If you enable credential vault slots, grant access to credential vault slots for all authenticated users.

   a. Log in to the portal as an administrator.

   b. Click the **Administration menu** icon. Then, click **Access** > **Resource Permissions**.

   c. From the list of resource types, click **Virtual Resources**.

   d. For the ADMIN_SLOTS resource, click the **Assign Access** icon.

   e. Edit the User role, and add the All Authenticated Portal Users group to the role.

3. Optional: Configure the amount of data that is returned for the summary metadata attribute of the document.

   a. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.

   b. Click **WCM WCMConfigService**.

   c. Under **Additional Properties**, click **Custom Properties**.

   d. Click **wcm.pzn.ecm.max.field.length**, and enter the number of characters to be returned. If no value is specified, the default value is 128 characters.

4. Optional: Configure whether property changes are automatically loaded.

By default, the Federated Documents service automatically reloads properties at a specified interval, without requiring you to restart the portal. You can change the automatic reloading behavior or modify the reloading interval.

a. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
b. Click **WP FederatedDocumentsService**.
c. Under **Additional Properties**, click **Custom Properties**.
d. Click **wp.federated.documents.document.service.reload.disabled**, and specify a value of `true` to disable automatic reloading of properties. The default value is `false`.
e. Click **wp.federated.documents.document.service.reload.interval**, and specify the interval in seconds for reloading properties. The default value is 3 seconds.

5. Save your changes. The Federated Documents service automatically reloads any updated properties. If you disabled automatic reloading, restart the portal server.

Related tasks:

Inserting a link to remote content
You can insert links to remote content into elements that contain a rich text field by using the **Insert Link to Remote Document** button in the rich text editor. Only remote content that is configured remote server access can be selected by using this button.

"Personalizing federated documents" on page 1827
Portal Personalization provides the federated documents feature to retrieve metadata about documents that are stored in external content management systems or document repositories. Examples of these systems include IBM Content Manager, IBM FileNet® Content Manager, and Microsoft Sharepoint. You can use a personalization component in IBM Web Content Manager to display metadata from federated documents and to create links to download or open the documents.

Configuring community pages

Configure how you want community pages access to work in your portal site. For example you can configure community pages to automatically grant access to the page, how many child pages affected by community associations, and more.

About this task

Procedures in this section are not required. Complete only the configuration procedures that you need.

"Automatically grant page access to community members" on page 759
If you want community members to automatically be able to access the page, without explicitly configuring access, you must enable that feature. Community membership must be integrated with portal security before you can enable this feature.

"Overriding access control integration during community page instantiation" on page 760
The **Restrict view access to this page to community members** setting on a community page automatically grants access to the page to members of the associated community. When specified on a page template, the setting is also applied to any pages that are created from the template. However, you can override this setting on a page template by defining a page parameter.

"Configure limits for propagation of community associations" on page 760
You can control how many nested child pages that are affected by community associations and changed community associations.

"Configuring the number of retrieved communities" on page 761
You can define how many communities are retrieved from the Connections server when searching for communities in the Page Associations window.

Related information:

Single sign-on for authentication

Automatically grant page access to community members

If you want community members to automatically be able to access the page, without explicitly configuring access, you must enable that feature. Community membership must be integrated with portal security before you can enable this feature.

Before you begin

About this task

After this feature is enabled, you see a **Restrict view access to this page to community members** check box on the Page Associations window. If you selected the check box, community members have access to the page. This access is in addition to any access that you explicitly grant to the page. Community member access to the parent page is not automatically granted to the community page.

Procedure

1. From the portal server, log in to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Click **WP ConnectionsIntegrationService**.
4. Under **Additional Properties**, click **Custom Properties**.
5. Update the value for the **community.group.mapper** property to the string `default`.
6. When integrating community membership and portal security, you can define the default access level that is granted to users.
   a. Edit the **community.member.access.level** property.
   b. Specify the role that is used to determine access. The default value is `Privileged User`.

      Depending on this role type, the set of role types that are blocked for inheritance on the page is defined as the set of all role types that are fully implied by the role type. For example, with the Privileged User role, the implied role types are Privileged User and User.

      For a list of available roles and information about the role types, see *Roles.*
7. Save your changes.
8. Restart the WebSphere_Portal server.

Results

The **Restrict view access to this page to community members** check box is available on the Page Association window.

Related concepts:

"Roles" on page 1533
Roles provide task permissions for users on resources. For example, Editor is a role that allows users to view, modify, and create resources. Roles are denoted as `Role@Resource`; for example, `Editor@Portal Page`.

Related information:


Integrating community membership with Portal security

Overriding access control integration during community page instantiation

The **Restrict view access to this page to community members** setting on a community page automatically grants access to the page to members of the associated community. When specified on a page template, the setting is also applied to any pages that are created from the template. However, you can override this setting on a page template by defining a page parameter.

Procedure

To override the **Restrict view access to this page to community members** setting for pages created from the template, set the **`ibm.portal.instantiation.community.access.control.integration`** parameter on the template.
You can set this parameter by editing the page properties in the user interface or by using the XML configuration interface (**`xmlaccess`** command).

Important: When specified, this parameter defines the behavior of the **Restrict view access to this page to community members** setting for pages that are created from the template. The corresponding setting in the Page Associations window for the page template is disregarded.
Specify one of the following values:

`true` The **Restrict view access to this page to community members** setting is enabled for all pages that are created from this template.

`false` The **Restrict view access to this page to community members** setting is disabled for all pages that are created from this template.

Configure limits for propagation of community associations

You can control how many nested child pages that are affected by community associations and changed community associations.

About this task

During this procedure modify the following properties in the WP_ConnectionsIntegrationService resource environment provider:

`child.page.propagation.levels`
The maximum number of page nesting levels that are affected when propagating a community association to child pages. The default value is `1`, which indicates that the association is propagated only to direct child pages of the current page.

To include all nested page levels, regardless of the number of levels, set the property value to `-1`.

To disable the propagation of a community association to child pages, set the property to `0`.

child.page.propagation.threshold

This threshold is based on how many child pages are affected by the propagation of a changed community association. If too many pages are updated, the operation might take too long and fail to complete. By setting this threshold, you can specify whether the control to copy the updated association is displayed in the user interface. If the number of pages to be updated is less than this threshold, the control is displayed. If the number of affected pages exceeds the threshold, the control is not displayed. The default value for this property is `50`.

The number of pages evaluated for this threshold also depends on the value of the **child.page.propagation.levels** property.

Procedure

1. On the portal server, log in to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Click **WP ConnectionsIntegrationService**.
4. Under **Additional Properties**, click **Custom Properties**.
5. Update the values for the **child.page.propagation.levels** property and the **child.page.propagation.threshold** property.
6. Save the property changes.

Configuring the number of retrieved communities

You can define how many communities are retrieved from the Connections server when searching for communities in the Page Associations window.

About this task

During this procedure, modify the following property in the WP_ConnectionsIntegrationService resource environment provider:

community.picker.page.size

This value defines the maximum number of community entries to retrieve from the Connections server when searching for communities in the Page Associations window. The default value is `50`.

The value of this property also determines the number of communities that are retrieved the first time that the Page Associations window is used.

Performance note: If you increase this value too much, the response times of the Connections server can become slow.

Procedure

1. On the portal server, log in to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Click **WP ConnectionsIntegrationService**.
4. Under **Additional Properties**, click **Custom Properties**.
5. Update the value for the **community.picker.page.size** property.
6. Save the property changes.

Configuring Portal to work with Connections in SmartCloud for Social Business

The Connections integration assets that are used to integrate WebSphere Portal Express with on-premise Connections can be used to integrate WebSphere Portal Express with IBM Connections in SmartCloud for Social Business as well.

CF03 "Known limitations for integrating WebSphere Portal Express with IBM Connections in SmartCloud for Social Business"
Some features that are available when you integrate WebSphere Portal Express with an on-premise IBM Connections server are not supported when you integrate WebSphere Portal Express with an Connections server that runs in the SmartCloud for Social Business.

CF03 "Establishing single sign-on (SSO) between the portal installation and IBM Connections in SmartCloud for Social Business" on page 763
Learn about the requirements that must be fulfilled in order to successfully integrate WebSphere Portal Express with IBM Connections in SmartCloud for Social Business. You can provide a user experience where a user logs in one time into WebSphere Portal Express and then automatically sees all integrated Connections information.

CF03 "Configuring a connection between WebSphere Portal Express and IBM Connections in SmartCloud for Social Business" on page 769
Learn how to configure the Connections integration assets to complete calls to IBM Connections in SmartCloud for Social Business.

Known limitations for integrating WebSphere Portal Express with IBM Connections in SmartCloud for Social Business

Some features that are available when you integrate WebSphere Portal Express with an on-premise IBM Connections server are not supported when you integrate WebSphere Portal Express with an Connections server that runs in the SmartCloud for Social Business.

Changes to features in IBM Connections in SmartCloud for Social Business

- The Connections business card on IBM Connections in SmartCloud for Social Business is not exposed to third-party applications.
- Connections Content Manager is not a part of IBM Connections in SmartCloud for Social Business
- Connections profiles reflect users of the same organization. Guests and visitors are ignored.
- Connections profiles API on IBM Connections in SmartCloud for Social Business no longer contains the deprecated board function.
- Seedlists for individual services are not exposed in IBM Connections in SmartCloud for Social Business.
- There is no public access to IBM Connections in SmartCloud for Social Business. A user must be logged in to access data.

The following services are available outside and within a community scope:

- Activity Stream
- Profiles
- Activities

All other services are available only within a community scope.

Changes to features in WebSphere Portal Express

Social Media Publisher
The Social Media Publisher does not support IBM Connections in SmartCloud for Social Business.

Social Rendering
Live names of authors and owners of social objects are not supported by SmartCloud for Social Business. Social rendering cannot be used on public pages, since access to IBM Connections in SmartCloud for Social Business is not possible without an authenticated user.

Community Membership Access Control Integration
The feature to restrict access to portal objects such as pages based on community membership is not supported by IBM Connections in SmartCloud for Social Business. Clear the check box **Limit access to this page to only community members** in the Community Picker Dialog in WebSphere Portal Express.

Portal Tag Cloud Federation
Portal Tag Cloud Federation with IBM Connections in SmartCloud for Social Business is not supported.

Search Integration
RCSS is the only available way to integrate Portal Search with IBM Connections in SmartCloud for Social Business to show both contents in one query, since seedlists are not exposed.

Enabling Portal Navigation in Connections by using Web Application Integrator (WAI)
This integration approach is not available for IBM Connections in SmartCloud for Social Business.

Visitors and Guests
The integration exposes only intra-organizational aspects. Nothing that is related to visitors or guests is exposed.

Connections Business Card
Connections business card in WebSphere Portal Express is not supported by SmartCloud for Social Business.

Establishing single sign-on (SSO) between the portal installation and IBM Connections in SmartCloud for Social Business

Learn about the requirements that must be fulfilled in order to successfully integrate WebSphere Portal Express with IBM Connections in SmartCloud for Social Business. You can provide a user experience where a user logs in one time into WebSphere Portal Express and then automatically sees all integrated Connections information.

Since SmartCloud for Social Business uses a specific user repository, the following requirements are established by using Identity Federation and Security Assertion Markup Language (SAML):

- Users do not have to maintain separate user names and passwords for WebSphere Portal Express and Connections.
- A user logs on one time in to your digital experience and can view:

  - Information from IBM Connections in SmartCloud for Social Business Rest APIs that are rendered by using portlets or IBM Web Content Manager rendering in WebSphere Portal Express.
  - The SmartCloud for Social Business Connections web user interface.
- The infrastructure to complete outbound calls must be configured to support SAML-based single sign-on (SSO), since single sign-on needs to be established for calls from the portal JVM to the Connections server in SmartCloud for Social Business APIs. To configure support for SAML-based single sign-on, you must use one of the following identity providers:
  - To use Tivoli Federated Identity Manager (TFIM) as your identity provider, view the information in *Configuration settings for Tivoli Federated Identity Manager (TFIM)* in the related links section.
  - CF05 To use Active Directory Federation Services (ADFS) as your identity provider, view the information in *Configuration settings for Active Directory Federation Services (ADFS)* in the related links.

  Notes:
  - SAML 2.0 is the only supported version.
  - Active Directory Federation Services (ADFS) is supported beginning with CF05 and later.

The infrastructure for your integrated system is as follows:

- There is an identity provider server in your local environment that uses the same user repository as your portal server.
- SSO between your portal server and the local identity provider system is established.
- The local identity provider server and the SmartCloud for Social Business infrastructure trust each other.

To establish trust between the local identity provider server and SmartCloud for Social Business, complete the steps that are outlined in the SmartCloud for Social Business documentation:

- For general information, see *Federated identity management* in the related links.
- For specifics of enabling federated Identity Management, see *Enabling federated identity management* in the related links.

CF03 "Configuring single sign-on (SSO) for backend calls to IBM Connections in SmartCloud for Social Business" on page 765
The Connections integration assets use a common infrastructure that is called HTTP Outbound to complete calls to the Connections backend server. Learn about the steps that are required to configure the HTTP outbound component to complete calls to IBM Connections in SmartCloud for Social Business.

CF03 "Configuring single sign-on (SSO) for browser-based access to IBM Connections in SmartCloud for Social Business" on page 768
Single sign-on (SSO) for browser-based access to IBM Connections in SmartCloud for Social Business is enabled by using Service Provider Initiated Authentication Flow. You can enable SSO for all links, including external URLs, custom markup, search results, and social portlets.

Related concepts:

"Configuration settings for Tivoli Federated Identity Manager (TFIM)" on page 3039
Learn about establishing an SAML-based SSO connection for Tivoli Federated Identity Manager. Tivoli Federated Identity Manager must be installed and

operational before an SSO connection can be established.

CF05 "Configuration settings for Active Directory Federation Services (ADFS)" on page 3048
Learn about establishing a single-sign on (SSO) connection for Active Directory Federation Services (ADFS).

Related information:

Federated identity management

Enabling federated identity management

Configuring single sign-on (SSO) for backend calls to IBM Connections in SmartCloud for Social Business

The Connections integration assets use a common infrastructure that is called HTTP Outbound to complete calls to the Connections backend server. Learn about the steps that are required to configure the HTTP outbound component to complete calls to IBM Connections in SmartCloud for Social Business.

Before you begin

If you have the default log level in the WebSphere Integrated Solutions Console set to `INFO` or lower, an increase in logged informational messages might occur when you establish an HTTP Outbound connection to IBM Connections in SmartCloud for Social Business. To disable the logging of informational messages so that only error and warning messages are logged, complete the following steps:

1. Log in to the WebSphere Integrated Solutions Console.
2. Click **Troubleshooting** > **Logs and trace** > **WebSphere_Portal**.
3. Click **Diagnostic trace** in the General Properties section.
4. Click **Change log detail levels** in the Additional Properties section.
5. Change the log detail level for `org.apache.commons.httpclient.*` to `WARNING` or higher. For example, if you are logging informational messages by default and therefore have `*=info` already specified, change the log detail level to `*=info:org.apache.commons.httpclient.*=WARNING`.
6. Click **Apply** > **OK**.
7. Restart your portal server for the changes to take effect.

Procedure

1. Set up a trust association between your portal server and the Connections server by using the SSL certificate. For detailed steps on how to do this, see *Import SSL certificate to set up trust association* in the related links.
2. Create a policy file that defines the URL and metadata that specifies the location of the Tivoli Federated Identity Manager server and which mapping to use. For complete instructions on how to complete this step, see *Configuration settings for Tivoli Federated Identity Manager (TFIM)* in the related links section. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<proxy-rules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://www.ibm.com/xmlns/prod/sw/http/outbound/proxy-config/2.0
 <mapping contextpath="/proxy" url="*" name="proxy"/>
 <mapping contextpath="/myproxy" url="*" name="myproxy"/>

  <!-- todo: change the URL policy according to your need -->
  <policy name="Connections_Cloud_Global" url="https://apps.na.collabserv.lotus.com/*">
   <actions>
    <method>GET</method>
```

```
 <method>HEAD</method>
 <method>POST</method>
 <method>PUT</method>
 <method>DELETE</method>
</actions>
<headers>
 <header>User-Agent</header>
 <header>Accept-Language</header>
 <header>Authorization*</header>
 <header>Content*</header>
</headers>
<cookie-rule name="Connections_Cloud_Global_Cookie_Rule">
 <cookie>*</cookie>
 <scope>user</scope>
 <handling>store-in-request</handling>
</cookie-rule>
<meta-data>
 <name>SSO_SAML20_IDP</name>
 <value>saml-tfim</value>
</meta-data>
<meta-data>
 <name>saml-tfim.IDP_PROTOCOL</mame>
 <value>https</value>
</meta-data>
<meta-data>
 <name>saml-tfim.IDP_HOST</name>
 <value>IDP_HOST</value>
</meta-data>
<meta-data>
 <name>saml-tfim.IDP_PORT</name>
 <value>IDP_PORT</value>
</meta-data>
<meta-data>
 <name>saml-tfim.IDP_URI</name>
 <value>IDP_URI</value>
</meta-data>
<meta-data>
 <name>saml-tfim.PARAM_NAME.1</name>
 <value>RequestBinding</value>
</meta-data>
<meta-data>
 <name>saml-tfim.PARAM_VALUE.1</name>
 <value>HTTPPost</value>
</meta-data>
<meta-data>
 <name>saml-tfim.PARAM_NAME.2</name>
 <value>PartnerId</value>
</meta-data>
<meta-data>
 <name>saml-tfim.PARAM_VALUE.2</name>
 <value>PARTNER_ID</value>
</meta-data>
<meta-data>
 <name>saml-tfim.PARAM_NAME.3</name>
 <value>TARGET</value>
</meta-data>
<meta-data>
 <name>saml-tfim.PARAM_VALUE.3</name>
 <value>https://apps.na.collabserv.lotus.com/</value>
</meta-data>
<meta-data>
 <name>saml-tfim.PARAM_NAME.4</name>
 <value>NameIdFormat</value>
</meta-data>
<meta-data>
 <name>saml-tfim.PARAM_VALUE.4</name>
 <value>Email</value>
```

```
    </meta-data>
    <meta-data>
     <name>saml-tfim.IDP_AUTH_COOKIE.1</name>
     <value>PD-ID</value>
    </meta-data>
    <meta-data>
     <name>forward-http-errors</name>
     <value>true<value>
    </meta-data>
    <meta-data>
     <name>socket-timeout</name>
     <value>10000</value>
    </meta-data>
    <meta-data>
     <name>retries</name>
     <value>2</value>
    </meta-data>
    <meta-data>
     <name>max-connections-per-host</name>
     <value>50</value>
    </meta-data>
   </policy>
</proxy-rules>
```

3. Push the policy to the WebSphere Portal Express policy store by running the **update-outbound-http-connection-config** task, where *XML_file* is the name of the policy file that you created in step 2:
   - AIX, HP-UX, Linux, Solaris: `./ConfigEngine.sh update-outbound-http-connection-config -DConfigFileName=XML_file -DOutboundProfileType=global`
   - IBM i: `ConfigEngine.sh update-outbound-http-connection-config -DConfigFileName=XML_file -DOutboundProfileType=global`
   - Windows: `ConfigEngine.bat update-outbound-http-connection-config -DConfigFileName=XML_file -DOutboundProfileType=global`

   Note: A preexisting Connections dynamic policy cannot be used for SmartCloud for Social Business. An error can occur if two policies exist. Verify that there is no value set for the **wp.proxy.config.urlreplacement.ibm_connections_policy** in the WP ConfigService Resource Environment Provider.
4. Verify the setup by logging in to WebSphere Portal Express and then calling the following URL: `http://<portalserver>:<portalserverport>/wps/myproxy/https/apps.na.collabserv.com/homeppage/web/updates/#imFollowing/all`.

   Note: You might need to change `apps.na.collabserv.com` to match the host name of your environment.
   If the setup is correct, Activity Stream markup displays without CSS styling. If the setup is incorrect, one of the following errors might occur:
   - A 403 error message displays, which means the proxy is not allowing the request.
   - Markup of the login page displays, which means that authentication failed.

   If an error occurs, check `SystemOut.log` for error messages, verify that the policy file is correct, and rerun the task.

Related concepts:

"Configuration settings for Tivoli Federated Identity Manager (TFIM)" on page 3039

Learn about establishing an SAML-based SSO connection for Tivoli Federated Identity Manager. Tivoli Federated Identity Manager must be installed and

operational before an SSO connection can be established.

CF05 "Configuration settings for Active Directory Federation Services (ADFS)" on page 3048
Learn about establishing a single-sign on (SSO) connection for Active Directory Federation Services (ADFS).

Related tasks:

"Import SSL certificate to set up trust association" on page 749
To prevent security alert pop-up windows, add Connections certificates to your portal server. The signer certificate that is used by the Connections server must be trusted by the portal server. The instructions for deploying the IBM Connections portlets include a procedure to import the SSL certificate. If you did not complete that procedure, you must import the certificate to use community pages. For best results, follow the procedures that are provided in the IBM Connections documentation.

Configuring single sign-on (SSO) for browser-based access to IBM Connections in SmartCloud for Social Business

Single sign-on (SSO) for browser-based access to IBM Connections in SmartCloud for Social Business is enabled by using Service Provider Initiated Authentication Flow. You can enable SSO for all links, including external URLs, custom markup, search results, and social portlets.

About this task

By default, if an unauthenticated user is working within an Connections Integration Asset, such as a social rendering list, and clicks a URL that points to the web user interface for IBM Connections in SmartCloud for Social Business, the user is redirected to the Connections login screen. This redirect to the Connections login screen occurs if the user is not authenticated or if their authentication expired. Enabling Service Provider Initiated Authentication Flow prevents this redirect to the Connections login screen. Instead, the user is authenticated by using the Tivoli Federated Identity Manager server.

Procedure

1. Contact SmartCloud for Social Business support by emailing `cloudcsg@us.ibm.com`.
2. In your email, request to have Service Provider Initiated Authentication Flow enabled for your system.

Results

If you enable a Service Provider Initiated Authentication Flow, the following scenario occurs the first time that a user clicks a URL that points to the web user interface for IBM Connections in SmartCloud for Social Business:

Note: The user is not yet authenticated with IBM Connections in SmartCloud for Social Business.

1. In the prompt that opens, the user selects **Use My Organization's Login Page**.
2. The user specifies their email address.
3. The user is redirected to the local Tivoli Federated Identity Manager server and then back to the IBM Connections in SmartCloud for Social Business web user interface without any further user interaction.
4. The user is asked if they want to be remembered. The user specifies yes.
5. The requested content displays for the user.

If the user specifies that they want to be remembered and does not clear their browser's cookies, the following scenario occurs the next time that the user clicks a URL that points to the web user interface for IBM Connections in SmartCloud for Social Business:

Note: The user is not yet authenticated with IBM Connections in SmartCloud for Social Business.

1. The user is redirected to the local Tivoli Federated Identity Manager server and then back to the IBM Connections in SmartCloud for Social Business web user interface without any further user interaction.
2. The requested content displays for the user.

The automatic redirect to the local Tivoli Federated Identity Manager server and back to IBM Connections in SmartCloud for Social Business occurs because the portal server and the local Tivoli Federated Identity Manager server are in a single sign-on domain. Therefore, the Tivoli Federated Identity Manager server knows the user and can directly initiate a redirect back to the IBM Connections in SmartCloud for Social Business page without any user interaction.

Configuring a connection between WebSphere Portal Express and IBM Connections in SmartCloud for Social Business

Learn how to configure the Connections integration assets to complete calls to IBM Connections in SmartCloud for Social Business.

About this task

The instructions for deploying the Connections portlets include a procedure to configure the following settings. If you did not complete that procedure, you must complete the following steps.

Procedure

1. Configure the URL for the Connections server by completing the following steps. You must create a custom property that stores the URL for the Connections server. The URL must include the port number and specify the HTTPS protocol.
   a. Log in to the WebSphere Integrated Solutions Console.
   b. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
   c. Select **WP ConnectionsIntegrationService**.
   a. Within **Additional properties**, click **Custom properties**.
   b. Click **globalBaseURL** and update the value to the URL of the Connections server. For example:

      ```
      https://apps.na.collabserv.com
      ```

   c. Click **globalBaseURLUnsecured** and update the value to the URL of the Connections server. For example:

      ```
      https://apps.na.collabserv.com
      ```

2. Configure the Connections server type by setting the custom property **`server.type`** in the WP Connections Integration Service resource environment provider in the WebSphere Integrated Solutions Console. For detailed steps on how to set the **`server.type`** property, see *Configuring the IBM Connections server type* in the related links.

Related tasks:

"Configuring the Connections server type" on page 2122
You can use social rendering with an on-premises IBM Connections server or with an Connections server that runs in the Smart Cloud for Social Business. If you use the latter type of connections server, you need to adapt the configuration accordingly.

Related information:

IBM Connections: Configuring the resource environment provider

Chapter 8. Backup and restore

Backup and recovery of data files and databases is an essential operation for any business system, particularly for data and applications that run in production environments. Create and follow a plan for backing up and recovering data on all tiers of your IBM WebSphere Portal Express deployment. IBM Installation Manager must also be included in backup and recovery planning. If you back up the WebSphere Portal Express file structure and then install a fix pack, the WebSphere Portal Express and IBM Installation Manager become out of sync after you restore the WebSphere Portal Express file system. This condition is not recoverable.

Backup and recovery include the WebSphere Portal Express file system and databases. Your backup and recovery plan needs to address each deployment tier: Complete system backup for catastrophic failures, back up of middleware such as WebSphere Portal Express and IBM WebSphere Application Server, and backup of individual applications that run on the middleware. Backup and recovery can be done on any or all of these tiers, depending on the needs of your portal deployment.

When you create a backup and recovery plan, consider these general questions:

- What procedure will you use to back up data?
- How often will you back up data?
- What are the trade-offs between online and offline backups?
- How does the scope of your portal deployment affect the backup and recovery strategy? For example, the number of users and the volume and importance of the data that is stored and used in applications affect your decisions about backup and recovery practices.
- Will you use IBM Tivoli Storage Manager or other utility to back up the file system?

Attention: Backing up and recovering a WebSphere Portal Express installation includes the WebSphere Application Server runtime environment and all applications that are deployed on WebSphere Portal Express. However, if applications use remote information sources outside of the WebSphere Portal Express databases and the LDAP directory, you need to consider these remote sources. Develop backup and recovery procedures for these remote sources as part of your comprehensive strategy.

"Guidelines for Idle Standby deployments" on page 772
If IBM WebSphere Portal Express is deployed in an Idle Standby topology, both a primary node and a secondary node are running in a cluster. The primary node is the active node, and the secondary node is the backup node. In this topology, there is no difference in database backup and restore because all cluster members use the same WebSphere Portal Express databases. However, you need to consider some additional factors when you back up and recover the file system in an Idle Standby deployment.

"Database considerations for backup and restore" on page 772
Before you back up IBM WebSphere Portal Express databases, determine whether you need to perform offline or online backup, what your requirements are for storage capacity and backup frequency, and your preferred utility.

"Backing up files, databases, and the LDAP server(s)" on page 774
Periodically run an automated backup procedure for the IBM WebSphere Portal

Express file system, databases, and LDAP server(s) using a backup and recovery utility of your choice. Remember to run the backup procedure before performing critical system-wide changes, such as upgrading to a new version of WebSphere Portal Express or installing interim fixes and fix packs.

"Restoring files, databases, and the LDAP server(s)" on page 779
When necessary, restore the IBM WebSphere Portal Express file system, databases, and LDAP server or servers that you backed up.

Guidelines for Idle Standby deployments

If IBM WebSphere Portal Express is deployed in an Idle Standby topology, both a primary node and a secondary node are running in a cluster. The primary node is the active node, and the secondary node is the backup node. In this topology, there is no difference in database backup and restore because all cluster members use the same WebSphere Portal Express databases. However, you need to consider some additional factors when you back up and recover the file system in an Idle Standby deployment.

About this task

In an Idle Standby deployment, adhere to these guidelines:

Procedure

1. Follow the file system backup and recovery procedures on both the primary and additional nodes.
2. Back up the file system of the Deployment Manager configuration, the master configuration repository for the cell, by using one of the following methods:
   - Use the simple commands that are provided by IBM WebSphere Application Server: `backupConfig` and `restoreConfig`.
   - Use IBM Tivoli Storage Manager or other utility to back up the Deployment Manager profile directory, where this configuration information resides.
3. When you restore WebSphere Portal Express databases and file systems, make sure to restore the Deployment Manager configuration at the same time that the primary node and secondary node file systems are restored. Then, complete a node synchronization of both nodes by using the Deployment Manager WebSphere Integrated Solutions Console. This step ensures that the configuration is identical on both nodes and is consistent with the restored databases.

Database considerations for backup and restore

Before you back up IBM WebSphere Portal Express databases, determine whether you need to perform offline or online backup, what your requirements are for storage capacity and backup frequency, and your preferred utility.

The backup method that you use depends on the type of deployment and the level of service that is being offered. Consider the following factors to determine whether to perform offline or online backup:

- Small or pilot deployments, including deployments with intermittent use, are best served by performing offline backups.

  Offline backups require that you stop all active applications that are connected to the database. When you perform an offline backup, the backup file that is created is a complete copy of all data stored in the database. Therefore, each offline backup requires the same amount of storage space as the entire database.

With small deployments, creating multiple full backups is more manageable than backing up larger systems with large amounts of data. Schedule regular downtime for backups during the organization's off-hours.

- Large deployments, including deployments that require continuous availability, are best served by performing online backups.

  Online backups can be performed while the database remains in service running applications; therefore, online backups require planning.

  As part of its normal operation, databases create logs that record changes to the database. Databases created with default parameters create *circular* logs, in which data is eventually overwritten. Setting up online backups requires that you store the logs on disk as *archive* logs. An online backup uses checkpoints in the log to back up incremental data or data changes rather than making a complete copy of the data as the offline backup process does. Because the logs grow in number and size over time as the database is used, you must ensure that the required storage is available. You will also need to back up the archived logs periodically because they are used during the data recovery and restoration process.

  For online backups, decide when to use incremental backups or delta backups. Both types of online backup are more space-efficient than full backups.

  – An *incremental backup* is a backup image that contains only portal pages that have been updated since the last full backup.
  – A *delta backup* is a copy of all database data that has changed since the last successful backup.

  A typical scheme is to schedule a full online backup on the first day of the week, followed by delta backups on the next two days, an incremental backup on the following day, delta backups on the next two days, and concluding with an incremental backup on the last day of the week.

Next, consider the data storage requirements and the resources and utilities that are available to you:

- Identify the resources needed for storing the backed-up database files.
- Determine a strategy for archiving the backed-up database files. For example, you may want to save a weekly full backup to tape and store the tape offsite.
- Decide which backup utilities to use.

  You can use IBM Tivoli Storage Manager to automate database backups and to organize and maintain the database backup files.

  You can also use tools and commands provided by your database to manage the backup procedures.

  Note if using DB2: For scheduling backup scripts, you can use either the DB2 Backup Wizard or the DB2 Task Center.

  – The Backup Wizard provides a simple, step-by-step user interface for configuring backup options, selecting a storage location for backup data files, and scheduling the backup.
  – The Task Center is an administrative component that allows you to create and schedule scripted functions.

Backing up files, databases, and the LDAP server(s)

Periodically run an automated backup procedure for the IBM WebSphere Portal Express file system, databases, and LDAP server(s) using a backup and recovery utility of your choice. Remember to run the backup procedure before performing critical system-wide changes, such as upgrading to a new version of WebSphere Portal Express or installing interim fixes and fix packs.

About this task

Perform the following tasks to back up the WebSphere Portal Express file system, databases, and LDAP server(s):

1. "Completing prerequisites for backup"
   Before you back up the WebSphere Portal Express installation, decide which utility to use and whether to perform online backup or offline backup.
2. "Backing up the WebSphere Portal Express file system"
   Periodically run an automated backup procedure for the IBM WebSphere Portal Express file system.
3. "Backing up the LDAP server(s)" on page 775
   If you have made system-wide changes and are using an LDAP user registry, back up your LDAP server.
4. "Backing up the WebSphere Portal Express database" on page 775
   Periodically run an automated backup procedure for the IBM WebSphere Portal Express databases.
5. "Backing up the IBM Installation Manager" on page 779
   IBM Installation Manager must also be included in backup and recovery planning. If you back up the WebSphere Portal Express file structure and then install a fix pack, the WebSphere Portal Express and IBM Installation Manager become out of sync after you restore the WebSphere Portal Express file system. This condition is not recoverable.

Completing prerequisites for backup

Before you back up the WebSphere Portal Express installation, decide which utility to use and whether to perform online backup or offline backup.

Before you begin

- Determine whether you want to use IBM Tivoli Storage Manager or another backup and recovery utility. Refer to the Tivoli Storage Manager documentation or the documentation for the utility that you choose to use.
- Determine whether you want to perform an online backup or an offline backup.
  - An online backup occurs while the portal servers are operational when you run the backup procedure.
  - An offline backup occurs when the portal servers are stopped prior to running the backup procedure.

  If you choose to use an online backup, make sure that the techniques and tools used for file system and database backup support the capturing of online backups where open files and database changes can be encountered during the backup procedure.

Backing up the WebSphere Portal Express file system

Periodically run an automated backup procedure for the IBM WebSphere Portal Express file system.

Procedure

1. If you are performing an online backup, go to Step 3.
2. If you are performing an offline backup, stop all servers.
   a. If applicable, stop all external HTTP servers.
   b. Stop all of the WebSphere Portal Express servers.
3. Back up all WebSphere Portal Express files and subdirectories under the directory *PortalServer_root*.
4. Back up all IBM WebSphere Application Server files under the directory *wp_profile_root* to capture the complete runtime environment.
5. Back up other WebSphere Application Server files under the directory *AppServer_root*.
6. Back up the IBM Installation Manager agent data location. This location contains information about the installed offerings and fixes. For details on the default location of this directory, see Agent Data Location in the Related information section.
7. Back up additional custom files outside of the `PortalServer`, profile, or `AppServer` directories; such as SSL certificates or shared libraries; that your WebSphere Portal Express server might use.

Related information:

Agent data location

Backing up the LDAP server(s)

If you have made system-wide changes and are using an LDAP user registry, back up your LDAP server.

About this task

Refer to the appropriate product documentation for information about how to back up the LDAP server:

- For IBM Directory Server, refer to IBM Directory Server Information Center.
- For IBM Domino, refer to the Directory Services topics in the Domino Administrator Help.
- For other LDAP servers, refer to that server documentation for instructions.

Related tasks:

"Completing prerequisites for backup" on page 774
Before you back up the WebSphere Portal Express installation, decide which utility to use and whether to perform online backup or offline backup.

Backing up the WebSphere Portal Express database

Periodically run an automated backup procedure for the IBM WebSphere Portal Express databases.

About this task

Note: The database backup should correspond to when you back up your file system.

Refer to the appropriate product documentation for information about how to back up the database server:

- For IBM DB2 Universal Database Enterprise Server Edition, refer to the DB2 documentation.
- For other database servers, refer to that database server documentation for instructions.

"Enabling logging"
Enable logging for the type of backup that you are performing.

"Using the DB2 Backup wizard" on page 778
Use the Backup wizard of IBM DB2 Universal Database Enterprise Server Edition to create online or offline backups. This sample scenario describes how to create a full offline backup of a IBM WebSphere Portal Express database.

Related tasks:

"Completing prerequisites for backup" on page 774
Before you back up the WebSphere Portal Express installation, decide which utility to use and whether to perform online backup or offline backup.

Enabling logging

Enable logging for the type of backup that you are performing.

"Enabling circular logging for offline backup"
Enable circular logging if you are performing offline backup of the database. Use the Configure Database Logging Wizard to specify the parameters that control circular logging.

"Enabling archive logging for online backup" on page 777
Enable archive logging if you are performing online backup of the database. Use the Configure Database Logging Wizard to specify the parameters that control archive logging.

Enabling circular logging for offline backup:

Enable circular logging if you are performing offline backup of the database. Use the Configure Database Logging Wizard to specify the parameters that control circular logging.

About this task

Note: This procedure refers to the *WPSDB* database, the default database name for the Release database domain.

To enable circular logging for an offline DB2 database backup, follow these steps:

Procedure

1. Open the DB2 Control Center: Click **General Administration Tools** > **Control Center**.
2. Select the *WPSDB* database from the navigation tree.
3. From the menu, choose **Selected** > **Configure Database Logging**.
4. In the Configure Database Logging Wizard, select **Circular Logging** for your logging type and click **Next**.
5. Accept the default values for the number and size of your log files and click **Next**.
6. Accept the default location for your log files or choose a new location.

   You can mirror your log files to store a copy of your log files in a second location, usually on a different drive. This is important for Idle Standby environments.

7. Click **Next**.
8. Select **Run now without saving history** and click **Next**.
9. Click **Finish** to execute the DB2 commands needed to update the logging configuration.
10. Select the *WPSDB* database from the navigation tree and, from the Control Center menu, choose **Selected** > **Configure Parameters**.
11. In the Recovery section, perform these steps:
    a. Set the value of the **TRACKMOD** parameter to *No*.
    b. Set the **LOG_RETAIN** parameter to *No* and click **OK**.
12. Restart DB2 to enable the changes.

Enabling archive logging for online backup:

Enable archive logging if you are performing online backup of the database. Use the Configure Database Logging Wizard to specify the parameters that control archive logging.

About this task

Note: This procedure refers to the *WPSDB* database, the default database name for the Release database domain.

To enable archive logging for an online DB2 database backup, follow these steps:

Procedure

1. Open the DB2 Control Center: Click **General Administration Tools** > **Control Center**.
2. Select the *WPSDB* database from the navigation tree.
3. From the menu, choose **Selected** > **Configure Database Logging**.
4. In the Configure Database Logging Wizard, select **Archive Logging** for your logging type and click **Next**.
5. Select **Use DB2 to automatically archive the log files** for the method of handling archived logs. Type a directory path for the **Primary archive log location** and **Failure archive log location** and click **Next**.
6. Accept the default values for the number and size of your log files and click **Next**.
7. Accept the default location for your log files or choose a new location.

   You can mirror your log files to store a copy of your log files in a second location, usually on a different drive. This is important for Idle Standby environments. If you do not mirror the log files, then you must back up the log files in addition to backing up the *WPSDB* database.
8. Click **Next**.
9. Type a directory path to store the initial backup. Because you are changing the logging method, this wizard automatically performs an initial backup.
10. Click **Next**.
11. Optional: Specify any options for the backup. Click **Next**.
12. Schedule the backup task for a later time or select the option **Run now without saving task history**. Click **Next**.
13. On the Summary screen, review the selections that you made with the wizard and click **Finish** to accept the changes and create the initial backup.

14. Select the *WPSDB* database from the navigation tree and, from the Control Center menu, choose **Selected** > **Configure Parameters**.
15. In the Recovery section, specify the following parameters:
    a. Set the value of the **TRACKMOD** parameter to *Yes*.
    b. Set the **LOG_RETAIN** parameter to *Recovery*.
    c. Click **OK** to save these settings.
16. To enable the changes, restart the DB2 instance for the WebSphere Portal Express server that you are backing up.

Using the DB2 Backup wizard

Use the Backup wizard of IBM DB2 Universal Database Enterprise Server Edition to create online or offline backups. This sample scenario describes how to create a full offline backup of a IBM WebSphere Portal Express database.

Before you begin

Before you perform an offline backup, stop all services that access the DB2 database: *WPSDB*, which is the default database for the Release database domain.

About this task

To perform a full offline backup of the *WPSDB* database, follow these steps:

Procedure

1. Stop the servers.
2. Open the DB2 Control Center: Click **General Administration Tools** > **Control Center**.
3. To open the Backup wizard, expand the object tree until you find the *WPSDB* database, display its menu, and select **Backup**.
4. After confirming that the details of your database are correct, click **Next**.
5. Select a storage location for the backup and click **Next**.
6. Select the backup type **Full backup** and an availability of **Offline**.
7. Unless you have configured a separate task to quiesce the database, select **Quiesce** to ensure that all users are disconnected from the database before the backup task begins.
8. Click **Next**.
9. If you are experiencing performance problems during backups, skip the next step and go to Step 11.
10. At Specify performance options for the backup, click **Next** to choose one of the following options:
    - To specify a schedule for the backup task, click **Change** and click **Next**.

      Important: Select a time when no users are working with the database or with applications that might be running on the server.
    - To perform a backup without setting a schedule, click **Run now without saving task history** and click **Next**.
11. Optional: Click **Show Command** to view the DB2 commands that the Backup wizard runs.

    Tip: You can copy these commands to use in the Task Center as an alternative to using the Backup wizard the next time you want to back up the database.

Note: The following example includes line breaks for readability. In actual practice, you need to type all commands, including the semicolons, on one line.

```
CONNECT TO WPSDB;QUIESCE DATABASE IMMEDIATE FORCE CONNECTIONS;CONNECT RESET;
BACKUP DATABASE WPSDB TO "D:\BackupFiles\DB2backup\" WITH 2 BUFFERS
BUFFER 1024 PARALLELISM 1 WITHOUT PROMPTING;CONNECT TO WPSDB;
UNQUIESCE DATABASE;CONNECT RESET;
```

12. When you are satisfied with your settings for the database backup, click **Finish** to start the backup task.

Results

DB2 displays a message confirming that the backup task has completed successfully or that the task encountered and logged problems that need to be fixed.

Backing up the IBM Installation Manager

IBM Installation Manager must also be included in backup and recovery planning. If you back up the WebSphere Portal Express file structure and then install a fix pack, the WebSphere Portal Express and IBM Installation Manager become out of sync after you restore the WebSphere Portal Express file system. This condition is not recoverable.

WebSphere Process Server V7 is now installed and maintained through IBM Installation Manager, which saves information on installed products in the agent data store and shared files. The damage or loss of shared files or IBM Installation Manager agent data can prevent the Installation Manager from working within your environment.

Without agent data files, you cannot change installed products or manage products and their components that you installed with the Installation Manager. For example, you cannot modify, update or install the products. Therefore, as part of your Disaster Recovery Plan, you need to have regular backups for the IBM Installation Manager agent and shared files.

For more information about the appdata locations for all versions of IBM Installation Manager, see *Backing up and restoring Installation Manager*.

Related tasks:

"Completing prerequisites for backup" on page 774
Before you back up the WebSphere Portal Express installation, decide which utility to use and whether to perform online backup or offline backup.

Related information:

Agent data location

Restoring files, databases, and the LDAP server(s)

When necessary, restore the IBM WebSphere Portal Express file system, databases, and LDAP server or servers that you backed up.

Before you begin

Before you restore WebSphere Portal Express databases, remember to adhere to the following principles:

- Whenever you restore WebSphere Portal Express databases, you must restore the databases for all of the nonshared database domains to ensure consistency: Release, LikeMinds, Java Content Repository (JCR), and Feedback are the nonshared database domains.
- Restoring the databases of the shareable database domains is optional and might not be required when you do not want to lose recent user and community information: Customization and Community are the shareable database domains.
- Whenever you restore the WebSphere Portal Express databases, you must also restore the configuration and data files that were archived when the backup image was taken.
- Be sure that your database restoration rolls forward to the point when you performed the file system backup for the WebSphere Portal Express deployment. If you fail to do this step, the configuration and data files might not be synchronized with the information in the databases.
- The file system backup for WebSphere Portal Express includes directories that contain the configuration files.
- Consider whether you implemented separate databases for each database domain or multiple database domains by using the same database. For example, if all database domains were implemented by using a single WebSphere Portal Express database, then restoring this database restores the contents of all domains.
- If you are using IBM DB2 Universal Database Enterprise Server Edition, you should understand the basics of DB2 data backup.
- If you are using a different database management system (DBMS), refer to the DBMS documentation for backup instructions.

Procedure

1. Stop all servers.
   a. If applicable, stop all external HTTP servers.
   b. Stop all of the WebSphere Portal Express servers.
2. Move the backed up files to their original location.

   Refer to the documentation of the backup utility that you used for instructions on restoring files.

   WARNING: Do not overwrite the existing AppServer, PortalServer, or wp_profile root directories with the backed up files because you risk corrupting the WebSphere Portal Express file system. First, remove these old directories and then extract the backed up version in its place.
3. Restore the WebSphere Portal Express databases:
   - For IBM DB2 Universal Database Enterprise Server Edition, refer to the DB2 documentation.
   - For other database servers, refer to that database server documentation for instructions.
4. Optional: If necessary, restore your LDAP server.
   - For IBM Directory Server, refer to IBM Directory Server documentation.
   - For IBM Domino, refer to the Directory Services topics in the Domino Administrator Help.
   - For other LDAP servers, refer to the product documentation for instructions.

"Using the DB2 RESTORE DATABASE command"
Use the RESTORE DATABASE command of IBM DB2 Universal Database Enterprise Server Edition as an alternative to the DB2 Restore wizard to restore the databases that you backed up.

"Using the DB2 Restore wizard"
Use the Restore wizard of IBM DB2 Universal Database Enterprise Server Edition as an alternative to the DB2 RESTORE DATABASE command to restore the databases that you backed up.

Using the DB2 RESTORE DATABASE command

Use the RESTORE DATABASE command of IBM DB2 Universal Database Enterprise Server Edition as an alternative to the DB2 Restore wizard to restore the databases that you backed up.

About this task

To replace an existing *WPSDB* database with the backup copy using the **RESTORE DATABASE** command, follow these steps:

Procedure

1. Stop all application servers connected to the *WPSDB* database.
2. Determine which backup copy to use for the restoration by looking at the timestamp of the available backup files.
3. Run the **RESTORE DATABASE** command: This example refers to the *WPSDB* database, the default name for the Release database domain. **RESTORE DATABASE** *WPSDB* **FROM** *backup_directory_path* **TAKEN AT** *timestamp* **REPLACE EXISTING**
4. Roll the restored database forward by choosing the same point in time for the database. Consider these factors:
   - The time needs to be expressed as an ISO timestamp string in this format *YYYY-MM-DD-HH.MI.SS.NNNNNN*.
   - The time should be shortly after the completion of the backup.
   - The best way to determine this time is to use **Show History** in the Task Center to review the output saved for the backup commands.
   - Alternately, you can use the modification time of the saved backup file to determine when the backup task completed.

   For example, use the following roll-forward command when the last backup completed at 1:23 AM:ROLLFORWARD DATABASE *WPSDB* **TO** *2007-06-25-01.23.00.000000* **USING LOCAL TIME** AND **COMPLETE**

Using the DB2 Restore wizard

Use the Restore wizard of IBM DB2 Universal Database Enterprise Server Edition as an alternative to the DB2 RESTORE DATABASE command to restore the databases that you backed up.

About this task

To replace an existing *WPSDB* database with the backup copy using the DB2 Restore wizard, follow these steps:

Procedure

1. Stop all application servers connected to the *WPSDB* database.
2. Stop and start DB2 to make sure that there are no connections to the database.

3. In the DB2 Control Center, select **Tools** > **Wizards** > **Restore Wizard** from the list of available wizards to define and schedule the restoration of the database.
4. Select the database instance *DB2* and the database *WPSDB* to restore. If the *WPSDB* database does not display in the Database field, type `WPSDB` and click **OK**. A series of screens prompt you to specify information that controls how the database restoration is performed.
5. Choose one of the following options:
   - Select **Restore to an existing database** if you have an existing *WPSDB* database, but it is corrupted or does not contain recent data.

     For example, consider the case where both IBM WebSphere Portal Express and DB2 are installed on the same machine, and that machine suffered a disk failure. To get back to full functionality, you must reinstall DB2, reinstall WebSphere Portal Express, redeploy the server, and run the LDAP and DB2 transfer wizards. This operation would cause a new *WPSDB* database to be created in DB2. However, that database would not contain any data.
   - Select **Restore to a new database** if you no longer have an existing *WPSDB* database in DB2.

     For example, if you had two machines, one contained WebSphere Portal Express, and the second machine contained DB2 and the DB2 data. If the second machine failed, then you would need to reinstall DB2. You could then use the Restore Wizard to re-create the *WPSDB* database from a backed up *WPSDB* database.
6. Click **Next**.
7. If you selected to restore to a new database, type `WPSDB` as the name of the new database, and type the location of the database after the restore. Also, specify the location of the log files for the database after the restore.
8. Click **Next.**
9. Specify a backup image using one of the following options:
   - If you selected to **Restore to an existing database**, select the most recent backup file to use for the restore operation.
   - If you selected to **Restore to a new database**, type the media type, path, and date and time of the most recent backup.
10. Click **Next.**
11. On the Set your containers for a redirected restore screen, do not make any changes. Click **Next**.
12. On the Choose your restore options screen, do not enable **Datalink columns**. Click **Next**.
13. On the Select performance options for the restore screen, do not make any changes. Click **Next**.
14. On the Enabling the DB2 scheduling function screen, select one of these options:
    - Select **Run now without saving task history** to perform the restore immediately.
    - Select **Enable scheduler** to schedule the restore operation for a later time.
15. Click **Next.**
16. Optional: On the final screen that shows summary information, click the **Show Command** to view the DB2 commands that the Restore wizard will run. You can copy these commands to use in the DB2 Task Center as an alternative to using the Restore Wizard.

17. Click **Finish** to begin the restore process.
18. Stop and restart all DB2 services.
19. Restart all servers.

Chapter 9. Migrating

Successful migration requires significant planning and preparation, understanding the tools that are involved, and careful execution of the appropriate steps in the order provided.

"Migration overview"
Migration is the process of collecting configuration data and applications from an earlier installed version of IBM WebSphere Portal Express and merging them into a newer installed version. So that the new environment is identical to the earlier environment.

"Planning for migration" on page 787
Completing a thorough plan before migrating to the latest version of WebSphere Portal Express has a direct impact in the effort invested during the actual migration. Become familiar with the environment you are migrating to (target environment). Also, make sure that the environment you are migrating from (source environment) is up to date with fixes and meets the requirements for migration.

"Preparing your source environment" on page 814
Before you begin the steps for migration, you must perform some critical tasks on your source environment, such as creating back ups, installing the latest cumulative fix and one of the two most recent fix packs, and disabling automatic synchronization if you are migrating a cluster. Review the topics in this section, and perform the required tasks to ensure that your source environment remains functional and the migration completes successfully.

"Setting up the target environment" on page 831
The migration requires you to install the required Portal and WebSphere binary installations. To prepare your target environment, ensure that you have applied the latest cumulative fix and the most recent fix pack before you start migration. In addition, create new copies of the source databases for the target environment to use, and prepare the target environment for any custom applications that have dependencies or any other tasks that need to be performed for remote or cluster migrations.

"Migrate data using the configuration wizard" on page 841
For Version 8.5, data, applications, databases, property files, security settings, and configuration are migrated using the Configuration Wizard. Use the roadmaps for cluster and stand-alone environments to guide you through the process.

"Next steps" on page 856
To complete migration, you must first perform several post-migration steps that depend on how Portal is being used. After completing the post-migration steps, review the Enabling new functionality section to take advantage of the new tools available in IBM WebSphere Portal Express Version 8.5. Enabling new functionality should not be started until all post-migration steps have been completed.

Migration overview

Migration is the process of collecting configuration data and applications from an earlier installed version of IBM WebSphere Portal Express and merging them into a newer installed version. So that the new environment is identical to the earlier environment.

Migration is different from upgrading. With upgrading, you replace an existing installed out-of-date version of files with current files. With migration, you install the new version of a product alongside of the earlier version and then copy data from the earlier version to the new version. By migrating information from the earlier version to the new version, you can use that information in the new version without having to re-create it from scratch. Migration enables customizations to be carried forward that were implemented in the earlier portal so that you can continue to use them in the new portal.

WebSphere Portal Express also supports integration with additional products to extend core functionality. If the earlier portal environment is configured to work with one or more supported products that provided integrated features, you need to follow the migration procedures for the integrated product.

Migrated elements are not automatically upgraded to use features that are available in the new version. Taking advantage of new features that were not available in the earlier portal requires extra attention after migration is complete.

You can migrate to WebSphere Portal Express Version 8.5 from either Version 7.0 or 8.0. For additional information about supported migrations, see *Supported migration paths*.

Note: If you are migrating from IBM WebSphere Portal Express Version 8.0.0.1 on WebSphere Application Server Version 8.5.5.2 to WebSphere Portal Express Version 8.5, you must follow a different migration process. For more information, see *Migrating from Portal 8.0.0.1 on WebSphere Application Server 8.5.5.2*.

Migration to Version 8.5

For Version 8.5, data, applications, databases, property files, security settings, and configuration are migrated using the Configuration Wizard. Use the roadmaps for cluster and stand-alone migrations to guide you through planning, preparing your source environment, setting up your target environment, migrating data, and post-migration steps. For a high-level overview of this process, see the *Roadmaps for migration*.

Related concepts:

"Supported migration paths" on page 788
Migration is supported between equivalent offerings. For example, you can migrate from WebSphere Portal Enable Version 7.0 to WebSphere Portal Enable Version 8.5, but not from WebSphere Portal Express Version 7.0 to WebSphere Portal Extend Version 8.5.

"Migrating from Portal 8.0.0.1 on WebSphere Application Server 8.5.5.2" on page 800
If you are migrating from IBM WebSphere Portal Express Version 8.0.0.1 on WebSphere Application Server Version 8.5.5.2 to WebSphere Portal Express Version 8.5, you must follow a different migration process.

"What to expect after you complete migration" on page 811
During the migration process, your portal applications, portlets, and databases are updated to the IBM WebSphere Portal Express 8.5 versions. However, not all of the new WebSphere Portal Express Version 8.5 functionality and features are enabled by default. The following sections provide information on how various components are handled during migration, and what you can expect after the migration is complete.

Related tasks:

"Roadmaps for migration" on page 92
Choose the appropriate migration roadmap for your environment.

Planning for migration

Completing a thorough plan before migrating to the latest version of WebSphere Portal Express has a direct impact in the effort invested during the actual migration. Become familiar with the environment you are migrating to (target environment). Also, make sure that the environment you are migrating from (source environment) is up to date with fixes and meets the requirements for migration.

The following sections provide a starting point for your migration planning. Consider each migration planning unique to the environment you are migrating. Refer to the Roadmaps for migration when you are planning and completing the migration process for an high-level view of the process and direct links to important documentation resources.

Before you start the more detailed planning, here are some additional migration considerations:

- Take into account new development environment needs.
- Education and cultural changes.
- Contemplate different migration approaches and the impact on your business.
- What is contained within migration and what is not.
- Deprecated functionality.
- Vendor software unique to your environment and how it might interact with the new version of WebSphere Portal Express.

  "Supported migration paths" on page 788
  Migration is supported between equivalent offerings. For example, you can migrate from WebSphere Portal Enable Version 7.0 to WebSphere Portal Enable Version 8.5, but not from WebSphere Portal Express Version 7.0 to WebSphere Portal Extend Version 8.5.

  "Hardware considerations" on page 791
  There might be cases where you are completing a remote migration to a new hardware configuration. This new server might have different requirements from the current environment you are migrating from. Generally, here are some key points to consider.

  "Operating systems considerations" on page 791
  There might be cases where you upgrade not just to a newer version of WebSphere Portal Express but also to a different operating system version. In that case, there might be a different set of system requirements and considerations to keep in mind.

  "Migration considerations" on page 792
  There are a number of ways in which you can migrate WebSphere Portal Express to a newer version. Some migration scenarios might offer a higher availability percentage over another. There are some scenarios where the migration can be done in parallel while your source environment remains in production. Other scenarios might require the production system to be disconnected just before you go live with the newly migrated system. Depending on your needs on high availability systems, you might choose one approach or another.

  "Development considerations" on page 808
  The goal of the migration process is to ensure that the target environment

works similarly to the source environment. However, there are deprecated and unsupported features and changes in supported technical specifications that can prevent this transition from being seamless. Review the following topics for guidance on the development work that is required to maintain the functionality of the source environment and also begin preparation for enabling new features and functionality.

"What to expect after you complete migration" on page 811
During the migration process, your portal applications, portlets, and databases are updated to the IBM WebSphere Portal Express 8.5 versions. However, not all of the new WebSphere Portal Express Version 8.5 functionality and features are enabled by default. The following sections provide information on how various components are handled during migration, and what you can expect after the migration is complete.

Related concepts:

"What's changed" on page 16
WebSphere Portal Express includes changes to existing features.

"Unsupported and deprecated features for V8.5" on page 17
Review the features that were available in previous versions of IBM WebSphere Portal Express but are no longer available.

Related information:

WebSphere Portal V8.5 and V8.0 detailed system requirements

Supported migration paths

Migration is supported between equivalent offerings. For example, you can migrate from WebSphere Portal Enable Version 7.0 to WebSphere Portal Enable Version 8.5, but not from WebSphere Portal Express Version 7.0 to WebSphere Portal Extend Version 8.5.

The following table summarizes supported migration paths:

Table 100. Offerings and supported migration paths

Offering	WebSphere Portal Express Version 8.5	WebSphere Portal Version 8.5 (Server)	WebSphere Portal Version 8.5 (Enable, Extend)	Web Content Manager Version 8.5
WebSphere Portal Express Version 7.x	Supported	Not Supported	Not Supported	Not Supported
WebSphere Portal Server Version 7.x (Server)	Not Supported	Supported	Supported	Not Supported
WebSphere Portal Server Version 7.x (Enable, Extend)	Not Supported	Not Supported	Supported	Not Supported
Web Content Manager Version 7.x	Not Supported	Not Supported	Not Supported	Supported

Table 100. Offerings and supported migration paths (continued)

Offering	WebSphere Portal Express Version 8.5	WebSphere Portal Version 8.5 (Server)	WebSphere Portal Version 8.5 (Enable, Extend)	Web Content Manager Version 8.5
WebSphere Portal Express Version 8.0 on WebSphere Application Server Version 8.0	Supported	Not Supported	Not Supported	Not Supported
WebSphere Portal Server Version 8.0 (Server) on WebSphere Application Server Version 8.0	Not Supported	Supported	Supported	Not Supported
WebSphere Portal Server Version 8.0 (Enable, Extend) on WebSphere Application Server Version 8.0	Not Supported	Not Supported	Supported	Not Supported
Web Content Manager Version 8.0 on WebSphere Application Server Version 8.0	Not Supported	Not Supported	Not Supported	Supported
WebSphere Portal Server Version 8.0 (Server) on WebSphere Application Server Version 8.5.0	Not Supported	Supported	Supported	Not Supported
WebSphere Portal Server Version 8.0 (Enable, Extend) on WebSphere Application Server Version 8.5.0	Not Supported	Not Supported	Supported	Not Supported

Table 100. Offerings and supported migration paths (continued)

Offering	WebSphere Portal Express Version 8.5	WebSphere Portal Version 8.5 (Server)	WebSphere Portal Version 8.5 (Enable, Extend)	Web Content Manager Version 8.5
Web Content Manager Version 8.0 on WebSphere Application Server Version 8.5.0	Not Supported	Not Supported	Not Supported	Supported

You can also migrate from a Server install to the Enable or Extend versions of IBM WebSphere Portal Express.

Note for Version 6.1 customers: If you are currently on Version 6.1, you must perform a two-step migration from Version 6.1 to Version 8.0, and then from Version 8.0 to 8.5. Go to Migrating from WebSphere Portal 6.1 to Portal 8.5 for guidance on the two-step migration process.

Important Fix Pack Requirements: Migration is supported from the two most recent fix packs for WebSphere Portal Express Version 7.0.0.x and Version 8.0.0.x. However, you must apply the latest cumulative fix to your source environment. Your target environment must also have the latest cumulative fix and the most recent fix pack applied.

If you are not sure which earlier version is installed, run the following command on the earlier portal server:

Windows: *wp_profile_root*`\PortalServer\bin\WPVersionInfo.bat`

Linux: *wp_profile_root*`/PortalServer/bin/WPVersionInfo.sh`

IBM i: *wp_profile_root*`/PortalServer/bin/WPVersionInfo.sh`

When you migrate to Version 8.5, WebSphere Portal Express automatically migrates the following applications and configuration data:

- Security configuration
- Access control
- Portal behavior
- Portlet applications
- Customized portal resources, such as themes and skins, pages, and portlets
- Personalized content
- Virtual portals

Note: You cannot upgrade the source portal with a fix pack after migration if you intend to remigrate the JCR. For example, if your source portal is Version 7.0.0.4 and you migrate it to Version 8.5, you cannot then upgrade the source portal to Version 7.0.0.5 and remigrate the JCR. This path is not supported.

Related information:

WebSphere Portal detailed system requirements

Hardware considerations

There might be cases where you are completing a remote migration to a new hardware configuration. This new server might have different requirements from the current environment you are migrating from. Generally, here are some key points to consider.

Source environment:

- Revisit the hardware the software requirements for the platform you are migrating from. Ensure that you are still in a fully supported environment.
- Make sure that you are current with, at least, the last two maintenance levels for that platform.
- Be aware of the current overall performance and throughput. Know what the current activity is on your WebSphere Portal Express server.
- Make sure that you have detailed information about your entire environment.
  - Identify all the systems your WebSphere Portal Express server connects to.
  - Keep a detailed list of all the credentials and connection information as well as the services provided by the other systems.
  - If firewalls are involved, make sure that you note ports, IP addresses, and host names used.

Target environment:

- Review the system requirements for your new environment.
- Review the supported migration paths.
- Will this new system be dedicated as a WebSphere Portal Express server or it will be shared with other services?
  - If shared, be aware of potential resource conflicts such as ports as well as overall system performance.
- Make sure that you are at a supported maintenance level.
- Be mindful of the projected WebSphere Portal Express server activity. Once migrated, or even during the testing phase of your migration, you should be able to compare performance data between your current and new WebSphere Portal Express servers.

Architecture:

- Your target environment must be on the same system architecture than your source environment for migration. Migrating to a different architecture is not supported.

Related information:


WebSphere Portal detailed system requirements

Operating systems considerations

There might be cases where you upgrade not just to a newer version of WebSphere Portal Express but also to a different operating system version. In that case, there might be a different set of system requirements and considerations to keep in mind.

Important: Migrating between platforms (for example, migrating from Windows to Linux) is not supported.

Use the following list as a starting point.

- Review the supported migration paths.
- Review the system requirements for your new environment to make sure that you are migrating to a fully supported environment.
- Ensure that you applied all necessary fixes in preparation for migration.
- Get a complete understanding of the differences between the source and target environments. The following questions will help you get started.
  - Overall system administration. Are there any monitoring or administering tools in use today that might not work on the new environment?
  - Operating system security. Is there any security polices in effect that would require being ported to the new environment?
  - Will WebSphere Portal Express security implementation be any different?
  - File system. Will the files be placed in a different directory structure? Will those files require special permissions to run?

Migration considerations

There are a number of ways in which you can migrate WebSphere Portal Express to a newer version. Some migration scenarios might offer a higher availability percentage over another. There are some scenarios where the migration can be done in parallel while your source environment remains in production. Other scenarios might require the production system to be disconnected just before you go live with the newly migrated system. Depending on your needs on high availability systems, you might choose one approach or another.

Refer to the Server topologies section of this documentation for details on high availability installations.

"Backup and recovery" on page 793
Backup and recovery of data files and databases is an essential operation for any business system, particularly for data and applications that run in production environments. Create and follow a plan for backing up and recovering data on all tiers of your IBM WebSphere Portal Express deployment. IBM Installation Manager must also be included in backup and recovery planning. If you back up the WebSphere Portal Express file structure and then install a fix pack, the WebSphere Portal Express and IBM Installation Manager become out of sync after you restore the WebSphere Portal Express file system. This condition is not recoverable.

"Local or remote migration" on page 794
Review the considerations for local and remote migrations. Plan and review your options for an appropriate migration path that is based on your current environment.

"Automated or manual migration" on page 795
The IBM WebSphere Portal Express Knowledge Center documents the automated migration process that is the commonly used and supported method for migrating to a new version of WebSphere Portal Express. However, this approach might not be the ideal type of migration for all customers. Read the following considerations to determine which approach fits your needs.

"Portal farm migration" on page 797
The Portal farm migration consists of disabling farm mode and running the stand-alone migration. If you are migrating a unique installation farm configuration, you must migrate each farm member. If each farm instance is a clone, you can migrate one instance, and then create clones that are based on the migrated instance. When the migration is complete, you can re-enable farm mode.

Backup and recovery

Backup and recovery of data files and databases is an essential operation for any business system, particularly for data and applications that run in production environments. Create and follow a plan for backing up and recovering data on all tiers of your IBM WebSphere Portal Express deployment. IBM Installation Manager must also be included in backup and recovery planning. If you back up the WebSphere Portal Express file structure and then install a fix pack, the WebSphere Portal Express and IBM Installation Manager become out of sync after you restore the WebSphere Portal Express file system. This condition is not recoverable.

Backup and recovery include the WebSphere Portal Express file system and databases. Your backup and recovery plan needs to address each deployment tier: Complete system backup for catastrophic failures, back up of middleware such as WebSphere Portal Express and IBM WebSphere Application Server, and backup of individual applications that run on the middleware. Backup and recovery can be done on any or all of these tiers, depending on the needs of your portal deployment.

When you create a backup and recovery plan, consider these general questions:

- What procedure will you use to back up data?
- How often will you back up data?
- What are the trade-offs between online and offline backups?
- How does the scope of your portal deployment affect the backup and recovery strategy? For example, the number of users and the volume and importance of the data that is stored and used in applications affect your decisions about backup and recovery practices.

- Will you use IBM Tivoli Storage Manager or other utility to back up the file system?

Attention: Backing up and recovering a WebSphere Portal Express installation includes the WebSphere Application Server runtime environment and all applications that are deployed on WebSphere Portal Express. However, if applications use remote information sources outside of the WebSphere Portal Express databases and the LDAP directory, you need to consider these remote sources. Develop backup and recovery procedures for these remote sources as part of your comprehensive strategy.

Related tasks:

"Guidelines for Idle Standby deployments" on page 772
If IBM WebSphere Portal Express is deployed in an Idle Standby topology, both a primary node and a secondary node are running in a cluster. The primary node is the active node, and the secondary node is the backup node. In this topology, there is no difference in database backup and restore because all cluster members use the same WebSphere Portal Express databases. However, you need to consider some additional factors when you back up and recover the file system in an Idle Standby deployment.

"Backing up files, databases, and the LDAP server(s)" on page 774
Periodically run an automated backup procedure for the IBM WebSphere Portal Express file system, databases, and LDAP server(s) using a backup and recovery utility of your choice. Remember to run the backup procedure before performing critical system-wide changes, such as upgrading to a new version of WebSphere Portal Express or installing interim fixes and fix packs.

"Restoring files, databases, and the LDAP server(s)" on page 779
When necessary, restore the IBM WebSphere Portal Express file system, databases, and LDAP server or servers that you backed up.

Local or remote migration

Review the considerations for local and remote migrations. Plan and review your options for an appropriate migration path that is based on your current environment.

Local migration

There are a few migration combinations that can be considered local migration depending on the availability of the environment being migrated. At a minimum, you have two distinctive options with this migration path. You can perform a local migration by taking the source environment offline, or you can maintain both source and target environments coexisting. Depending on the migration path you choose, there might be additional factors to consider.

Note: IBM i supports local migration only.

Local migration without coexistence

Generally, local migration without coexistence is the simplest scenario. Both source and target WebSphere Portal Express installations are on the same system and resource conflicts are minimal.

There are still some directory structure considerations to keep in mind. You might want to use the same directory structure in you new WebSphere Portal Express installation to hold logs and backups for example.

Other configurations such as ports or virtual portals should not conflict. Both, source and target WebSphere Portal Express will not be running in parallel at the same time.

Local migration with coexistence

Coexistence is the process of running both the original and the newly migrated WebSphere Portal Express systems on the same system at the same time. This approach allows for maintaining your current production environment online while performing the migration to a newer version of WebSphere Portal Express. This statement is not exclusive of local migration with coexistence. You can also achieve this by performing a remote migration.

When you plan migration with coexistence you need to consider the hardware and software requirements for the new version of WebSphere Portal Express. Make sure the system currently running your production WebSphere Portal Express has enough resources to handle the new installation.

As you have two installations of WebSphere Portal Express you need to carefully plan to avoid resource conflicts. Default settings, like port assignments, require updating on the migrated WebSphere Portal Express.

Remote migration

In this scenario, you do not need to worry about resource conflicts as much as with the local migration with coexistence approach. However, being on a remote system you need consider factors outside the actual WebSphere Portal Express environment such as operating system accounts and security in general.

You also need to keep in mind that you must be able to move files between the source and the target environments. There are procedures that create files on the source WebSphere Portal Express environment that might need to be updated and placed in the new WebSphere Portal Express installation. So, once again, operating system security planning plays an important role in remote migrations.

Automated or manual migration

The IBM WebSphere Portal Express Knowledge Center documents the automated migration process that is the commonly used and supported method for migrating to a new version of WebSphere Portal Express. However, this approach might not be the ideal type of migration for all customers. Read the following considerations to determine which approach fits your needs.

The automated approach is intended to provide a one-size fits all migration that brings over the entire source site to the target environment. It takes into account all of the possible artifacts that a customer site might be using and packages them all to be brought over to the target environment. This is a great reproducible approach that satisfies the needs for most customers, but it can take longer because it attempts to move everything over.

However, it is also possible to migrate environments manually using administrative tools such as wsadmin, XMLAccess, and the ConfigEngine to move essential configurations, applications, and content from your existing source environment to the target environment.

Here are some factors to consider when deciding whether a manual approach is more effective:

- If you plan to re-create your site completely on the target version, you might not need to go through the automated migration process. You can start with a new 8.5 server, and then use the available administrative tools to bring over the configuration and artifacts from your source environment that you plan to preserve.
- If you are planning a two-step migration, from Version 6.1 to Version 8.0, and then Version 8 to Version 8.5, then review the deprecated and unsupported list for versions 7, 8, and 8.5 to determine whether your current site will work on Version 8.5 immediately after migration. In most cases, the theme must be re-created and many references to deprecated portlets and pages must be removed or updated. For this scenario, it might be more efficient to start with a new 8.5 installation and manually bring over the required artifacts from your source environment, and then update them as they are brought over.
- If you already have a well-documented deployment process, and you already know how to quickly create up a new environment that meets all of your requirements, then it is possible to use the core of that same process to create an equivalent version on the new version of Portal. There will be changes to the process, but it might be more efficient to use that process and update it for Version 8.5 as needed to redefine and use the process after migrating to Version 8.5.

If you plan to do a manual migration, review the following considerations first:

- IBM Support is limited if you choose to perform a manual migration.
  - XMLAccess imports are compatible with earlier versions of Portal. Therefore, it is reasonable to expect an XMLAccess script from a previous version of Portal to work in a newer version of Portal.
  - If there is a defect in the XMLAccess import process, you can contact IBM support. However, if the XMLAccess import fails and the failure is not defect related, then IBM Support will not be available to assist with troubleshooting or customizing the XMLAccess import file to make the import successful. Creating customized scripts is beyond the scope of the IBM Support.
  - IBM Services is available if assistance is needed with a manual migration, including customizing the XMLAccess import file.
- You need to manually copy over any files that are required by custom applications, third-party applications, or Portal add-ons that are required for your server to work properly.
- You can use wsadmin and jacl/jython scripts to bring over your custom applications and configuration.
  - The WebSphere Portal Express configurations such as security, performance tuning, datastore configurations, and more are not carried over. You need either a documented or scripted process to re-create these configurations in your new environment.
- You can use XMLAccess to bring over Portal artifacts such as portlets and pages:
  - There might be references to deprecated or unsupported portlets, pages, or features that prevent imports from working properly. You must correct these by removing the references or manually installing the feature if it is still available.
  - It is not recommend to bring any administrative features such as the admin portlets from the source system, but you can bring custom pages and applications.
  - For Virtual Portals, it is recommend to re-create them on the target system, and then manually moving the artifacts over from the source Virtual portal to the target system Virtual Portal.

  - If you customized the Virtual Portal creation scripts, you must redo the script customization for the 8.5 scripts
- You can use cross-version syndication from your source environment to the target environment to bring over JCR content:
  - You must apply the latest combined cumulative fix on your source environment in order to use cross-version syndication to your target environment.
  - You must disable managed pages on the target OOB installation before syndicating any changes.
  - From Version 6.1 and earlier, no cross-version syndication is possible
  - If you are performing a two-step migration from Version 6.1 to Version 8, and then Version 8 to Version 8.5, you must use the Version 8 content refresh task, and then use cross-version syndication to your Version 8.5 environment.
- You must manually bring over anything that is stored in WebDAV:
  - The WebDAV information is stored in JCR database, but is not brought over with cross-version syndication. It must be copied to the target environment manually.
- You lose customization and personalization during a manual migration:
  - There is not a process to preserve and bring this information to a new environment.

Related information:

IBM Support handbook: How technical questions are handled by support

IBM Software Services Zone for WebSphere

Portal farm migration

The Portal farm migration consists of disabling farm mode and running the stand-alone migration. If you are migrating a unique installation farm configuration, you must migrate each farm member. If each farm instance is a clone, you can migrate one instance, and then create clones that are based on the migrated instance. When the migration is complete, you can re-enable farm mode.

Migrating a portal farm

1. Disable farm mode:
   - Deactivate all farm instances that share the required server's file system.
   - Stop sharing the required server's file system.
   - Run the following task to disable farm mode:
     - Linux : `./ConfigEngine.sh disable-farm-mode -DWasPassword=`*`password`* from the *wp_profile_root*`/ConfigEngine` directory
     - IBM i: `ConfigEngine.sh disable-farm-mode -DWasPassword=`*`password`* from the *wp_profile_root*`/ConfigEngine` directory
     - Windows: `ConfigEngine.bat disable-farm-mode -DWasPassword=`*`password`* from the *wp_profile_root*`\ConfigEngine` directory
2. Follow the steps that are covered in the Migrating a stand-alone environment roadmap.

   Note: If you are migrating a unique installation farm configuration, each farm member must be migrated. If you are migrating a clone installation farm configuration, you can migrate one farm member, and then clone the migrated instance.

3. Enable farm mode:
   - The **systemTemp** parameter specifies where the server-specific directory is located. This directory contains all directories and files that the running portal instance writes to, such as for logging and page compiling. Create the target directory path. For example:
     - Linux : `/var/log/was_tmp`
     - IBM i: `/var/log/was_tmp`
     - Windows: `C:\temp\was_tmp`
   - Run the following task to enable farm mode:
     - Linux : `./ConfigEngine.sh enable-farm-mode -DsystemTemp=/var/log/was_tmp -DWasPassword=`*password*
     - IBM i: `ConfigEngine.sh enable-farm-mode -DsystemTemp=/var/log/was_tmp -DWasPassword=`*password*
     - Windows: `ConfigEngine.bat enable-farm-mode -DsystemTemp=C:\temp\was_tmp -DWasPassword=`*password*

Related concepts:

"Roadmap: Migrating a stand-alone server environment" on page 92
Roadmaps provide a high-level overview of complex tasks such as migrating a stand-alone server environment to a new version of IBM WebSphere Portal Express.

Multiple tier environments

When you migrate multiple tier environments, you have two options for completing the migration process. You can choose to migrate each tier independently, or you can migrate the lowest tier and use staging to production techniques to build out the other tiers based on the migrated environment.

Choose from the two options when you are migrating multiple environments, and review the considerations for each:

Migrate tiers using staging to production techniques

The lowest level tier is migrated, and then that tier is moved to the next levels using staging to production techniques. For example, if you have Development, Authoring, and Production tiers, you can migrate the development environment and use staging to production to move that environment to the new authoring and production tier. This option might require less time than migrating tiers independently, but it is imperative that you take a disciplined approach to ensure that all artifacts and settings are properly moved to the next tiers.

Review the considerations for migrating tiers using staging to production techniques:

- Ensure managed pages enablement in the higher-level tiers matches. Migration from a version before Version 8.0 leaves the managed pages feature off. If you build new higher tiers using the Portal installer, then managed pages are turned on.
- Follow the Portal tuning guide to set up all tiers. Migration ensures that all settings in the lowest tier are migrated to the target environment, but since the higher tiers are newly built, they need to be tuned. Tuning might also be necessary on the lowest tier, if the tuning recommendation changed since the previous Portal version.
- Review and decide if you want to use the staging to production documentation to populate the higher tiers. You must ensure that all of

your custom applications and shared libraries are updated on the higher tiers, and plan for a test phase that validates your application before you move to the next tier.

Migrate tiers independently
Each tier is migrated independently. This option might require more time, but it ensures that all settings are migrated to the new environment.

Related concepts:

Chapter 15, "Staging to production," on page 2481
During portal solution development, the solution is initially developed, tested, and refined on one server or a limited number of servers. The solution is deployed later on live systems, referred to as the production environment. The process of moving the solution from the development environment to the production environment is called staging.

Multiple cluster environments

If you are migrating an environment with multiple clusters, you can run the **create-alias-multiple-cluster** task to support multiple clusters that use different credentials.

Supporting multiple clusters that use different database credentials

Using the same database user ID and password for each identically named domain/data source allows the existing JAAS Authentication Aliases to be functional. If a unique database user ID and password are required, extra manual configuration is needed to create new JAAS Authentication Aliases for each data source and map these accordingly.

Complete the following steps on the primary node of Cluster A:

1. Open a command line.
2. Change to the *wp_profile_root*/ConfigEngine directory.
3. Run the following task to create the JAAS Authentication Aliases:
   - Linux : `./ConfigEngine.sh create-alias-multiple-cluster -DauthDomainList=release,jcr -DWasPassword=`*dmgr_password*
   - IBM i: `ConfigEngine.sh create-alias-multiple-cluster -DauthDomainList=release,jcr -DWasPassword=`*dmgr_password*
   - Windows: `ConfigEngine.bat create-alias-multiple-cluster -DauthDomainList=release,jcr -DWasPassword=`*dmgr_password*

   Where **authDomainList** is set to a list of domains that use unique database user ID and passwords and those domain properties are set correctly in the wkplc_comp.properties file, including user ID and password.

Related concepts:

Adding secondary nodes to a clustered environment
The standard process of adding IBM WebSphere Portal Express server nodes to a cluster is to set up a stand-alone WebSphere Portal Express server with the correct database settings, and run the **enable-profiles** task to generate profile templates that can be used to create the secondary node profiles. However, there is a limitation with the enable-profiles task and it cannot be run in a clustered environment.

Migrating from Portal 8.0.0.1 on WebSphere Application Server 8.5.5.2

If you are migrating from IBM WebSphere Portal Express Version 8.0.0.1 on WebSphere Application Server Version 8.5.5.2 to WebSphere Portal Express Version 8.5, you must follow a different migration process.

Review the two different methods that you can use to migrate to Version 8.5 from Version 8.0.0.1 on WebSphere Application Server Version 8.5.5.2.

Use staging to production to set up a staging server, and migrate the staging server to WebSphere Portal Express 8.5.
: You can set up a staging server with a WebSphere Portal 8.0.0.1 and WebSphere Application Server 8.0.0.5 installation, and use the staging to production tools to create a new environment that is based on the source environment. Then, you can use the configuration wizard to migrate the staging environment. The benefit of this method is that your source environment stays in tact on WebSphere Portal Version 8, and it allows for the source and target environments to co-exist. It can require extra hardware, and these steps can take longer to complete.

Manually perform an in-place migration.
: You must manually uninstall and install the Portal binary files, and back up and restore the source Portal profile. This method might be faster, but you must update the source environment in-place, so you must have a backup and a plan for rolling back in case there is a failure. In addition, this procedure cannot be reversed, and when complete the Portal 8.0.0.1 version server is fully updated to WebSphere Portal Version 8.5.

"Using staging to production techniques to complete the migration"
Choose this option if you want to use staging to production techniques to migrate from IBM WebSphere Portal Express Version 8.0.0.1 on WebSphere Application Server Version 8.5.5.2 to WebSphere Portal Express Version 8.5.

"Performing a manual in-place migration" on page 801
If you proceed with this method of migration for IBM WebSphere Portal Express Version 8.0.0.1 on WebSphere Application Server Version 8.5.5.2, be aware that you cannot use the WebSphere remote migration tool because the WebSphere version of the source and target environments are the same. This migration is an in-place migration, and the source environment will no longer be available after the migration is complete.

Using staging to production techniques to complete the migration:

Choose this option if you want to use staging to production techniques to migrate from IBM WebSphere Portal Express Version 8.0.0.1 on WebSphere Application Server Version 8.5.5.2 to WebSphere Portal Express Version 8.5.

About this task

You can set up a staging server with a WebSphere Portal Express 8.0.0.1 and WebSphere Application Server 8.0.0.5 installation, and use the staging to production tools to create a new environment that is based on the source environment.

Procedure

1. Install WebSphere Portal Express 8.0.0.1 and WebSphere Application Server 8.0.0.5 on a staging server with the same operating system as the source server.

2. Use the WebSphere Portal Express staging to production tools to deploy a stand-alone server that is based on the source server.
3. Install the WebSphere Portal Express 8.5 and WebSphere Application Server 8.5.5.2 binary files on the target server.
4. Access the Configuration Wizard on the target server, and click **Migrate to a New Version** > **Migrate a Stand-alone Server**.
5. Optional: When you complete the migration of the stand-alone staging server, you can build it into a cluster if needed.

Related concepts:

"Configuration Wizard" on page 232
Use the Configuration Wizard to set up stand-alone servers and new deployments, create clusters, migrate and update to new versions, and add new capabilities to existing deployments.

Performing a manual in-place migration:

If you proceed with this method of migration for IBM WebSphere Portal Express Version 8.0.0.1 on WebSphere Application Server Version 8.5.5.2, be aware that you cannot use the WebSphere remote migration tool because the WebSphere version of the source and target environments are the same. This migration is an in-place migration, and the source environment will no longer be available after the migration is complete.

About this task

If you are performing an in-place update of a stand-alone server, complete steps 1 - 10. Then, continue with the **upgradeConfigEngine** step of the **Migrate a Stand-alone Server** option in the Configuration Wizard as it is detailed in steps 11 - 16.

If you are performing an in-place update of a cluster, the deployment manager should already have the WebSphere Application Server Version 8.5.5.2 installed. Therefore, you do not need to complete the **Migrate a Cluster Step 1: Migrate the Deployment Manager Profile** option in the Configuration Wizard. You can also skip **Migrate a Cluster Step 2: Migrate Node Profiles** option, but you must complete the following steps 1 - 9 on all nodes. Then, you must complete **Migrate a Cluster Step 3: Upgrade Node Profiles** on all nodes as detailed in steps 11 - 16.

Procedure

1. Back up your databases.
2. Clear out the `temp` and `wstemp` paths in *wp_profile_root*. If you do not complete this step, it can create long path names that prevent the **restoreProfile** task from working successfully.
3. Back up the *wp_profile_root* using `manageprofiles -backupProfile` from `AppServer/bin`. For example: `./manageprofiles.sh -backupProfile -profileName wp_profile -backupFile /tmp/wp_profile_bak`.
4. Uninstall WebSphere Portal Express Version 8 using the IBM Installation Manager. Do not uninstall WebSphere Application Server 8.5.5.
5. Delete the following files from the WebSphere Application Server AppServer path:
   - lib/ext/commons-codec-1.3.jar
   - lib/ext/commons-httpclient-3.0.1.jar
   - lib/ext/openid4java-full-0.9.5.jar

- lib/ext/wp.auth.base.sua_RedirectServletFilter.jar
- lib/ext/wp.auth.base.sua_loginmodule.jar
- lib/ext/wp.auth.tai.jar
- lib/wp.user.connections.jar
- lib/wp.wire.jar
- plugins/com.ibm.patch.was.plugin.jar
- plugins/com.ibm.wp.was.plugin.jar
- plugins/wp.ext.jar
- properties/jndi.properties

6. Ensure that the `wp_profile` and `cw_profile` are cleaned up and the paths are deleted.
7. Install only the Portal 8.5 binary. Do not create a Portal profile.
8. Test connecting to the Configuration Wizard in a browser. `http://your_server:10200/ibm/wizard`
9. Restore the *wp_profile_root* using `manageprofiles restoreConfig` from `AppServer/bin`. For example, `./manageprofiles.sh -restoreProfile -backupFile /tmp/wp_profile_bak`.

   Note: The *wp_profile_root* is restored to the original path.
10. If you are migrating a cluster, copy the `filesForDmgr.zip` in the *PortalServerRoot*`/filesForDmgr` path on your target primary node server to the existing WebSphere Application Server 8.5.5.2 deployment manager. Then, extract the files in the `AppServer` path.

    Note: This task is essential to update the deployment manager to user the portal 8.5 plug-ins. Complete this step once. Do not repeat this step on all nodes.

Complete the remaining steps for using the Configuration Wizard:

11. Access the Configuration Wizard. `http://your_server:10200/ibm/wizard`.
12. Click **Migrate to a New Version** > **Migrate a Stand-alone Server** or **Migrate to a New Version** > **Migrate a Cluster Step 3: Upgrade Node Profiles**.
13. Select **Same server** when you answer questions about your system.
14. Enter your properties with the correct values.
15. Mark all steps complete before **Upgrade the ConfigEngine**.
16. Start your configuration with the **Upgrade the ConfigEngine** step.

Related concepts:

"Configuration Wizard" on page 232
Use the Configuration Wizard to set up stand-alone servers and new deployments, create clusters, migrate and update to new versions, and add new capabilities to existing deployments.

Migration from Portal 7.0 server-only to Portal 8.5

When you migrate from a 7.0 server-only installation to WebSphere Portal Express Version 8.5, you must take extra steps that are not covered in the migration options completed using the Configuration Wizard. Instead of using the wizard to complete the final upgrade the Portal profile step, you must complete this step manually.

To complete a 7.0 server-only migration, follow the steps for stand-alone and cluster migrations that are available in the *Roadmaps for migration*. However, when

you use the Configuration Wizard options to **Migrate to a New Version**, complete all steps from the wizard except for the final **Upgrade the Portal Profile** step. You must run this step manually.

Manually run the two following ConfigEngine tasks from the *wp_profile_root*/ConfigEngine directory:

1. Run the **add-disabled-wcm-to-server** task:
   - Linux : `./ConfigEngine.sh add-disabled-wcm-to-server -DWasPassword=yourpassword -DPortalAdminPwd=yourpassword`
   - IBM i: `ConfigEngine.sh add-disabled-wcm-to-server -DWasPassword=yourpassword -DPortalAdminPwd=yourpassword`
   - Windows: `ConfigEngine.bat add-disabled-wcm-to-server -DWasPassword=yourpassword -DPortalAdminPwd=yourpassword`
2. Run the **upgrade-profile** task including the additional **-Dprevious.family.WPFamilyName=server** parameter:
   - Linux : `./ConfigEngine/ConfigEngine.sh upgrade-profile -DWasPassword=yourpassword -DPortalAdminPwd=yourpassword -javaoption -Xms512m -javaoption -Xmx2048m -Dwcm.transactionTimeout=1200 -Dprevious.family.WPFamilyName=server`
   - IBM i: `ConfigEngine/ConfigEngine.sh upgrade-profile -DWasPassword=yourpassword -DPortalAdminPwd=yourpassword -javaoption -Xms512m -javaoption -Xmx2048m -Dwcm.transactionTimeout=1200 -Dprevious.family.WPFamilyName=server`
   - Windows: `ConfigEngine/ConfigEngine.bat upgrade-profile -DWasPassword=yourpassword -DPortalAdminPwd=yourpassword -javaoption -Xms512m -javaoption -Xmx2048m -Dwcm.transactionTimeout=1200 -Dprevious.family.WPFamilyName=server`

   Note: If you encounter any problems running the **upgrade-profile** task and you need to restart the task using **-Dwp.migration.framework.resume=parameter**, then ensure that you continue to use the **-Dprevious.family.WPFamilyName=server** parameter.

Related concepts:

"Migrate data using the configuration wizard" on page 841
For Version 8.5, data, applications, databases, property files, security settings, and configuration are migrated using the Configuration Wizard. Use the roadmaps for cluster and stand-alone environments to guide you through the process.

Related tasks:

"Roadmaps for migration" on page 92
Choose the appropriate migration roadmap for your environment.

Migrating from Web Content Manager version 7.0 or 8.0

These are the migration options available when migrating from Web Content Manager version 7.0.x or 8.0.x.

"Planning for changes to web content when migrating from Web Content Manager version 7.0" on page 804
Migration projects require careful planning to synchronize changes from the old production system with the new system. Content creation can still continue on the old system while the new system is installed. A typical migration plan includes time for provisioning, configuring, testing, and tuning infrastructure and software. Custom or third-party applications require their own migration procedures.

"IBM Portlet API Web Content Viewer and Remote Web Content Viewer" on page 805
If you use the Web Content Viewer portlet or the Remote Web Content Viewer portlet that are based on the IBM Portlet API, then you must plan for the replacement of those portlets when migrating.

"Cross version syndication" on page 806
Cross version syndication is the preferred method of refreshing web content after an initial migration. The portal migration process is used to migrate web content, however after the initial migration is complete, syndication is used to keep the migrated system synchronized with the older system.

"Managed pages options when migrating" on page 807
When you migrate your IBM WebSphere Portal Express to Version 8.5, here are some considerations with regards to managed pages.

"Library export and import options" on page 807
Web content libraries can be exported from one system and imported onto another. This tool can be used as part of a migration, but there are some limitations to the use of this tool.

Planning for changes to web content when migrating from Web Content Manager version 7.0:

Migration projects require careful planning to synchronize changes from the old production system with the new system. Content creation can still continue on the old system while the new system is installed. A typical migration plan includes time for provisioning, configuring, testing, and tuning infrastructure and software. Custom or third-party applications require their own migration procedures.

In addition to continuous content updates, many organizations plan to restructure their website, or introduce new websites or new sections and to have these changes go live with the migrated system. During migration planning and execution, it is important to understand how to manage structural change at the same time as keeping regular content updates synchronized.

Options for keeping data synchronized during migration

Cross-version syndication

Cross version syndication is supported on these versions:

- From Web Content Manager version 7.0.0.2 with CF26 or higher.
- From Web Content Manager version 8.0.0.1 with CF09 or higher.

On these systems, syndication is used to synchronize web content after an initial migration. Syndication replaces the existing function of a post migration data update. See "Cross version syndication" on page 806 for details.

Web content library export and import

A Web content library export and import can be used during migration. However, there are important limitations which preclude the usage of these tools in most migrations. A Web content library export and import can be used when updates are isolated to a small library on the source system. Export and import should only be used where simple modifications or additions are happening in the source system. A Web content library import must not be used to update content which has changed since the last import.

Syndication strategies

- Use cross-version syndication to keep the syndicator on your new system synchronized with the syndicator on your old system. These are usually your authoring servers.
- Updates that are applicable to both the old and new systems are made on the old syndicator.
- Updates that are only applicable to the new system should be made on the new syndicator.

Note: Any changes made on your old system will override changes made on your new system. If making updates on both systems, it is recommended that the items updated or added to the new system are unique to that system. If not, changes on your new system can be replaced by changes from the old system.

Recent items and favorites

Recent items and favorites are not preserved during migration. Lists of favorite items will need to be recreated post-migration. Users should make note of any favorite items prior to migration.

IBM Portlet API Web Content Viewer and Remote Web Content Viewer:

If you use the Web Content Viewer portlet or the Remote Web Content Viewer portlet that are based on the IBM Portlet API, then you must plan for the replacement of those portlets when migrating.

About this task

Both portlets were removed from Web Content Manager Version 8.5 and are no longer supported. You must replace them with the JSR 286 version of the Web Content Viewer portlet that is available.

If you already use only the JSR 286 version, then you do not need to plan for any additional steps for migration.

Procedure

Check which version you currently have installed on your source environment.

1. Click the **Administration menu** icon. Then, click **Portlet Management** > **Web Modules**.
2. Search for the following files names:

   ilwwcm-localrendering-portlet.war
   This file represents the IBM Portlet API Web Content Viewer portlet and its portlet clones. If you find this web module, you need to perform a conversion for the migration. For more information, see *Converting the IBM Portlet API Web Content Viewer to the JSR 286 Web Content Viewer*.

   ilwwcm-remoterendering-portlet.war
   This file represents the IBM Portlet API Remote Web Content Viewer portlet and its portlet clones. If you find this web module, you need to perform a conversion for the migration. For more information, see *Converting the IBM Portlet API Remote Web Content Viewer to the JSR 286 Web Content Viewer*.

ilwwcm-localrendering-portlet-jsr.war
This file represents the JSR 286 Web Content Viewer portlet and its portlet clones. A conversion of this portlet is not required for the migration.

Related tasks:

"Converting an IBM API Web Content Viewer to the JSR 286 API" on page 874
As installed by default, the Web Content Viewer is based on the JSR 286 API. If you have a Web Content Viewer that is based on the older IBM API, you can convert the viewer to the JSR 286 API. Use the **convert-wcm-rendering-portlet** task to convert the IBM API Web Content Viewer settings and instances to the JSR 286 Web Content Viewer portlet.

Cross version syndication:

Cross version syndication is the preferred method of refreshing web content after an initial migration. The portal migration process is used to migrate web content, however after the initial migration is complete, syndication is used to keep the migrated system synchronized with the older system.

Cross-version syndication is supported between the following releases. Syndication from a newer release to an older release is not supported:

- WebSphere Portal version 7.0.0.2 with CF26 or higher.
- WebSphere Portal 8.0.0.1 with CF09 or higher.
- WebSphere Portal 8.5 or higher.

There are no special steps required to configure syndication between releases. The procedure to enable syndication is the same no matter if the versions differ or not. When an existing subscriber is migrated, the existing pair continues to function normally. There are no additional limits to the kind of items or changes that can be updated between versions, although care must be taken whenever changes are made directly to the subscriber since this carries the risk of creating a conflict that can block items from syndicating.

Fix-pack upgrades

Cross version syndication can also be used to syndicate content between environments for the same release, but on different cumulative fix-packs. This allows individual environments, such as authoring and delivery, to be upgraded separately.

The upgrade can proceed in two ways: syndicator first or subscriber first. It might be preferable to upgrade the system with the least uses first, to lessen the impact of an unintended interruption of service. However the subscriber first approach is preferable because there is a greater degree of compatibility when syndicating from an older software level to a newer level.

- When upgrading a syndicator or subscriber, review the release notes of the new software level to check compatibility with the existing levels the server is syndicating to or subscribing from.
- The goal is to have an entire deployment running at the same level release and fix-pack. This capability is not intended to allow different release and fix-pack levels to coexist indefinitely.
- When syndicating from a newer level to an older level, it might be possible to use features on the syndicator that are not available on the subscriber. This should be avoided since it may result in syndication errors. If a failure is

encountered because of a new feature, the change must be reverted on the syndicator and then syndication of the affected items will resume. New features can be tried before the subscriber is upgraded, by saving content using the new features in a library that is not syndicated to an older software level.

- When a library is syndicated to another release level, that library must not be syndicated back to any server that it has already been syndicated from. Disable any reverse syndicator pairs before attempting a fix-pack upgrade.
- Syndicating from a newer software level to an older software level is only supported between different fix-pack levels of the same release. When syndicating between releases, only older to newer is supported.

  Restriction: CF07 The **Rebuild with mirror** option can only be used when syndicating between servers that use CF07 or higher.

Managed pages options when migrating:

When you migrate your IBM WebSphere Portal Express to Version 8.5, here are some considerations with regards to managed pages.

If you migrate from WebSphere Portal Express Version 8.0 to Version 8.5, the portal preserves the state of managed pages enablement:

- If you had managed pages enabled in your version 8.0 portal, managed pages remain enabled after migration.
- If you had managed pages disabled in your version 8.0 portal, managed pages remain disabled after migration.

If you migrate from WebSphere Portal Express Version 7.0, managed pages are not automatically enabled after migration. In this case, consider enabling managed pages after migration. With managed pages, you can take advantage of new features, such as syndication and versioning of pages, and the ability to manage pages in projects. When you migrate systems that access web content from virtual portals, you need to complete extra steps after you enable managed pages.

Portal Site Library

The Portal Site Library stores page-related web content items. The Portal Site Library, and related views in the authoring portlet, are only visible when managed pages are enabled.

Virtual portal isolation

When managed pages are disabled, web content is shared between all virtual portals. However, when managed pages are enabled, each virtual portal has its own unique and isolated workspace for web content. Before the migrated web content can be accessed from a virtual portal with managed pages, it must be syndicated from the base portal. Syndication between virtual portals is no different from syndication between stand-alone servers.

Library export and import options:

Web content libraries can be exported from one system and imported onto another. This tool can be used as part of a migration, but there are some limitations to the use of this tool.

Important: A library import must not be used to replace previously imported web content library if the previously imported library contains subsequent modifications.

Supported data

Only published and expired web content items are exported. The following data is not supported:

- Draft, deleted, purged web content items are not exported.
- Saved versions of web content items are not exported.
- Theme resources stored in WebDAV are not exported.
- Projects are not exported.

Performance

The library export and import tool is slower than the data migration tool because the library export and import tool creates copies of each item from the web content library. Data migration updates items in the database itself.

Port conflicts

During migration, it is possible that a port conflict might occur when starting up the target environment deployment manager, node agents, or IBM WebSphere Portal Express servers.

If you are completing a local migration, you might encounter a port conflict due to the source and target environments using the same ports. You can manually update the ports in the `serverindex.xml` located in the *`target_wp_profile`*`/config/cells/`*`cellname`*`/nodes/`*`nodename`* path to correct the port conflict issues.

You might also encounter a port conflict with the configuration wizard server. If this occurs, update the configuration wizard ports in the `serverindex.xml` located in *`cw_profile`*`/config/cells/`*`cellname`*`nodes/`*`nodename`*.

Related tasks:

"Using copies of source database domains to minimize downtime" on page 835
To keep the earlier portal environment in production and reduce the amount of downtime during migration copy the earlier portal server JCR and Release domains. Connect to the domain copies and then update the new portal server with the domain copies. The process of connecting to the domain copies must be done after you upgrade the ConfigEngine tool but before you upgrade the Portal profile.

Development considerations

The goal of the migration process is to ensure that the target environment works similarly to the source environment. However, there are deprecated and unsupported features and changes in supported technical specifications that can prevent this transition from being seamless. Review the following topics for guidance on the development work that is required to maintain the functionality of the source environment and also begin preparation for enabling new features and functionality.

"Deprecated features" on page 809
Review the features that were available in previous versions of IBM WebSphere Portal Express but are no longer available.

"Exploitation vs. toleration of applications and themes" on page 809
Toleration is the ability for the new version of IBM WebSphere Portal Express to

support and host the portal site of the source environment exactly the way it was on the previous version. And, exploitation is the enhancement of the source environment site to take advantage of the new functionality that is made available in the new version of Portal and WebSphere.

"Prepare applications and themes" on page 810
After the move to IBM WebSphere Portal Express Version 8.5, it is possible that applications and themes that depend on deprecated features will not function properly. To validate custom applications and themes, it is recommended to set up a basic stand-alone WebSphere Portal Express Version 8.5 server.

"Supported toolbar customization" on page 810
In Version 8.5, only specific customizations are allowed for the IBM WebSphere Portal Express site toolbar. Only the supported toolbar customizations can be migrated to the new version.

"Default changed for JavaServer Faces implementation" on page 811
The default JavaServer Faces (JSF) implementation has changed starting in WebSphere Application Server Version 8.

Deprecated features

Review the features that were available in previous versions of IBM WebSphere Portal Express but are no longer available.

After a migration, features that are no longer supported or are deprecated might result in unpredictable behavior. Remove unsupported and deprecated features before you start your migration. As they become available, links to more information are provided to help you move away from deprecated features. For more information on which features are deprecated for Version 8.5, see "Unsupported and deprecated features for V8.5" on page 17.

Exploitation vs. toleration of applications and themes

Toleration is the ability for the new version of IBM WebSphere Portal Express to support and host the portal site of the source environment exactly the way it was on the previous version. And, exploitation is the enhancement of the source environment site to take advantage of the new functionality that is made available in the new version of Portal and WebSphere.

The goal of the migration process is to carry over the portal site and all of its artifacts from the source environment to the target environment, while keeping everything in tact and functioning as it did in the previous version. Ideally, if everything is migrated perfectly, there will be no difference between the site hosted by the source and target environments other than the updated Portal and WebSphere versions that are hosting the site.

Toleration

The ability to support and host the source environment's portal site exactly the way it was on the previous source version. The migration process automates this process, but complete toleration cannot always be contained due to the adoption of new technical specifications, features, and specifications that are deprecated when updating to a new version. The steps that are not automated and that need to be performed following the migration process are documented in the "Post-migration activities" section of the product documentation. In many cases, deprecated features will be replaced by new features that provide equivalent or enhanced functionality, and these will either be included in the new version of WebSphere Portal Express or available in Content Template Catalog.

Exploitation

The enhancement of the source environment portal site to take advantage of the new functionality that is made available in the new version of Portal and WebSphere. There are many new features in Version 8.5 and since the goal is to replicate the original site during migration, none of the features are enabled by default during the migration process. They will need to be enabled by following the instructions in the "Enabling new features" section of the product documentation.

Related tasks:

"Post-migration activities" on page 857
After you migrate to IBM WebSphere Portal Express Version 8.5, you need to complete extra tasks depending on how you customized the source portal environment and which components you used. First, complete the **Applying the latest combined cumulative fix updates** task, then you can begin the post-migration tasks followed by enabling new functionality.

"Enabling new functionality in a migrated portal" on page 909
The migration process collects configuration data and applications from an earlier installed version of IBM WebSphere Portal Express and merges them into the newer installed version so that the new environment is identical to the earlier environment. Taking advantage of new functionality that was not available in the earlier portal requires additional attention after migration is complete.

Prepare applications and themes

After the move to IBM WebSphere Portal Express Version 8.5, it is possible that applications and themes that depend on deprecated features will not function properly. To validate custom applications and themes, it is recommended to set up a basic stand-alone WebSphere Portal Express Version 8.5 server.

The benefit of setting up a basic stand-alone server for validating the custom applications and themes is that the work of preparing the applications and themes for WebSphere Portal Express Version 8.5 can be done in parallel with migration, which can generally save time in the overall migration process.

This tactic can also be applied to any new custom applications or themes that are developed for Version 8.5. Rather than waiting for the WebSphere Portal Express Version 8.5 server migration to complete, you can begin development of the new applications and themes on a stand-alone server while the migration is taking place.

Supported toolbar customization

In Version 8.5, only specific customizations are allowed for the IBM WebSphere Portal Express site toolbar. Only the supported toolbar customizations can be migrated to the new version.

If you plan on migrating a customized, modularized Version 8.0 theme, review the "Preparing the site toolbar" on page 1900 section to understand what customizations of the toolbar are supported.

After you migrate your data, you can choose to enable the Version 8.5 toolbar, and remove the Version 8.0 toolbar on your modularized Version 8.0 theme. See the "Enabling new functionality" section of the documentation to enable the new toolbar or remove the older toolbar.

Related concepts:

"Migration - Add the version 8.5 site toolbar to a version 8.0 theme" on page 921
You can easily add the modularized site toolbar of WebSphere Portal Express Version 8.5 to a WebSphere Portal Express 8.0 theme. Or, you can add a toolbar to a custom theme that is derived from the WebSphere Portal Express 8.0 theme. The theme must be a modularized theme, which supports theme profiles and theme modules.

Related tasks:

Removing the WebSphere Portal Express 8.0 site toolbar from a WebSphere Portal Express 8.0 theme
To use the new site toolbar of WebSphere Portal Express 8.5 within a WebSphere Portal Express 8.0 theme, you must first remove the existing toolbar that comes with your WebSphere Portal Express 8.0 theme.

Default changed for JavaServer Faces implementation

The default JavaServer Faces (JSF) implementation has changed starting in WebSphere Application Server Version 8.

When you are migrating JSF portlets from an earlier version of IBM WebSphere Portal Express, be aware that WebSphere Application Server has changed the default JSF implementation starting in WebSphere Application Server Version 8. For more information, see *JavaServer Faces migration* in the WebSphere Application Server documentation.

Related information:


JavaServer Faces migration

What to expect after you complete migration

During the migration process, your portal applications, portlets, and databases are updated to the IBM WebSphere Portal Express 8.5 versions. However, not all of the new WebSphere Portal Express Version 8.5 functionality and features are enabled by default. The following sections provide information on how various components are handled during migration, and what you can expect after the migration is complete.

"Themes" on page 812
During the migration process, IBM WebSphere Portal Express moves your theme to your target environment without modifications. The pages in your target environment still reference the same theme that was used in the source environment.

"Site toolbar" on page 812
During migration, the IBM WebSphere Portal Express 8.5 theme is deployed. The 8.5 theme includes the site toolbar, but the toolbar is enabled only for the default virtual portal. For all other virtual portals, you must install the site toolbar separately.

"Virtual portals" on page 813
During the migration process, your existing virtual portals are moved over to your target environment without modification.

"WebSphere Portal and Web Content Manager administration" on page 814
In WebSphere Portal Express 8.5, the update of the IBM WebSphere Portal Express and Web Content Manager administration themes was automated. In previous versions of WebSphere Portal Express, the WebSphere Portal Express and Web Content Manager administration themes were not updated during migration.

"Page order" on page 814
If the page order was altered for IBM pages in the Applications,

Administration, or Hidden Pages areas on your source environment, then the page order might not be correct on the target environment after migration.

Themes

During the migration process, IBM WebSphere Portal Express moves your theme to your target environment without modifications. The pages in your target environment still reference the same theme that was used in the source environment.

If you are using standard portal pages, you must convert your pages to static pages. Then, create an WebSphere Portal Express Version 8.5 theme and the layouts that are required for your pages to take take advantage of the WebSphere Portal Express Version 8.5 theme features.

For more information about preparing and optimizing your applications and themes for migration, read the planning section of the product documentation. Also, see the following related links for information about developing themes and skins, troubleshooting modular themes, working with layouts, and converting standard portal pages to static pages by using the IBM WebSphere Portal Express Page Migration Tool.

Related concepts:

"Developing themes and skins" on page 2520
You can create themes using modules to contribute to separate areas of pages to provide flexibility, enhance the user experience, and maximize performance. To optimize themes on your website, use the theme optimization module framework. The framework separates feature-specific logic and capabilities from the theme code.

"Troubleshooting modular themes" on page 2600
You can debug your modules to improve performance.

"Layouts" on page 2677
You can apply ready-use layouts to your portal pages, modify the existing skins, or add your own custom layout to change how your pages display.

Related information:

IBM WebSphere Portal Page Migration Tool

Site toolbar

During migration, the IBM WebSphere Portal Express 8.5 theme is deployed. The 8.5 theme includes the site toolbar, but the toolbar is enabled only for the default virtual portal. For all other virtual portals, you must install the site toolbar separately.

The site toolbar contains artifacts that are not scoped to virtual portals. The global menus, such as the site menu, applications menu, and the administration menu, are displayed in virtual portals. However, the full site toolbar menu options, such as opening the toolbar, the project selector, as well as the edit mode switch, are hidden. To enable the full site toolbar, you must install and enable the toolbar for the respective virtual portal.

Related tasks:

"Enabling the 8.5 site toolbar" on page 878
During migration, the IBM WebSphere Portal Express Version 8.5 theme is deployed. The 8.5 theme includes the site toolbar, but the toolbar is enabled only for the default virtual portal. For all other virtual portals, including migrated and newly created virtual portals, you must install the site toolbar separately.

Virtual portals

During the migration process, your existing virtual portals are moved over to your target environment without modification.

The default sample virtual portal content scripts are not updated during migration. A new sample virtual portal script is available with each new version of WebSphere Portal Express. However, this script is not installed or made available during migration. If you create new virtual portals by using the Virtual Portal Manager portlet in the portal Administration area, the portal continues to use your existing virtual portal content scripts. If you want to create new virtual portals by using new scripts that you create based on the updated samples, configure the Virtual Portal Manager portlet to use these new scripts.

For more information about working with virtual portals, read the *Virtual portals* in the related links.

Adding a portal administrator to virtual portal realms

When you migrate an environment with virtual portals, the portal administrator must be added to the virtual portal realms. If the portal administrator is not added to the realms associated with the virtual portals, the migration fails.

Add the portal administrator to a virtual portal realm:

1. Enter a value for the following parameters in the `wkplc.properties` file in the VMM realm configuration section:
   - **`realmName=`***realmName*
   - **`addBaseEntry=`***o=BaseEntryName*
2. Open a command line.
3. Change to the *wp_profile_root*`/ConfigEngine` directory.
4. Run the following command:
   - AIX HP-UX Linux Solaris: `./ConfigEngine.sh wp-add-realm-baseentry -DWasPassword=`*`password`*
   - IBM i: `ConfigEngine.sh wp-add-realm-baseentry -DWasPassword=`*`password`*
   - Windows: `ConfigEngine.bat wp-add-realm-baseentry -DWasPassword=`*`password`*

Related concepts:

"Virtual portals" on page 1361
View information on how you can scope your WebSphere Portal Express to have multiple virtual portals.

Related tasks:

"Adding realm support" on page 582
A realm is a group of users from one or more user registries that form a coherent group within IBM WebSphere Portal Express. Realms allow flexible user management with various configuration options. A realm must be mapped to a Virtual Portal to allow the defined users to log in to the Virtual Portal. When you configure realm support, complete these steps for each base entry that exists in your LDAP and database user registry to create multiple realm support.

Related reference:

"Pre-configuring the default content for virtual portals" on page 1400
When you create the virtual portal by using the Virtual Portal Manager portlet, the virtual portal is pre-filled with default content. This default content is determined by the default XML script file for initializing virtual portals.

Related information:

"Tasks for administering virtual portals" on page 1389
Administering virtual portals and their content comprises the tasks described in the following topics.

WebSphere Portal and Web Content Manager administration

In WebSphere Portal Express 8.5, the update of the IBM WebSphere Portal Express and Web Content Manager administration themes was automated. In previous versions of WebSphere Portal Express, the WebSphere Portal Express and Web Content Manager administration themes were not updated during migration.

For more information about administering WebSphere Portal Express, read the *Administering* section of the product documentation.

Related concepts:

Chapter 11, "Administering," on page 1041
Use the administration tools that are provided with the portal to do various day-to-day administration tasks. There are two methods for editing portal setup: using the administration portlets or the XML configuration interface. The administration portlets are a convenient way to make real-time updates to the portal's configuration. While the XML configuration interface is suited to more advanced administration, including batch processing of updates.

Page order

If the page order was altered for IBM pages in the Applications, Administration, or Hidden Pages areas on your source environment, then the page order might not be correct on the target environment after migration.

You can adjust the page order of your pages using the Portal Administration menu or XMLAccess. If you want your pages to be listed first, then ensure that they have ordinal values less than 100. For more information, see the post-migration step for "Updating page order" on page 869.

Related tasks:

"Updating page order" on page 869
If the page order was altered for IBM pages in the Applications, Administration, or Hidden Pages areas on your source environment, then the page order might not be correct on the target environment after migration.

Preparing your source environment

Before you begin the steps for migration, you must perform some critical tasks on your source environment, such as creating back ups, installing the latest cumulative fix and one of the two most recent fix packs, and disabling automatic synchronization if you are migrating a cluster. Review the topics in this section, and perform the required tasks to ensure that your source environment remains functional and the migration completes successfully.

Before you begin

Important: If you are using *locahost* in the database URL, update the URL to use the actual database server host name before you continue with your migration.

For more details on preparing your source environment, review the following topics in this section and the Roadmaps for migration.

"Install fix packs on the source environment" on page 816
Periodically fix packs are released to integrate product code fixes. Between fix

pack releases, there are interim fixes to ensure product reliability and stability. You must apply the latest cumulative fix to the source environment. The *Recommended fixes* link provides links to fix pack and interim fix downloads. There is also information about what is recommended and what is required.

"Verifying property files" on page 816
Ensure that the existing portal environment is at the appropriate service level for migration.

"Backing up the system" on page 816
Make sure that you have a current backup and recovery policy before you alter the source environment. Follow the instructions in the Backup and restore section to ensure that you cover all the affected assets.

"Disabling automatic synchronization to protect your clustered source environment" on page 817
The target environment initially uses the same ports as the source environment. There are three important steps you must complete to ensure that the source and target environments do not become corrupted.

"Verifying that WebSphere Application Server Trust Association Interceptor is enabled" on page 818
The automated migration of the WebSphere Portal Express profile requires that the Trust Association Interceptor (TAI) is enabled so that you can configure content in WebDAV during migration.

"Preparing Web Content Manager content" on page 819
To migrate IBM Web Content Manager data, you must remove the locks on content and rename the content libraries.

"Migrating search components" on page 820
The search components in your source portal might require preparation steps and then extra steps on the target portal.

"Migrating from a 32-bit source environment to 64-bit target environment" on page 825
The 32-bit Portal installation is no longer supported in IBM WebSphere Portal Express Version 8.5. If you are migrating from a 32-bit source environment to a 64-bit Version 8.5 environment, you need to take extra steps to ensure that the **WASPreMigration** task completes successfully.

"Prepare UX Screen Flow Manager" on page 825
If you are migrating from Version 8.0.0.1 with the UX Screen Flow Manager (UXFM) enabled, then you must remove the dialog definitions, and then uninstall UXFM before you migrate to Version 8.5. Before you remove the dialog definitions and uninstall UXFM, it is highly recommended that you export your dialog definitions, and when migration is complete, you can import your dialog definitions into your upgraded system. If you do not export your dialog definitions, then your data will be lost.

"Removing WebSphere Commerce integration" on page 828
If you have WebSphere Commerce integrated with IBM WebSphere Portal Express, you must remove WebSphere Commerce from your source portal profile before you begin migration.

"Removing unsupported composite applications" on page 828
Composite applications are no longer supported. If you have a composite application in your system and you migrate to Version 8.5, the migration fails. Ensure that all composite applications are deleted before you start the migration. When you delete a composite application, you must also run the resource cleaner, otherwise pages can still exist in the database.

"Removing obsolete portlets from virtual portal scripts" on page 829
Some portlets were deprecated or removed in this release. If you have

references to deprecated or removed portlets in your virtual portal scripts, you must manually remove those references.

"Distinguished names" on page 829
If you are using LDAP in your source environment, make sure that the `wkplc.properties` file is properly configured. You might have a configuration that is working, but it might not be supported after migration. Short distinguished names (DN) are not supported. Make sure that the properties files in your source environment are set with the fully qualified distinguished names.

"Maximum open file descriptors for Unix-based platforms" on page 830
For Unix-based platforms, the default open file descriptor must be set to 200000 to allow the configuration wizard commands to run properly during migration.

"Disabling wsadmin client debug" on page 831
If you use either IBM WebSphere Application Server Version 8.5.5.4 or WebSphere Application Server Version 8.5.5.5, disable the wsadmin client trace to avoid a failure.

Install fix packs on the source environment

Periodically fix packs are released to integrate product code fixes. Between fix pack releases, there are interim fixes to ensure product reliability and stability. You must apply the latest cumulative fix to the source environment. The *Recommended fixes* link provides links to fix pack and interim fix downloads. There is also information about what is recommended and what is required.

Related information:


Recommended Updates for WebSphere Portal and IBM Web Content Manager

Verifying property files

Ensure that the existing portal environment is at the appropriate service level for migration.

Procedure

1. Open a command prompt and change to the *wp_profile_root*`/ConfigEngine` directory.
2. Run the following command to ensure that the `wkplc_comp.properties` file contains the correct information:
   - Linux : `./ConfigEngine.sh validate-database-connection`
   - IBM i: `ConfigEngine.sh validate-database-connection`
   - Windows: `ConfigEngine.bat validate-database-connection`

Backing up the system

Make sure that you have a current backup and recovery policy before you alter the source environment. Follow the instructions in the Backup and restore section to ensure that you cover all the affected assets.

About this task

Before you start the migration, make a fresh backup of your environment. At a minimum, you must back up the following information:

- Databases
- Directory structures of the source environment
- Security configurations

Related concepts:

Chapter 8, "Backup and restore," on page 771
Backup and recovery of data files and databases is an essential operation for any business system, particularly for data and applications that run in production environments. Create and follow a plan for backing up and recovering data on all tiers of your IBM WebSphere Portal Express deployment. IBM Installation Manager must also be included in backup and recovery planning. If you back up the WebSphere Portal Express file structure and then install a fix pack, the WebSphere Portal Express and IBM Installation Manager become out of sync after you restore the WebSphere Portal Express file system. This condition is not recoverable.

Disabling automatic synchronization to protect your clustered source environment

The target environment initially uses the same ports as the source environment. There are three important steps you must complete to ensure that the source and target environments do not become corrupted.

Procedure

1. Disable automatic synchronization on all nodes in the cluster.

   In a clustered environment, turn off automatic node synchronization before you start the migration. When the automatic synchronization is enabled, the node agent on each node automatically contacts the deployment manager every synchronization interval. As you migrate nodes, some of your migration-specific configuration for a node might get replicated in your clustered environment. For this reason, it is preferable to disable the automatic synchronization and manually sync.

   a. Start the WebSphere Integrated Solutions Console.
   b. Select **System Administration** > **Node Agents** in the navigation tree.
   c. Click **nodeagent** for the required node.
   d. Click **File Synchronization Service** under the **Additional Properties** section.
   e. Clear the **Enable service at server startup** check box selection to disable the synchronization service at startup.
   f. Clear the **Automatic Synchronization** check box selection to disable the automatic synchronization feature.
   g. Click **OK** and **Save**.
   h. Repeat these steps for all remaining nodes.
   i. Select **System Administration** > **Nodes** in the navigation tree.
   j. Select all nodes that must be manually synchronized, and click **Synchronize**.
   k. Select **System Administration** > **Node Agents** in the navigation tree.
   l. For the primary node, select the node agent and click **Restart**.
2. Stop the source deployment manager and node agents before you start your cluster migration.

   This step is required to ensure that the source and target environments do not become corrupted during the migration process. The application servers can continue to run, but administration is disabled until the ports for the target environment are changed in Migrate a Cluster Step 3: Upgrade nodes. Ensure that the source deployment manager and node agents do not start until you complete Migrate a Cluster Step 2: Migrate node profiles.

   a. Open the WebSphere Integrated Solutions Console.
   b. Click **System administration** > **Node agents**.

c. Select the check box for the node agents and click **Stop**.
d. Click **System administration** > **Deployment manager**.
e. On the **Configuration** tab of the deployment manager settings, click **Stop**.

3. Update the ports on the target environment.

   You will update the ports for the target environment in a step that is detailed in Migrate a Cluster Step 3: Upgrade node profiles. After you update the ports, the source deployment manager and node agents can be started.

Verifying that WebSphere Application Server Trust Association Interceptor is enabled

The automated migration of the WebSphere Portal Express profile requires that the Trust Association Interceptor (TAI) is enabled so that you can configure content in WebDAV during migration.

Procedure

Verify that TAI is enabled:

1. Log in to the WebSphere Integrated Solutions Console.
2. Go to **Security** > **Global security**.
3. Ensure that **Enable administrative security** and **Enable application security** are selected.
4. In the **Authentication** section, expand **Web and SIP security**. Click **Trust association**.
5. Ensure that **Enable trust assocation** is selected.

If TAI is not enabled, complete the following steps:

6. Open a command line and change to the directory where WebSphere Portal Express ConfigEngine is installed, on the corresponding operating system:
   - Linux: *wp_profile_root*`/ConfigEngine`
   - IBM i: *wp_profile_root*`/ConfigEngine`
   - Windows: *wp_profile_root*`\ConfigEngine`
7. Enter the following command:
   - Linux: `ConfigEngine.sh enable-http-basic-auth-tai-sitemgmt -DPortalAdminPwd=`*`password`* `-DWasPassword=`*`password`*
   - IBM i: `ConfigEngine.sh enable-http-basic-auth-tai-sitemgmt -DPortalAdminPwd=`*`password`* `-DWasPassword=`*`password`*
   - Windows: `ConfigEngine.bat enable-http-basic-auth-tai-sitemgmt -DPortalAdminPwd=`*`password`* `-DWasPassword=`*`password`*

   Use `-DPortalAdminPwd=`*`password`* `-DWasPassword=`*`password`* to specify the portal and WebSphere Application Server passwords.

   Note: This task uses the settings in the file `wkplc_comp.properties` to configure the TAI. Although the TAI settings are pre-configured to work without requiring adjustment, you can change the settings before you run the task if you need to configure the TAI differently.
8. Stop and restart the portal.
9. Optional: Perform this step if you have SSL configured. Establish trust between two WebSphere cells:
   a. For preparation, determine the URL to the administrative console of the client WebSphere cell. For example, the URL can be similar to `https://myclientserver.yourco.com:9043/ibm/console`.

b. Open the administrative console by using the URL that you obtained by the previous step.
c. Click **Security** > **SSL certificate and key management** > **Key stores and certificates**.
d. On the keystores and certificates panel click **CellDefaultTrustStore** or **NodeDefaultTrustStore**, depending on whether you have a cluster or single node configuration.
e. On the *xxx*DefaultTrustStore panel, locate the column Additional properties and click **Signer certificates**.
f. On the Signer certificates panel, click the button **Retrieve from port**.
g. Complete the fields and select the options as follows:

   Host The host name of the client server, for example `your_target_server.your_co.com`.

   Port The secure port on the client server, for example `9043`.

   SSL configuration for outbound connection
   Select the SSL configuration for the outbound connection, such as `CellDefaultSSLSettings` or `NodeDefaultSSLSettings`.

   Alias The alias name, for example `name_of_your_alias`.
h. Select **Retrieve signer information**. The signer information is displayed.

   Note: The error message `CWPKI0661E: Unable to get certificate signer information from host name "yourtargetserver.yourco.com" and port "9043". Verify host name and port are correct` might appear for one of two reasons:
   - A certificate has already been imported from the target location.
   - A previously deleted certificate has not timed out and been removed.
i. Click **OK**. Your alias is now shown in the list.
j. Click **Save**.
k. Stop and restart the portal.
l. Optional: At this time, if you have a clustered environment without automatic synchronization, you need to resynchronize the node agents.

Related tasks:

"Disabling TAI if disabled previously" on page 864
If TAI was disabled before you began migration, and you had to enable it in order to run the migration, then you might need to disable TAI on both the source and target environments as a post-migration step.

Preparing Web Content Manager content

To migrate IBM Web Content Manager data, you must remove the locks on content and rename the content libraries.

"Removing locks" on page 820
Before migrating, it is recommended that locks are removed on Web content items.

"Renaming web content libraries" on page 820
If the name of a web content library is the same as the URL context of a virtual portal, you can experience incorrect rendering behavior. To prevent this issue, rename the library to a different name before you start the migration.

"Preventing items from being expired during migration"
By default, items with no expiration date defined are automatically expired. Change this setting to prevent automatic expiration from occurring.

Removing locks

Before migrating, it is recommended that locks are removed on Web content items.

Procedure

1. Log on to IBM WebSphere Portal Express.
2. Click the **Administration menu** icon. Then, click **Portal Content** > **Web Content Libraries**.
3. Click **View locked items**.
4. Select all items, and then click **Unlock**.

Renaming web content libraries

If the name of a web content library is the same as the URL context of a virtual portal, you can experience incorrect rendering behavior. To prevent this issue, rename the library to a different name before you start the migration.

Procedure

1. Log in to IBM WebSphere Portal Express.
2. Click the **Administration menu** icon. Then, click **Portal Content** > **Web Content Libraries**.
3. For the library that you want to rename, click the **Edit Library** icon.
4. Enter a new name for the library that is distinct from the URL context of the virtual portal.

Preventing items from being expired during migration

By default, items with no expiration date defined are automatically expired. Change this setting to prevent automatic expiration from occurring.

1. Log in to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Select **WCM WCMConfigService.**
4. In the **Additional Properties** section, select **Custom Properties**.
5. Locate the `expire.blankdate.immediately` setting.
   - If the setting does not exist, add it by clicking **New**. Enter `expire.blankdate.immediately` for **Name** and `false` for **Value.** Click **OK** to save the changes.
   - If the setting does exist, ensure that it is set to `false`. Click to edit the setting. Enter `false` for **Value.** Then, click **OK** to save the changes.

When you click **OK**, a message box appears that prompts you either to save your changes directly to your master configuration or to review your changes before you save them. Click **Save** to save your changes directly or **Review** to review them before saving.

Migrating search components

The search components in your source portal might require preparation steps and then extra steps on the target portal.

"Migrating portal search collections" on page 821
When you migrate or upgrade IBM WebSphere Portal Express to a later version, the data storage format and index structure of Portal Search is not

compatible with an earlier version. If you migrate your portal to a later version and want to continue using your search collections, you must preserve them before you migrate your portal and import them into the upgraded portal after the migration.

"Migrating web search collections" on page 823
You migrate web search collections to IBM WebSphere Portal Express Version 8.5 by exporting each web search collection from the earlier portal and then importing the web search collection into the new portal. You can also use the same functionality to move web search collections to a production portal after verifying them on a test portal, or to move web search collections to a configuration with remote search after verifying them locally on a portal.

"Migrating a remote search server" on page 824
If the source portal environment uses a remote search server, update the remote search server to work with the current portal environment.

"Removing JCR search collections" on page 824
In your source environment, there are JCR search collections that are prefixed with JCRCollection. Although search collections are covered in a separate section, JCRCollection prefixed collections must be handled differently.

Migrating portal search collections

When you migrate or upgrade IBM WebSphere Portal Express to a later version, the data storage format and index structure of Portal Search is not compatible with an earlier version. If you migrate your portal to a later version and want to continue using your search collections, you must preserve them before you migrate your portal and import them into the upgraded portal after the migration.

About this task

Search collections require manual migration; use the **Import or Export Collection** option of the Manage Search portlet to export and import the search collections. For more information about these tasks and the Manage Search portlet, see the portlet help.

Procedure

1. To include the security information when you export the search collection, add the **WS_KEY** parameter to the search service that contains the source search collection that you want to export. Complete the following steps:
   a. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
   b. Click **Search Services**.
   c. Click the **Edit** icon for the search service that contains the search collection that you want to export.
   d. In the **Parameter key** field, enter `WS_KEY`.
   e. In the **New parameter value** field, enter `secret`.
   f. Click **Add Parameter**.
   g. Click **OK**.

   Note: If you do not export the security information when you export a search collection, you must manually add the user name and password to each content source after you import the search collection into the target portal.
2. Before you migrate your portal to a later version, export your search collections. This step exports the configuration data of your search collections.
   a. Click **Manage Search.**

b. Click **Search Collections**.
c. Click the **Import or Export Collection** icon for the collection that you want to export.
d. In the **Specify Location** field, enter the full directory path and XML file name to which you want to export the document collection and its data.
e. Click **Export**.

Notes:

a. Before you export a collection, make sure that the user who is running the portal application process has write access to the target directory location. Otherwise, you might get an error message, such as `File not found`.
b. When you specify the target directory location for the export, be aware that the export overwrites files in that directory.

3. For each collection, document the following data:
   - The target file names and directory locations to which you export the collection.
   - Location, name, description, and language.
   - Settings for the **Specify collection language** and **Remove common words from queries** options.
4. Delete the search collections from your existing portal. Otherwise, they can be corrupted by the import step that follows later.
5. Upgrade your WebSphere Portal Express as needed.
6. Create empty search collections that you can use later to hold the imported collections configuration. Complete the following fields and select the following options according to the information that you documented in step 2:

   Location of Collection
   The location can match the old setting, but does not have to match it.

   Name of Collection
   The name can match the old setting, but does not have to match it.

   Description of Collection
   The description can match the old setting, but does not have to match it.

   Specify Collection Language
   Select this field to match the old setting as documented in step 2.

   Select Summarizer
   The value is overwritten by the import process.

   Remove common words from queries (for example. in, of, on, and so on)
   Check or clear this setting to match the old setting as documented in step 2.

   You do not have to add content sources or documents, as that is completed by the import process.
7. To include the security information when you import the search collection, add the **WS_KEY** parameter to the search service that contains the target search collection. Complete the following steps:
   a. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
   b. Click **Search Services**.

c. Click the **Edit** icon for the search service that contains the target search collection.
d. In the **Parameter key** field, enter `WS_KEY`.
e. In the **New parameter value** field, enter `secret`.
f. Click **Add Parameter**.

Note: If you do not import the security information when you import a search collection, you must manually add the user name and password to each content source after you import the search collection into the target portal.

8. Check that the target search collections that you created in step 6 are empty. Do not import collection data into a target collection that already contains sources or documents.
9. Import the search collection data into the portal. For the import source information, use your documented file names and directory locations to which you exported the collections before the portal upgrade.
10. Briefly review the content source's configuration settings to check whether information such as host name and security credentials are still valid or must be modified. Make any necessary changes to the content source's configuration settings.

    Note: When you import search collection data into a collection, most of the configuration data such as content sources, schedulers, filters, and language settings are also imported. If you configured such settings when you created the collection, they are overwritten by the imported settings.

Migrating web search collections

You migrate web search collections to IBM WebSphere Portal Express Version 8.5 by exporting each web search collection from the earlier portal and then importing the web search collection into the new portal. You can also use the same functionality to move web search collections to a production portal after verifying them on a test portal, or to move web search collections to a configuration with remote search after verifying them locally on a portal.

"Exporting search web collections"
Use the Manage Search portlet to export search web collections from a source portal. Before you export a collection, make sure that the user who is running the portal application process has write access to the target directory location.

Exporting search web collections:

Use the Manage Search portlet to export search web collections from a source portal. Before you export a collection, make sure that the user who is running the portal application process has write access to the target directory location.

Procedure

1. To include the security information when you export the search collection, add the **WS_KEY** parameter to the search service that contains the source search collection that you want to export. Complete the following steps:
   a. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
   b. Click **Search Services**.
   c. Click the **Edit** icon for the search service that contains the search collection that you want to export.

d. In the **Parameter key** field, enter `WS_KEY`.
e. In the **New parameter value** field, enter `secret`.
f. Click **Add Parameter**.
g. Click **OK**.

Note: If you do not export the security information when you export a search collection, you must manually add the user name and password to each content source after you import the search collection into the target portal.

2. Click **Manage Search.**
3. Click **Search Collections**.
4. Click the **Import or Export Collection** icon for the collection that you want to export.
5. In the **Specify Location** field, enter the full directory path and XML file name to which you want to export the document collection and its data.
6. Click **Export**

Related tasks:

"Importing search web collections" on page 883
As a part of preparing your source environment, you exported web collections. After you export a search web collection from a source portal, you can import the data into a new, empty collection on the target portal. Importing a web collection retains most of the configuration data such as content sources, schedulers, filters, and language settings. If you configured such settings when creating the new collection, they are overwritten by the imported settings.

Migrating a remote search server

If the source portal environment uses a remote search server, update the remote search server to work with the current portal environment.

Procedure

1. Update IBM WebSphere Application Server on the remote search server to the supported version.

   To avoid port conflicts, ensure that the earlier installed version of WebSphere Application Server is up and running. Then install the current version of WebSphere Application Server to a different directory. After installation, check that both the earlier version and the newer version are running, and then continue configuring the remote server. For more information, see *Using remote search service*.
2. On the migrated portal, re-create the search collections for use with the updated remote search server.

Removing JCR search collections

In your source environment, there are JCR search collections that are prefixed with JCRCollection. Although search collections are covered in a separate section, JCRCollection prefixed collections must be handled differently.

About this task

When you migrate search collections, you usually export and delete them from the source environment. Then, proceed with the migration. Finally, import them into the target environment. These steps are covered in detail in the *Migrating Portal search collections* section.

For **JCRCollection** prefixed collections, the process is simpler. You do not need to export, migrate, and import later. Instead, you must delete the **JCRCollection** prefixed search collections from your source environment before you start with the migration. Migration automatically re-creates the JCR search collection as part of the post-migration activities.

Migrating from a 32-bit source environment to 64-bit target environment

The 32-bit Portal installation is no longer supported in IBM WebSphere Portal Express Version 8.5. If you are migrating from a 32-bit source environment to a 64-bit Version 8.5 environment, you need to take extra steps to ensure that the **WASPreMigration** task completes successfully.

About this task

The binary files that are used to run the **WASPreUpgrade** task on the source server are typically created on the target environment. However, the remote migration package includes a Java JRE that matches the 64-bit architecture of the target environment, and it cannot be run on the source environment. If you attempt to use the 64-bit remote migration package on the 32-bit source environment, you are prompted with an error message that states that the 64-bit JVM cannot run on 32-bit hardware.

Choose one of the following methods to resolve this issue:

1. Change the JAVA_HOME path that is used by the remote migration package to use the Java installed with source Portal server:
   a. Create the remote migration package as directed by the migration instructions in the Configuration Wizard.
   b. Copy the remote migration package to the source environment, and extract.
   c. Edit the *extracted remote migration package root*/bin/sdk/_setupsdk1.6_64.sh file, and update the **JAVE_HOME** property to be in the *Source AppServer root*/java path of the source environment.
   d. Continue with the rest of the migration process in the Configuration Wizard.
2. Copy the JRE from the source environment to the remote migration package:
   a. Create the remote migration package as directed by the migration instructions in the Configuration Wizard.
   b. Copy the remote migration package to the source environment, and extract.
   c. Move the Java folder from the remote migration package that is in *extracted remote migration package root*/java to another location.
   d. Copy the Java folder from your source that is in *Source AppServer root*/java to the Java folder in the remote migration package.
   e. Continue with the rest of the migration process in the Configuration Wizard.

Prepare UX Screen Flow Manager

If you are migrating from Version 8.0.0.1 with the UX Screen Flow Manager (UXFM) enabled, then you must remove the dialog definitions, and then uninstall UXFM before you migrate to Version 8.5. Before you remove the dialog definitions and uninstall UXFM, it is highly recommended that you export your dialog

definitions, and when migration is complete, you can import your dialog definitions into your upgraded system. If you do not export your dialog definitions, then your data will be lost.

1. "Exporting UXFM dialog definitions"
   Export and save your dialog definitions before you migrate to Version 8.5.
2. "Removing UXFM dialog definitions" on page 827
   Remove the UX Screen Flow Manager (UXFM) dialog definitions before uninstalling UXFM. This is a required step before you uninstall UXFM.
3. "Uninstalling UX Screen Flow Manager" on page 827
   If you are migrating from Version 8.0.0.1 with the UX Screen Flow Manager (UXFM) enabled, then you must uninstall UXFM before migrating to Version 8.5.

Exporting UXFM dialog definitions

Export and save your dialog definitions before you migrate to Version 8.5.

Procedure

1. Save the following example of code as *ExportSampleCode.xml* to use when exporting your dialog definitions.

```
<?xml version="1.0" encoding="UTF-8"?>
<request type="update" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="PCM_1.0.xsd">

    <!-- export all dialog definitions that are currently deployed -->
    <portal action="export">
        <dialog-set>
            <dialog name="*" />
        </dialog-set>
    </portal>
</request>
```

2. Run the following task from the *wp_profile_root*/PortalServer/bin directory to export your dialog definitions:
   - Linux : ./xmlaccess.sh -user *userID* -password *password* -url *http://local_host:local_port/wps/config* -in *ExportSampleCode.xml* -out export.xml
   - IBM i: xmlaccess.sh -user *userID* -password *password* -url *http://local_host:local_port/wps/config* -in *ExportSampleCode.xml* -out export.xml
   - Windows: xmlaccess.bat -user *userID* -password *password* -url *http://local_host:local_port/wps/config* -in *ExportSampleCode.xml* -out export.xml
3. Back up the file that contains the exported dialog definitions to a secure location. You can import your dialog definitions after you migrate to V8.5.

Related tasks:

"Importing UX Screen Flow Manager dialog definitions" on page 884
If you migrated from Version 8.0.0.1 with the UX Screen Flow Manager (UXFM) enabled, then you exported and removed your dialog definitions before migrating to Version 8.5. Run the following task to import the dialog definitions into your upgraded system.

"Uninstalling UX Screen Flow Manager" on page 827
If you are migrating from Version 8.0.0.1 with the UX Screen Flow Manager (UXFM) enabled, then you must uninstall UXFM before migrating to Version 8.5.

Related reference:

"Syntax elements for the XML configuration interface command line" on page 1067
This topic lists the syntax elements for using the XML configuration interface command line client over an HTTP connection.

Removing UXFM dialog definitions

Remove the UX Screen Flow Manager (UXFM) dialog definitions before uninstalling UXFM. This is a required step before you uninstall UXFM.

Procedure

1. Save the following example of code as *RemoveSampleCode.xml* to use to remove your dialog definitions.

```
<?xml version="1.0" encoding="UTF-8"?>
<request type="update" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="PCM_1.0.xsd">

    <!-- remove all dialog definitions that are currently deployed -->
    <portal action="delete">
       <dialog-set>
            <dialog name="*" />
        </dialog-set>
    </portal>
</request>
```

2. Run the following task from the *wp_profile_root*/PortalServer/bin directory to remove your dialog definitions:
   - Linux : `./xmlaccess.sh -user userID -password password -url http://local_host:local_port/wps/config -in RemoveSampleCode.xml -out delete.xml`
   - IBM i: `xmlaccess.sh -user userID -password password -url http://local_host:local_port/wps/config -in RemoveSampleCode.xml -out delete.xml`
   - Windows: `xmlaccess.bat -user userID -password password -url http://local_host:local_port/wps/config -in RemoveSampleCode.xml -out delete.xml`

Related reference:

"Syntax elements for the XML configuration interface command line" on page 1067
This topic lists the syntax elements for using the XML configuration interface command line client over an HTTP connection.

Uninstalling UX Screen Flow Manager

If you are migrating from Version 8.0.0.1 with the UX Screen Flow Manager (UXFM) enabled, then you must uninstall UXFM before migrating to Version 8.5.

Before you begin

You must remove the dialog definitions before you uninstall UXFM. If you do not remove the dialog definitions prior to completing the uninstall, it can have a catastrophic effect on your system rendering it unusable.

Procedure

1. If UXFM portlets are installed, run the following task:
   - Linux : `./ConfigEngine.sh action-undeploy-pcm-portlets`
   - IBM i: `ConfigEngine.sh action-undeploy-pcm-portlets`
   - Windows: `ConfigEngine.bat action-undeploy-pcm-portlets`
2. Stop the Portal Server.

3. Run the following task:
   - Linux : `./ConfigEngine.sh uninstall-pcm`
   - IBM i: `ConfigEngine.sh uninstall-pcm`
   - Windows: `ConfigEngine.bat uninstall-pcm`

Related tasks:

"Exporting UXFM dialog definitions" on page 826
Export and save your dialog definitions before you migrate to Version 8.5.

Related information:

IBM UX Screen Flow Manager: Installation and setup

Removing WebSphere Commerce integration

If you have WebSphere Commerce integrated with IBM WebSphere Portal Express, you must remove WebSphere Commerce from your source portal profile before you begin migration.

About this task

Complete the following steps on your source WebSphere Portal Express Version 8 profile.

Procedure

1. Run the following command:
   - Linux : `./ConfigEngine.sh remove-paa -DappName=wcm.wcs.integrator.paa`
   - IBM i: `ConfigEngine.sh remove-paa -DappName=wcm.wcs.integrator.paa`
   - Windows: `ConfigEngine.bat remove-paa -DappName=wcm.wcs.integrator.paa`
2. Run the following command:
   - Linux : `./ConfigEngine.sh uninstall-paa -DappName=wcm.wcs.integrator.paa`
   - IBM i: `ConfigEngine.sh uninstall-paa -DappName=wcm.wcs.integrator.paa`
   - Windows: `ConfigEngine.bat uninstall-paa -DappName=wcm.wcs.integrator.paa`
3. Delete the *Portal_V8_profile*/paa/wcm.wcs.integrator.paa directory.

What to do next

When migration is complete, you can install the 8.5 version of the PAA file.

Removing unsupported composite applications

Composite applications are no longer supported. If you have a composite application in your system and you migrate to Version 8.5, the migration fails. Ensure that all composite applications are deleted before you start the migration. When you delete a composite application, you must also run the resource cleaner, otherwise pages can still exist in the database.

Procedure

If you own or manage an application, you can delete it. The applications catalog is the only context where you can delete an application.

1. Use the following code sample to show the applications that you own or manage:

```
AppListString = Application.listall("applications")
if AppListString == '':
  print 'There are no composite applications.'
else:
  AppList = AppListString.split(' ')
  for app in AppList:
    print "- " + Application.nlsget(app,"title","en") + "(" + app + ")"
```

2. For each application that you want to delete, click the **Delete** icon.
3. Click **OK** to confirm the deletion.
4. Run the resource cleaner to ensure that no other pages exist in the database: "Scheduling the delayed cleanup of portal pages" on page 1079.

Related concepts:

"Unsupported and deprecated features for V8.5" on page 17
Review the features that were available in previous versions of IBM WebSphere Portal Express but are no longer available.

Removing obsolete portlets from virtual portal scripts

Some portlets were deprecated or removed in this release. If you have references to deprecated or removed portlets in your virtual portal scripts, you must manually remove those references.

In the virtual portal XML scripts, look for references to deprecated portlets such as,`Manage Seed List` and remove them.

- `<portlet action="locate" name="Manage Seed List" objectid="3_CGAH47L008DE402BK8543I1G63"></portlet>`
- `<p><portletinstance action="update" domain="rel" objectid="5_CGAH47L008DE402BK8543I18D4" portletref="3_CGAH47L008DE402BK8543I1G63" shareref="5_CGAH47L008DE402BK8543I18D4"></portletinstance></p>`

Note: Keep in mind that `objectid`, `portletref`, `shareref`, might vary in your installation.

You might also find references to `installer/wp.config/config/work` which is no longer valid and must be replaced to `installer/wp.config/config/templates`.

There might be other references to resources that do not exist in this new release of WebSphere Portal Express. Make sure that you update all those references as needed.

Distinguished names

If you are using LDAP in your source environment, make sure that the `wkplc.properties` file is properly configured. You might have a configuration that is working, but it might not be supported after migration. Short distinguished names (DN) are not supported. Make sure that the properties files in your source environment are set with the fully qualified distinguished names.

The following excerpt from the `wkplc.properties` file provides some examples of using fully qualified distinguished names (DN) for the **`PortalAdminId`** parameter.

`PortalAdminId`
This value is the user ID for the WebSphere Portal administrator. The installation program sets this value that is based on user input during installation. The user ID cannot contain a space: for example, `user ID`. The user ID cannot be longer than 200 characters.

(UNIX only) Some tasks might require you to enter the fully qualified user ID. If your fully qualified user ID contains a space; for example: `cn=wpsadmin,cn=users,l=SharedLDAP,c=US,ou=Lotus,o=Software Group,dc=ibm,dc=com`, then you must place the fully qualified user ID in the properties file or into a parent properties file instead of as a flag on the command line. To create a parent properties file called `mysecurity.properties`, enter the fully qualified user ID, and then run the following task: **`./ConfigEngine.sh task_name -DparentProperties=/opt/mysecurity.properties`**.

(Windows only) Some tasks might require you to enter the fully qualified user ID. If your fully qualified user ID contains a space; for example: `cn=wpsadmin,cn=users,l=SharedLDAP,c=US,ou=Lotus,o=Software Group,dc=ibm,dc=com`, then you must place quotations around the fully qualified user ID before you run the task, for example: `"cn=wpsadmin,cn=users,l=SharedLDAP,c=US,ou=Lotus,o=Software Group,dc=ibm,dc=com."`

Value: A valid user ID contains only ASCII characters and can contain the following characters:

Lowercase characters {a-z} and uppercase characters {A-Z}

Numbers {0-9}

Exclamation point {!}, Hyphen {-}, period {.}, question mark {?}, accent grave {`}, tilde {~}

Open parenthesis {(} and close parenthesis {)}

Open bracket {[} and close bracket {]}

Underscore {_}, which is the only special character that is allowed in IBM i

Examples: The following are example user IDs:

Development configuration without security: `PortalAdminId=uid=xyzadmin,o=defaultWIMFileBasedRealm`

IBM Tivoli Directory Server: { `uid=,cn=users,dc=yourco,dc=com` }

IBM Lotus Domino: { `cn=,o=yourco.com` }

Novell eDirectory: { `uid=,ou=people,o=yourco.com` }

Oracle Directory Server: { `uid=,ou=people,o=yourco.com` }

Windows Active Directory: { `cn=,cn=users,dc=yourco,dc=com` }

Windows Active Directory-Lightweight-Directory-Services: { `cn=,cn=users,dc=yourco,dc=com` }

Default: no default

Maximum open file descriptors for Unix-based platforms

For Unix-based platforms, the default open file descriptor must be set to 200000 to allow the configuration wizard commands to run properly during migration.

The default open file descriptor limit for the Unix user profile that is used to install and run the WebSphere Portal Express and configuration wizard servers should be set to 200000. Update this value in the user profile for any migrations that use the configuration wizard. The typical syntax is `ulimit -n 200000`. However, you can refer to the documentation specific to your platform to confirm.

If you run the migration commands manually, set the open file descriptor to 200000 prior to running the **`WASPreUpgrade.sh`** and **`WASPostUpgrade.sh`** commands.

Disabling wsadmin client debug

If you use either IBM WebSphere Application Server Version 8.5.5.4 or WebSphere Application Server Version 8.5.5.5, disable the wsadmin client trace to avoid a failure.

There is a known issue in wsadmin client connections to node agents on WebSphere Application Server Version 8.5.5.4 and WebSphere Application Server Version 8.5.5.5 that can cause wsadmin to fail if tracing is enabled.

Disable the wsadmin client trace:

- In *wp_profile_root*`/properties/wsadmin.properties`, comment out `com.ibm.ws.scripting.traceString`. For example:

  ```
  #com.ibm.ws.scripting.traceString=com.ibm.*=all=enabled
  ```

If the wsadmin client trace is needed, you can append the following line to the trace string to disable tracing for the MetaInfoCache class.

- Append `:com.ibm.ws.scripting.MetaInfoCache=off`. For example,

  ```
  com.ibm.ws.scripting.traceString=com.ibm.*=all=enabled:com.ibm.ws.scripting.MetaInfoCa
  ```

Setting up the target environment

The migration requires you to install the required Portal and WebSphere binary installations. To prepare your target environment, ensure that you have applied the latest cumulative fix and the most recent fix pack before you start migration. In addition, create new copies of the source databases for the target environment to use, and prepare the target environment for any custom applications that have dependencies or any other tasks that need to be performed for remote or cluster migrations.

About this task

Migration to a new release requires setting up a target environment. The target environment might be one of the following environments:

- A separate system
- A remote migration
- The same system if you choose a local migration path

Independently from the migration path that is chosen, the target environment requires a fresh, binary only, Portal installation. Meaning setting up the target environment without creating any profiles.

For more details on setting up the target environment, review the following topics in this section and the Roadmaps for migration.

"Target environment considerations" on page 832
Installation planning of the target environment must be part of your overall migration planning. There are some considerations to keep in mind when you install WebSphere Portal Express and you plan to do a migration from your existing installation.

"Installing Portal and WebSphere binary files" on page 833
On your target system, you must install the portal and WebSphere binary files.

"Installing fix packs on the target environment" on page 834
Periodically, fix packs are released to integrate product code fixes. Between fix pack releases, there are interim fixes to ensure product reliability and stability.

You must apply the latest available fix pack and cumulative fix to the target environment. The Recommended fixes link provides links to fix pack and interim fix downloads. There is also information about what is recommended and what is required.

"Copying portal binary files to the deployment manager" on page 834
Complete this task if the deployment manager is not sharing application binary files with a Portal install.

"Copying files for third party and custom applications" on page 835
The Portal migration attempts to install custom applications in the target environment, but it does not automatically copy the files required for those applications. If the files are not copied over, it is possible that the applications will fail to install or not work properly.

"Target environment: Maximum open file descriptors for Unix-based platforms" on page 835
For Unix-based platforms, the default open file descriptor must be set to 200000 to allow the configuration wizard commands to run properly during migration.

"Using copies of source database domains to minimize downtime" on page 835
To keep the earlier portal environment in production and reduce the amount of downtime during migration copy the earlier portal server JCR and Release domains. Connect to the domain copies and then update the new portal server with the domain copies. The process of connecting to the domain copies must be done after you upgrade the ConfigEngine tool but before you upgrade the Portal profile.

"Database considerations" on page 836
Depending on the type of database that you use, there might be extra considerations or tasks to complete before and after you migrate your data. Review the information that is tailored for your database type to ensure the migration process completes successfully.

Related information:

Recommended fixes and updates for WebSphere Portal and Web Content Management

WebSphere Portal detailed system requirements

Target environment considerations

Installation planning of the target environment must be part of your overall migration planning. There are some considerations to keep in mind when you install WebSphere Portal Express and you plan to do a migration from your existing installation.

Installing a brand new WebSphere Portal Express

When you install a new environment for migration, you must choose the binary-only option. This installation type does not create any profile that might conflict with the migration process. Refer to the Related information section for specific details on installing the target environment.

Installing the latest fix pack and cumulative fix

Ensure that you install the latest available fix pack and cumulative fix. They might include fixes and improvements to the migration process. Refer to Recommended fixes and updates for WebSphere Portal and Web Content Management for details.

Reserved user names and special characters

There are other considerations to keep in mind around the user IDs and passwords when compared to a regular installation. The `xyzadmin` is a reserved value and cannot be used as WebSphere Portal Express or WebSphere Application Server administrator name. You must change the administrator name from your source environment if you are using this name.

Related tasks:

"Replacing the WebSphere Portal Express administrator user ID" on page 1675
If you change your security configuration, you might need to replace your old IBM WebSphere Portal Express administrator user ID with a new WebSphere Portal Express administrator user ID.

"Replacing the WebSphere Application Server administrator user ID" on page 1674
If you change your security configuration, you might need to replace your old IBM WebSphere Application Server administrator user ID with a new WebSphere Application Server administrator user ID.

"Planning to install WebSphere Portal Express" on page 101
Before you install IBM WebSphere Portal Express in a production environment, you need to assess your hardware and software needs, possible database configurations, security options, and LDAP server options. Skipping this important step can lead to unexpected results and costly delays.

Related information:

"User IDs and passwords" on page 107
Understanding character limitations for user IDs and passwords is important because they are used throughout the system to provide access and secure content. The character limitations provided here apply to the IBM WebSphere Portal Express administrator, IBM WebSphere Application Server administrator, database administrator, LDAP server administrator, and user IDs. Database and LDAP servers can have more restrictive limitations than provided here. Therefore, check the database and LDAP server product documentation for restrictions. Failure to correctly define user IDs and passwords during the installation process can result in installation failure. In addition, your company might have more restrictive user ID and password requirements that you must also follow.

 Recommended fixes and updates for WebSphere Portal and Web Content Management

Installing Portal and WebSphere binary files

On your target system, you must install the portal and WebSphere binary files.

To effectively set up your target environment, install the Portal and WebSphere binary files on all target systems. Nodes and stand-alone servers require a binary Portal and WebSphere installation, while deployment managers require only a WebSphere binary installation.

IBM WebSphere Portal Express version 8.5 is packaged with IBM WebSphere Application Server version 8.5.5.2. There are some fixes in WebSphere Application Server version 8.5.5.4 and later that might prevent some problems from occurring during the migration. Therefore, when you install WebSphere Portal Express and the binary files, install the latest version of the WebSphere Application Server. Complete the following steps to install WebSphere Portal Express with the latest version of WebSphere Application Server:

1. Download the latest version of WebSphere Application Server from Passport Advantage.

2. Start the Installation Manager.
3. Go to **File** > **Preferences** > **Repositories**.
4. Add the repository for the latest version of WebSphere Application Server.
5. When you install WebSphere Portal Express, select the repositories for the latest version of WebSphere Application Server and the repository for WebSphere Portal Express.

The following high-level steps provide guidance for using the IBM Installation Manager. Refer to the Installing the Exceptional Digital Experience section for other installation alternatives.

Note: If you previously did a full Portal installation, you must remove all profiles.

1. Start the IBM Installation Manager.
2. Follow the IBM Installation Manager instructions through the different screens.
3. While on the **Features** screen, make sure that **Create a new Portal Server Profile** is not selected.
4. Proceed with the remaining instructions to complete the binary installation.

Related tasks:

"Installing the digital experience software" on page 167
IBM's Exceptional Digital Experience is designed to help create, manage, simplify, and integrate your processes into an engaging online experience. IBM WebSphere Portal and IBM Web Content Manager are a part of the Exceptional Digital Experience. The product documentation uses digital experience software as a shorthand for IBM WebSphere Portal and IBM Web Content Manager.

Installing fix packs on the target environment

Periodically, fix packs are released to integrate product code fixes. Between fix pack releases, there are interim fixes to ensure product reliability and stability. You must apply the latest available fix pack and cumulative fix to the target environment. The Recommended fixes link provides links to fix pack and interim fix downloads. There is also information about what is recommended and what is required.

Related information:

Recommended fixes and updates for WebSphere Portal and Web Content Management

Copying portal binary files to the deployment manager

Complete this task if the deployment manager is not sharing application binary files with a Portal install.

Procedure

Copy the required files (including the migration plug-in) from the IBM WebSphere Portal Express Version 8.5 binary installation to the target deployment manager system:

1. Copy the `filesForDmgr.zip` file to the remote deployment manager system.
2. Expand the `filesForDmgr.zip` file into the installation root directory of the deployment manager. For example, in the `C:\IBM\WebSphere\AppServer directory`.

If the deployment manager profile was not created in the default `AppServer/profiles/Dmgr01` directory, then the `metadata_wkplc.xml` file in the

AppServer/profiles/Dmgr01/config/.repository/metadata_wkplc.xml directory in the compressed file, must be copied into the config/.repository subdirectory under the deployment manager profile directory.
Failure to copy the migration plug-in files to the deployment manager can result in the following error when you try to upgrade
the ConfigEngine tool:com.ibm.websphere.management.exception.InvalidConfigDataTypeExcept
ADMG0007E: The configuration data type CellCompRegistryCollection is not valid.

Copying files for third party and custom applications

The Portal migration attempts to install custom applications in the target environment, but it does not automatically copy the files required for those applications. If the files are not copied over, it is possible that the applications will fail to install or not work properly.

Target environment: Maximum open file descriptors for Unix-based platforms

For Unix-based platforms, the default open file descriptor must be set to 200000 to allow the configuration wizard commands to run properly during migration.

The default open file descriptor limit for the Unix user profile that is used to install and run the WebSphere Portal Express and configuration wizard servers should be set to 200000. Update this value in the user profile for any migrations that use the configuration wizard. The typical syntax is `ulimit -n 200000`. However, you can refer to the documentation specific to your platform to confirm.

If you run the migration commands manually, set the open file descriptor to 200000 prior to running the **WASPreUpgrade.sh** and **WASPostUpgrade.sh** commands.

Using copies of source database domains to minimize downtime

To keep the earlier portal environment in production and reduce the amount of downtime during migration copy the earlier portal server JCR and Release domains. Connect to the domain copies and then update the new portal server with the domain copies. The process of connecting to the domain copies must be done after you upgrade the ConfigEngine tool but before you upgrade the Portal profile.

About this task

Review the following list before you begin.

Important:

- Copying the source portal JCR and Release domains is a recommendation but not a requirement. If you point the new portal server to the source portal domains, you cannot use the JCR and Release domains with the earlier portal server.
- During migration, the Community and Customization domains are upgraded to their new definitions required by the new WebSphere Portal Express release. This migration might break the compatibility with the source portal server. Carefully check the requirements for the source portal server if you plan to use the earlier portal server until migration is finished.

- If you are migrating from WebSphere Portal Express Version 7 to WebSphere Portal Express Version 8.5, be aware that there is a major schema change in the JCR database that may cause the JCR database to triple in size. Ensure that there is enough disk space for the new copies of the database when you create the new copies.
- DB2 only: When you migrate a source portal to a target portal that uses a different driver type, reconnect all of the affected domains. For example, the source portal is using DB2 with Type 2 drivers for all domains. You plan to use a Type 4 driver type for all of the domains in your target portal. In this case, you must reconnect all of the database domains that use the **connect-database** command.

Note: The **connect-database** configuration task does not preserve customizations to the data sources for the WebSphere Portal Express databases. If you previously tuned your data sources for the WebSphere Portal Express databases, make a note of the settings, run **connect-database**, and reapply the tuning after you run the configuration task.

Procedure

1. Use your Database tools to copy the source portal JCR domain and Release domain.

   Note: If you are using IBM DB2 Universal Database for z/OS, review the following considerations:
   - If you plan to use the DB2 Administration Tool to copy the database domains, make sure that APAR PM16847 is applied.
   - Make sure that you verify the databases are not in a `COPY PENDING` state before you connect to the database copies described in the following step.
2. DB2 only: On the database copies, verify that the Statement Heap size is set to at least 32k.
   a. List the database manager configuration parameters by running the following command **db2 get db cfg for** *dbname*.
   b. If the Statement Heap size is smaller than 32k, increase it by running the following command **db2 "UPDATE DB CFG FOR** *dbname* **USING stmtheap 32768"**

   Where *dbname* is the name of your WebSphere Portal Express database.
3. On the target portal, update the `wkplc_dbdomain.properties` file in the *wp_profile_root*`/ConfigEngine/properties` directory.
   a. Update the `jcr.*` and `release.*` database properties to point to the JCR and Release domain copies that you created in the previous step.

Database considerations

Depending on the type of database that you use, there might be extra considerations or tasks to complete before and after you migrate your data. Review the information that is tailored for your database type to ensure the migration process completes successfully.

"Configuring transaction logging space" on page 837
When you migrate from earlier versions of WebSphere Portal Express, certain portions of the actual database migration use database transactions that can contain large volumes of data change. In order for the database to accommodate these transaction changes, the amount of space available to the database transaction logs might need to be increased.

"Oracle: Disabling the auto space advisor background task" on page 839
To prevent deadlocks during migration, you must complete a task to disable the Oracle background task called "Auto Space Advisor" before your run the **upgrade-profile** task during migration using the Configuration Wizard. After you complete migration, you can enable "Auto Space Advisor" as a post-migration task.

"DB2" on page 840
Database migration is one of the most time consuming portions of the migration process. If you are migrating from DB2, review the following topics to improve the speed of the database migration process.

Configuring transaction logging space

When you migrate from earlier versions of WebSphere Portal Express, certain portions of the actual database migration use database transactions that can contain large volumes of data change. In order for the database to accommodate these transaction changes, the amount of space available to the database transaction logs might need to be increased.

About this task

There are certain database platforms that allow for the temporary usage of unlimited database transaction log space. It is advisable to enable that option during the migration to avoid any possible issues that are related to exhausting available transaction log space.

On platforms or environments where unlimited transaction logging is not an option, the following formula can be used to determine the amount of transaction log space that is required to complete the migration. Note that several of these queries can be long running (several minutes) depending on the amount of data in the Portal database.

Before you run the SQL commands, you need to replace the following references according to your environment.

- <*schema*> replace this token with the schema used in the JCR Domain database.
- <*ROOT_WSID*> replace this token with the result of `SELECT WSID FROM <schema>.ICMSTJCRWS WHERE WSNAME = 'ROOTWORKSPACE'`
- <*VER_WSID*> replace this token with the result of `SELECT WSID FROM <schema>.ICMSTJCRWS WHERE WSNAME = 'jcr:versioning'`

As you run the following steps, save the result as you need to do some computation to calculate the correct numbers for setting the transaction logging space.

Steps #1 - #8 are for calculating the space that is required to migrate hierarchy information for WebSphere Portal Express data that is stored in the JCR excluding all version information.

Steps #9 - #16 are for calculating the space that is required to migrate hierarchy information for version information of WebSphere Portal Express data.

Procedure

1. `SELECT COUNT(LID) FROM <schema>.ICMSTJCRLINKS WHERE WSID = <ROOT_WSID> AND INLEAFTREEFLAG = 1 AND SIID IN ( SELECT TIID FROM <schema>.ICMSTJCRLINKS WHERE INLEAFTREEFLAG = 0 AND WSID = <ROOT_WSID> )`

2. SELECT COUNT(LID) FROM <*schema*>.ICMSTJCRLINKS WHERE WSID = <*ROOT_WSID*> AND INLEAFTREEFLAG = 1 AND SIID IN (SELECT TIID FROM <schema>.ICMSTJCRLINKS WHERE WSID = <ROOT_WSID> AND INLEAFTREEFLAG = 1 AND SIID IN (SELECT TIID FROM <schema>.ICMSTJCRLINKS WHERE INLEAFTREEFLAG = 0 AND WSID = <*ROOT_WSID*>))
3. SELECT COUNT(LID) FROM <*schema*>.ICMSTJCRLINKS WHERE WSID = <*ROOT_WSID*> AND INLEAFTREEFLAG = 1
4. Run the following commands depending on your database
   - DB2 and SQL Server
     - WITH PATH (WSID , SIID, TIID, LVL) AS ((SELECT WSID, SIID, TIID, 1 AS LVL FROM <*schema*>.ICMSTJCRLINKS WHERE INLEAFTREEFLAG=1 AND WSID = <ROOT_WSID> AND SIID IN (SELECT TIID FROM <schema>.ICMSTJCRLINKS WHERE INLEAFTREEFLAG = 0 AND WSID = <*ROOT_WSID*>)) UNION ALL (SELECT L.WSID, L.SIID, L.TIID, P.LVL + 1 FROM PATH P, <*schema*>.ICMSTJCRLINKS L WHERE P.SIID = L.TIID AND P.LVL < 1000000 AND INLEAFTREEFLAG = 0 AND L.WSID = <*ROOT_WSID*>)) SELECT SUM(P1.LVL) FROM PATH P1, <*schema*>.ICMSTJCRWS WS WHERE P1.WSID = <*ROOT_WSID*> AND WS.WSID = <*ROOT_WSID*> AND P1.SIID = WS.ROOTIID
   - Oracle
     - SELECT SUM(LEV) FROM (SELECT WSID, SIID, TIID, LEVEL LEV FROM JCRPRODWCM.ICMSTJCRLINKS LINKS WHERE WSID = <*ROOT_WSID*> START WITH INLEAFTREEFLAG = 1 AND SIID IN (SELECT TIID FROM <*schema*>.ICMSTJCRLINKS WHERE INLEAFTREEFLAG = 0 AND WSID = <*ROOT_WSID*>) CONNECT BY PRIOR SIID = TIID AND INLEAFTREEFLAG = 0) P1, JCRPRODWCM.ICMSTJCRWS WS WHERE P1.WSID = <*ROOT_WSID*> AND WS.WSID = <*ROOT_WSID*> AND P1.SIID = WS.ROOTIID
5. Divide the value that is obtained in step #4 by the value of step #1 round up to nearest whole value and save that number.
6. Subtract the values that are obtained from steps #3, #2 and #1 and save that number.
7. Perform the following equation with the values previously calculated: ((value of #5 + 1) * value of #2) + ((value of #5 + 2) * value of #6) + value of #4
8. Multiply the value that is obtained in the previous step #7 by 700
9. SELECT COUNT(LID) FROM <*schema*>.ICMSTJCRLINKS WHERE WSID = <*VER_WSID*> AND INLEAFTREEFLAG = 1 AND SIID IN (SELECT TIID FROM <*schema*>.ICMSTJCRLINKS WHERE INLEAFTREEFLAG = 0 AND WSID = <*VER_WSID*>)
10. SELECT COUNT(LID) FROM <*schema*>.ICMSTJCRLINKS WHERE WSID = <*VER_WSID*> AND INLEAFTREEFLAG = 1 AND SIID IN (SELECT TIID FROM <*schema*>.ICMSTJCRLINKS WHERE WSID = <*VER_WSID*> AND INLEAFTREEFLAG = 1 AND SIID IN (SELECT TIID FROM <*schema*>.ICMSTJCRLINKS WHERE INLEAFTREEFLAG = 0 AND WSID = <*VER_WSID*>))
11. SELECT COUNT(LID) FROM <*schema*>.ICMSTJCRLINKS WHERE WSID = <*VER_WSID*> AND INLEAFTREEFLAG = 1
12. Run the following commands depending on your database
    - DB2 and SQL Server
      - WITH PATH (WSID , SIID, TIID, LVL) AS ((SELECT WSID, SIID, TIID, 1 AS LVL FROM <*schema*>.ICMSTJCRLINKS WHERE INLEAFTREEFLAG=1 AND WSID = <*VER_WSID*> AND SIID IN (SELECT TIID FROM <*schema*>.ICMSTJCRLINKS WHERE INLEAFTREEFLAG = 0 AND WSID = <*VER_WSID*>)) UNION ALL (SELECT L.WSID, L.SIID, L.TIID, P.LVL + 1 FROM PATH P, <*schema*>.ICMSTJCRLINKS L WHERE P.SIID = L.TIID AND

```
P.LVL < 1000000 AND INLEAFTREEFLAG = 0 AND L.WSID = <VER_WSID>) )
SELECT SUM(P1.LVL) FROM PATH P1, <schema>.ICMSTJCRWS WS WHERE
P1.WSID = <VER_WSID> AND WS.WSID = <VER_WSID> AND P1.SIID =
WS.ROOTIID
```

- Oracle
  - `SELECT SUM(LEV) FROM (SELECT WSID, SIID, TIID, LEVEL LEV FROM JCRPRODWCM.ICMSTJCRLINKS LINKS WHERE WSID = <VER_WSID> START WITH INLEAFTREEFLAG = 1 AND SIID IN (SELECT TIID FROM <schema>.ICMSTJCRLINKS WHERE INLEAFTREEFLAG = 0 AND WSID = <VER_WSID>) CONNECT BY PRIOR SIID = TIID AND INLEAFTREEFLAG = 0) P1, JCRPRODWCM.ICMSTJCRWS WS WHERE P1.WSID = <VER_WSID> AND WS.WSID = <VER_WSID> AND P1.SIID = WS.ROOTIID`

13. Divide the value that is obtained in step #12 by the value of #9 round up to nearest whole value and save that number.
14. Subtract the values that are obtained from steps #11, #10 and #9 and save that number.
15. Perform the following equation with the values previously calculated: ((value of #13 + 1) * value of #10) + ((value of #13 + 2) * value of #14) + value of #12
16. Multiply the value that is obtained in the previous step #15 by 700

Results

The estimated amount of database transaction log file space necessary in bytes is the greater of the two values that are determined in steps #8 and #16.

Oracle: Disabling the auto space advisor background task

To prevent deadlocks during migration, you must complete a task to disable the Oracle background task called "Auto Space Advisor" before your run the **upgrade-profile** task during migration using the Configuration Wizard. After you complete migration, you can enable "Auto Space Advisor" as a post-migration task.

Before you begin

You can check if the "Auto Space Advisor" task is enabled or disabled by entering the following command from SQL Plus:

```
SQL> select status from dba_autotask_client where client_name = 'auto space advisor';
```

Procedure

Enter the following SQL:

```
BEGIN
 dbms_auto_task_admin.disable(
  client_name => 'auto space advisor',
  operation  => NULL,
  window_name => NULL);
END;
/
```

Note: If you do not turn off the "Auto Space Advisor," it is possible for the migration to fail with the following Oracle error: `ORA-00060: deadlock detected while waiting for resource`.

What to do next

When you complete migrating your data using the Configuration Wizard, complete the post-migration step to enable the "Auto Space Advisor" Oracle background task.

Related tasks:

"Oracle: Enabling the auto space advisor background task" on page 896
If you use an Oracle database, you must complete a post-migration task to enable the Oracle background task called "Auto Space Advisor" after your run the **upgrade-profile** task during migration using the Configuration Wizard.

DB2

Database migration is one of the most time consuming portions of the migration process. If you are migrating from DB2, review the following topics to improve the speed of the database migration process.

"Update database statistics"
Statistics are information that is collected about the contents of the database tables and indexes. They are used by the cost-based optimizer in the database to influence the approach (query plan) that is used to run SQL queries. By collecting updated or new statistics, you provide the database optimizer the most up-to-date information about the tables and indexes. Therefore, the query execution plans are generated as efficiently as possible.

"Preparing DB2 for large data sets migration"
If you use DB2 and have a large amount of content in the JCR repository (for example, Web Content Manager data), prepare the DB2 for migration. You must update several settings that are related to performance.

Update database statistics:

Statistics are information that is collected about the contents of the database tables and indexes. They are used by the cost-based optimizer in the database to influence the approach (query plan) that is used to run SQL queries. By collecting updated or new statistics, you provide the database optimizer the most up-to-date information about the tables and indexes. Therefore, the query execution plans are generated as efficiently as possible.

This information is important during migration because there are several tables and indexes that are created and populated during this process. The default statistical information can lead query plans that show degrading performance as the migration progresses and can carry into the runtime performance.

JCR domain

Gather current statistics on all tables that are found in the JCR domain. If you are using DB2, you must run **reorgchk** and **runstats**.

Run the statistics on all the columns in all tables and indexes. Gather at least a minimum level of sampling and distribution. Refer to your database documentation for details on updating statistics.

Preparing DB2 for large data sets migration:

If you use DB2 and have a large amount of content in the JCR repository (for example, Web Content Manager data), prepare the DB2 for migration. You must update several settings that are related to performance.

About this task

These values are suggested starting points and were used successfully in IBM testing. Exact values can vary, depending on the size of the data set. Consult your database administrator as needed.

Important: The settings described here are optimized for migration rather than day-to-day operation. After migration completes successfully, you can revert any changes back to the values in use before migration.

Procedure

1. Update the lock list size for the database. A value of at least `10000` is recommended.
2. Disable all automatic maintenance on the database, including health monitoring.
3. Increase the package cache size. A value of at least `24000` is recommended.
4. Increase the statistics heap size. A value of at least `16384` is recommended.
5. Perform a complete reorganization of the tables and indexes on the database that is used for migration.
6. Collect detailed statistics with distribution for all tables and indexes.

   Note: While the migration is in progress, it might be necessary to collect detailed statistics to continue to allow the database optimizer to select efficient access plans.

Migrate data using the configuration wizard

For Version 8.5, data, applications, databases, property files, security settings, and configuration are migrated using the Configuration Wizard. Use the roadmaps for cluster and stand-alone environments to guide you through the process.

See the *Roadmaps for migration* section of the product documentation, and choose the roadmap that is specified for your environment for details on using the Configuration Wizard.

"Migrate a stand-alone server" on page 842
Use the Configuration Wizard to migrate a stand-alone server environment. Use the following information to get familiar with the information that you must provide in the wizard and the configuration procedure that it generates.

"Cluster: Migrate the deployment manager profile" on page 847
Use the Configuration Wizard to migrate the deployment manager profile for a cluster environment. Use the following information to get familiar with the information that you must provide in the wizard and the configuration procedure that it generates.

"Cluster: Migrate node profiles" on page 850
Use the Configuration Wizard to upgrade the node profiles for a cluster environment. Use the following information to get familiar with the information that you must provide in the wizard and the configuration procedure that it generates.

"Cluster: Upgrade node profiles" on page 853
Use the Configuration Wizard to upgrade the nodes profiles for a cluster environment. Use the following information to get familiar with the information you must provide in the wizard and the configuration procedure that it generates.

Related concepts:

"Configuration Wizard" on page 232
Use the Configuration Wizard to set up stand-alone servers and new deployments, create clusters, migrate and update to new versions, and add new capabilities to existing deployments.

Related tasks:

"Roadmaps for migration" on page 92
Choose the appropriate migration roadmap for your environment.

Migrate a stand-alone server

Use the Configuration Wizard to migrate a stand-alone server environment. Use the following information to get familiar with the information that you must provide in the wizard and the configuration procedure that it generates.

Configuration Wizard

Select **Migrate to a New Version**, and choose the **Migrate a Stand-alone Server** option.

Related concepts:

"Troubleshooting: Migrate a stand-alone server" on page 3599
If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

"Configuration Wizard" on page 232
Use the Configuration Wizard to set up stand-alone servers and new deployments, create clusters, migrate and update to new versions, and add new capabilities to existing deployments.

"Roadmap: Migrating a stand-alone server environment" on page 92
Roadmaps provide a high-level overview of complex tasks such as migrating a stand-alone server environment to a new version of IBM WebSphere Portal Express.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

Worksheet

To set up the migration, you answer questions about your wanted configuration. Some fields apply to migration configurations. Some fields are required based on your environment. The remaining fields are advanced and do not apply to most configurations.

Minimal required fields

The following table lists the fields that are unique to the migrate a stand-alone server configuration. You might be prompted for additional information about system or user IDs and passwords that you defined during the portal installation process.

Table 101. Minimal required fields

Field Label	Default	Your Value
Target operating system	Linux	
Target portal profile name	wp_profile	
Target portal profile home directory	/opt/IBM/WebSphere/wp_profile	
Is the target portal on the same server or a different server	Same server	
Database management software	Derby	
Target operating system		
WebSphere Application Server administrator	wpsadmin	
WebSphere Application Server administrator password		
Portal administrator password		
What is the portal profile name	wp_profile	
What is the cell name	CellName	
What is the portal node name	NodeName	
Where is the source application server installed **IBMi only:** Provide the path to the source profile directory instead of the application server directory.	/opt/IBM/WebSphere/AppServer	
What is the new host name		
Target portal soap port **Note:** Enter the same port number used for the source environment.	10033	
Where is the target application server installed	/opt/IBM/WebSphere/AppServer	
Where is the target portal installed	/opt/IBM/WebSphere/PortalServer	
Target temporary path	/opt/IBM/WebSphere/PortalServer	

Advanced fields

The following table lists the advanced fields that are unique to the migrate a stand-alone server configuration. Click **Advanced** on the Answer Questions page for the target system to see the advanced properties. Default values are provided for advanced fields that are required.

Table 102. Advanced fields

Field Label	Default	Your Value
JVM heap size	2048	

Migrate a stand-alone server option

After you answer questions and provide information about your migration, the wizard generates a custom configuration procedure.

About this task

Depending on your environment, the wizard generates a configuration process. The following steps reflect all possible steps in the configuration process. The steps do not represent a literal configuration. The steps are provided as a reference.

If you click **View Step Command**, you can see the task and properties that are associated with each step in the wizard.

Procedure

1. Manual Step: Install the latest fix packs

   Condition
   none

   ConfigEngine task
   none
2. Generate the files for remote migration

   Condition
   The target portal is on a different server than the source.

   ConfigEngine task
   none
3. Manual Step: Copy the remote migration package to the source environment

   Condition
   The target portal is on a different server than the source.

   ConfigEngine task
   none
4. Create a backup of the remote source portal profile

   Condition
   The target portal is on a different server than the source.

   ConfigEngine task
   none
5. Create a backup profile of the source portal profile

   Condition
   none

   ConfigEngine task
   none
6. Manual Step: If the backup profile is larger than 2 GB, clean up the backup profile

   Condition
   none

   ConfigEngine task
   none
7. Create a default profile

   Condition
   none

   ConfigEngine task
   none
8. Import backup profile

   Condition
   none

   ConfigEngine task
   none
9. Manual Step: If you cleaned up the backup profile, restore the JCR content

Condition
none

ConfigEngine task
none

10. Upgrade the ConfigEngine

    Condition
    none

    ConfigEngine task
    none

11. Manual Step: Update the ports on the target environment

    Condition
    none

    ConfigEngine task
    none

12. Manual Step: Update database settings

    Condition
    none

    ConfigEngine task
    none

13. Validate database settings

    Condition
    none

    ConfigEngine task
    `validate-database`

14. Connect to new database copies

    Condition
    none

    ConfigEngine task
    `connect-database`

15. Manual Step: Review database schema changes

    Condition
    none

    ConfigEngine task
    none

16. Upgrade the base portal database component

    Condition
    IBM z/OS

    DB2

    ConfigEngine task
    `grant-runtime-db-user-privileges`

    `upgrade-database`

17. Manual Step: Remove check pending statuses from table spaces

    Condition
    IBM z/OS

DB2

ConfigEngine task
none

18. Upgrade the remaining portal databases

    Condition
    IBM z/OS

    DB2

    ConfigEngine task
    grant-runtime-db-user-privileges

    upgrade-database

19. Upgrade the portal profile

    Condition
    none

    ConfigEngine task
    upgrade-profile

    Version 7.0 server-only migration: Do not complete the upgrade profile step using the Configuration Wizard. Instead you must run this task manually. For more information about running this task manually, see "Migration from Portal 7.0 server-only to Portal 8.5" on page 802.

    Note: When you run this step, the sub task that is named **action-deploy-portlets-applyMIGStatic-wp.oob.full** runs and completes successfully. However, the following error messages are shown. You can ignore these error messages:

    - `EJPXA0161W: The web module ContactList could not be activated. Please see previous messages for reasons and possible corrective actions.`
    - `EJPPH0048W: The synchronization mode of all nodes in the portal cluster is not consistently set. The portlet application PA_ContactList will not be started in the Application Server. Manual synchronization is assumed for all nodes. Manually start the application after all nodes were synchronized.`
    - `EJPXA0067E: The following configuration data is needed to create a content-node resource: content-parentref.`

What to do next

To complete migration, you must perform several post-migration tasks that depend on how you use WebSphere Portal Express.

1. Review the Next steps section of the product documentation.
   - Run the **PRE-APPLY-FIX** and **APPLY-FIX** tasks from Applying the latest combined cumulative fix updates.
   - Complete the post-migration activities that apply to how you are using WebSphere Portal Express before you move on to the next step. For example, if you are using a virtual portal, then complete the virtual portal post-migration steps.
   - Start the enabling new functionality tasks only after you complete the post-migration tasks.

Cluster: Migrate the deployment manager profile

Use the Configuration Wizard to migrate the deployment manager profile for a cluster environment. Use the following information to get familiar with the information that you must provide in the wizard and the configuration procedure that it generates.

Configuration Wizard

Select **Migrate to a New Version**, and choose the **Migrate a Cluster Step 1: Migrate the Deployment Manager Profile** option.

Related concepts:

"Troubleshooting: Migrate the deployment manager profile for a cluster environment" on page 3605
If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

"Configuration Wizard" on page 232
Use the Configuration Wizard to set up stand-alone servers and new deployments, create clusters, migrate and update to new versions, and add new capabilities to existing deployments.

"Roadmap: Migrating a clustered environment" on page 94
Roadmaps provide a high-level overview of complex tasks such as migrating a clustered environment to a new version of IBM WebSphere Portal Express.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

Worksheet

To set up the migration, you answer questions about your wanted configuration. Some fields apply to migration configurations. Some fields are required based on your environment. The remaining fields are advanced and do not apply to most configurations.

Minimal required fields

The following table lists the fields that are unique to the Migrate a cluster step 1: Migrate the deployment manager profile configuration option. You might be prompted for additional information about system or user IDs and passwords that you defined during the portal installation process.

Table 103. Minimal required fields

Field Label	Default	Your Value
Target operating system	Linux	
Target portal profile name	wp_profile	
Target portal profile home directory	/opt/IBM/WebSphere/wp_profile	
Is the target portal on the same server or a different server	Same server	
WebSphere Application Server administrator	wpsadmin	

Table 103. Minimal required fields (continued)

Field Label	Default	Your Value
WebSphere Application Server administrator password		
What is the deployment manager profile name	dmgr01	
What is the cell name	CellName	
What is the deployment manager node name	dmgr	
Where is the source application server installed **IBMi only:** Provide the path to the source profile directory instead of the application server directory.	/opt/IBM/WebSphere/AppServer	
What is new host name		
Target deployment manager profile name	dmgr01	
Where is the target application server installed	/opt/IBM/WebSphere/AppServer	
Target deployment manager profile path	/opt/IBM/WebSphere/AppServer/profiles/dmgr01	
Target temporary path	/tmp	

Advanced fields

The following table lists the advanced fields that are unique to the Migrate a cluster step 1: Migrate the deployment manager profile configuration option. Click **Advanced** on the Answer Questions page for the target deployment manager system to see the advanced properties. Default values are provided for advanced fields that are required.

Table 104. Advanced fields

Field Label	Default	Your Value
JVM heap size	2048	

Migrate a cluster step 1: Migrate the deployment manager profile option

After you answer questions and provide information about your migration, the wizard generates a custom configuration procedure.

About this task

Depending on your environment, the wizard generates a configuration process. The following steps reflect all possible steps in the configuration process. The steps do not represent a literal configuration. The steps are provided as a reference.

If you click **View Step Command**, you can see the task and properties that are associated with each step in the wizard.

Procedure

1. Manual Step: Disable automatic synchronization on all nodes in the cluster

   Condition

   none

ConfigEngine task
none

2. Manual Step: Stop the deployment manager

   Condition
   none

   ConfigEngine task
   none

3. Manual Step: Install the latest fix packs

   Condition
   none

   ConfigEngine task
   none

4. Manual Step: Install the Portal and WebSphere binary files

   Condition
   none

   ConfigEngine task
   none

5. Manual Step: Copy required portal binary files to the target deployment manager

   Condition
   none

   ConfigEngine task
   none

6. Manual Step: Generate files for remote migration on the deployment manager

   Condition
   The target portal is on a different server than the source.

   ConfigEngine task
   none

7. Manual Step: Copy the remote migration package to the source environment

   Condition
   The target portal is on a different server than the source.

   ConfigEngine task
   none

8. Manual Step: Create a backup of the source deployment manager

   IBMi only: You must remove the **`oldProfile`** parameter before running the command.

   Condition
   none

   ConfigEngine task
   none

9. Manual Step: Create a default deployment manager profile

   Condition
   none

   ConfigEngine task
   none

10. Manual Step: Import the backup profile

    Condition
    none

    ConfigEngine task
    none

Cluster: Migrate node profiles

Use the Configuration Wizard to upgrade the node profiles for a cluster environment. Use the following information to get familiar with the information that you must provide in the wizard and the configuration procedure that it generates.

Configuration Wizard

Select **Migrate to a New Version**, and choose the **Migrate a Cluster Step 2: Migrate Node Profiles** option.

Note: This option is not available for IBM z/OS.

Related concepts:

"Troubleshooting: Migrate node profiles for a cluster environment" on page 3608
If you encounter a failure during the migration of the node profiles for a cluster environment, learn how to correct the issue and recover from the failure.

"Configuration Wizard" on page 232
Use the Configuration Wizard to set up stand-alone servers and new deployments, create clusters, migrate and update to new versions, and add new capabilities to existing deployments.

"Roadmap: Migrating a clustered environment" on page 94
Roadmaps provide a high-level overview of complex tasks such as migrating a clustered environment to a new version of IBM WebSphere Portal Express.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

Worksheet

To set up the migration, you answer questions about your wanted configuration. Some fields apply to migration configurations. Some fields are required based on your environment. The remaining fields are advanced and do not apply to most configurations.

Minimal required fields

The following table lists the fields that are unique to the Migrate a cluster step 2: Migrate node profiles configuration. You might be prompted for additional information about system or user IDs and passwords that you defined during the portal installation process.

Table 105. Minimal required fields

Field Label	Default	Your Value
Target operating system	Linux	

Table 105. Minimal required fields (continued)

Field Label	Default	Your Value
Target portal profile name	wp_profile	
Target portal profile home directory	/opt/IBM/WebSphere/wp_profile	
Is the target portal on the same server or a different server	Same server	
WebSphere Application Server administrator	wpsadmin	
WebSphere Application Server administrator password		
What is the portal profile name	wp_profile	
What is the cell name	CellName	
What is the portal node name	NodeName	
What is the deployment manager node name	dmgr	
Where is the source application server installed **IBMi only:** Provide the path to the source profile directory instead of the application server directory.	/opt/IBM/WebSphere/AppServer	
What is the new host name		
Where is the target application server installed	/opt/IBM/WebSphere/AppServer	
Target temporary path	/tmp	

Advanced fields

The following table lists the advanced fields that are unique to the Migrate a cluster step 2: Migrate node profiles configuration option. Click **Advanced** on the Answer Questions page for the target system to see the advanced properties. Default values are provided for advanced fields that are required.

Table 106. Advanced fields

Field Label	Default	Your Value
JVM heap size	2048	

Migrate a cluster step 2: Migrate node profiles option

After you answer questions and provide information about your migration, the wizard generates a custom configuration procedure.

About this task

Depending on your environment, the wizard generates a configuration process. The following steps reflect all possible steps in the configuration process. The steps do not represent a literal configuration. The steps are provided as a reference.

If you click **View Step Command**, you can see the task and properties that are associated with each step in the wizard.

Procedure

1. Manual Step: Stop the source deployment manager and node agents

Condition
none

ConfigEngine task
none

2. Manual Step: Start the target deployment manager

   Condition
   none

   ConfigEngine task
   none

3. Generate the files for remote migration

   Condition
   The target portal is on a different server than the source.

   ConfigEngine task
   none

4. Manual Step: Copy the remote migration package to the source environment

   Condition
   The target portal is on a different server than the source.

   ConfigEngine task
   none

5. Create a backup of the source portal profile

   Condition
   none

   ConfigEngine task
   none

6. Manual Step: Create a backup of the remote source portal profile

   Condition
   The target portal is on a different server than the source.

   ConfigEngine task
   none

7. Manual Step: Update the deployment manager settings

   Condition
   none

   ConfigEngine task
   none

8. Manual Step: If the backup profile is larger than 2 GB, clean up the backup profile

   Condition
   none

   ConfigEngine task
   none

9. Create a default profile

   Condition
   none

   ConfigEngine task
   none

10. Import the backup profile

    Condition
    none

    ConfigEngine task
    none

11. Manual Step: If you cleaned up the backup profile, restore the JCR content

    Condition
    none

    ConfigEngine task
    none

Cluster: Upgrade node profiles

Use the Configuration Wizard to upgrade the nodes profiles for a cluster environment. Use the following information to get familiar with the information you must provide in the wizard and the configuration procedure that it generates.

Configuration Wizard

Select **Migrate to a New Version**, and choose the **Migrate a Cluster Step 3: Upgrade Node Profiles** option.

Related concepts:

"Troubleshooting: Upgrade node profiles for a cluster environment" on page 3612
If you encounter a failure while upgrading the node profiles for a cluster environment, learn how to correct the issue and recover from the failure.

"Configuration Wizard" on page 232
Use the Configuration Wizard to set up stand-alone servers and new deployments, create clusters, migrate and update to new versions, and add new capabilities to existing deployments.

"Roadmap: Migrating a clustered environment" on page 94
Roadmaps provide a high-level overview of complex tasks such as migrating a clustered environment to a new version of IBM WebSphere Portal Express.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

Worksheet

To set up the migration, you answer questions about your wanted configuration. Some fields apply to migration configurations. Some fields are required based on your environment. The remaining fields are advanced and do not apply to most configurations.

Minimal required fields

The following table lists the fields that are unique to the Migrate a cluster step 3: Upgrade node profiles configuration option. You might be prompted for additional information about system or user IDs and passwords that you defined during the portal installation process.

Table 107. Minimal required fields

Field Label	Default	Your Value
Target operating system	Linux	
Target portal profile name	wp_profile	
Target portal profile home directory	/opt/IBM/WebSphere/wp_profile	
Database management software	Derby	
WebSphere Application Server administrator	wpsadmin	
WebSphere Application Server administrator password		
Target deployment manager host name		
Target deployment manager soap port **Note:** Enter the same port number used for the source environment.	10033	
What is the new host name		
Where is the target portal installed	/opt/IBM/WebSphere/PortalServer	

Advanced fields

The following table lists the advanced fields that are unique to the Migrate a cluster step 3: Upgrade node profiles configuration option. Click **Advanced** on the Answer Questions page for the target node to see the advanced properties. Default values are provided for advanced fields that are required.

Table 108. Advanced fields

Field Label	Default	Your Value
JVM heap size	2048	

Migrate a cluster step 3: Upgrade node profiles option

After you answer questions and provide information about your migration, the wizard generates a custom configuration procedure.

About this task

Depending on your environment, the wizard generates a configuration process. The following steps reflect all possible steps in the configuration process. The steps do not represent a literal configuration. The steps are provided as a reference.

If you click **View Step Command**, you can see the task and properties that are associated with each step in the wizard.

Procedure

1. Manual Step: Update the ports for the deployment manager and nodes

   Condition
   none

   ConfigEngine task
   none
2. Upgrade the ConfigEngine

   Condition
   none

ConfigEngine task
none

3. Update database settings

   Condition
   none

   ConfigEngine task
   none

4. Validate the database settings

   Condition
   none

   ConfigEngine task
   validate-database

5. Connect to new databases

   Condition
   none

   ConfigEngine task
   connect-database

6. Manual Step: Review database schema changes

   Condition
   IBM z/OS

   DB2

   ConfigEngine task
   none

7. Upgrade the base portal database component

   Condition
   IBM z/OS

   DB2

   ConfigEngine task
   grant-runtime-db-user-privileges

   upgrade-database

8. Manual Step: Remove check pending statuses from table spaces

   Condition
   IBM z/OS

   DB2

   ConfigEngine task
   none

9. Upgrade the remaining portal databases

   Condition
   IBM z/OS

   DB2

   ConfigEngine task
   grant-runtime-db-user-privileges

   upgrade-database

10. Upgrade the portal profile

Condition
none

ConfigEngine task
upgrade-profile

Version 7.0 server-only migration: Do not complete the upgrade profile step using the Configuration Wizard. Instead you must run this task manually. For more information about running this task manually, see "Migration from Portal 7.0 server-only to Portal 8.5" on page 802.

Note: When you run this step, the sub task that is named **action-deploy-portlets-applyMIGStatic-wp.oob.full** runs and completes successfully. However, the following error messages are shown. You can ignore these error messages:

- `EJPXA0161W: The web module ContactList could not be activated. Please see previous messages for reasons and possible corrective actions.`
- `EJPPH0048W: The synchronization mode of all nodes in the portal cluster is not consistently set. The portlet application PA_ContactList will not be started in the Application Server. Manual synchronization is assumed for all nodes. Manually start the application after all nodes were synchronized.`
- `EJPXA0067E: The following configuration data is needed to create a content-node resource: content-parentref.`

What to do next

You must repeat the **Migrate a Cluster Step 3: Upgrade Node Profiles** steps on every node in the cluster. The task completes more quickly on the secondary nodes because it does not perform the same application and database upgrades that were performed on the primary node.

To complete migration, you must perform several post-migration tasks that depend on how you use WebSphere Portal Express.

1. Review the Next steps section of the product documentation.
   - Run the **PRE-APPLY-FIX** and **APPLY-FIX** tasks from Applying the latest combined cumulative fix updates.
   - Complete the post-migration activities that apply to how you are using WebSphere Portal Express before you move on to the next step. For example, if you are using a virtual portal, then complete the virtual portal post-migration steps.
   - Start the enabling new functionality tasks only after you complete the post-migration tasks.

Next steps

To complete migration, you must first perform several post-migration steps that depend on how Portal is being used. After completing the post-migration steps, review the Enabling new functionality section to take advantage of the new tools available in IBM WebSphere Portal Express Version 8.5. Enabling new functionality should not be started until all post-migration steps have been completed.

"Post-migration activities" on page 857
After you migrate to IBM WebSphere Portal Express Version 8.5, you need to complete extra tasks depending on how you customized the source portal

environment and which components you used. First, complete the **Applying the latest combined cumulative fix updates** task, then you can begin the post-migration tasks followed by enabling new functionality.

"Enabling new functionality in a migrated portal" on page 909
The migration process collects configuration data and applications from an earlier installed version of IBM WebSphere Portal Express and merges them into the newer installed version so that the new environment is identical to the earlier environment. Taking advantage of new functionality that was not available in the earlier portal requires additional attention after migration is complete.

Post-migration activities

After you migrate to IBM WebSphere Portal Express Version 8.5, you need to complete extra tasks depending on how you customized the source portal environment and which components you used. First, complete the **Applying the latest combined cumulative fix updates** task, then you can begin the post-migration tasks followed by enabling new functionality.

"Applying the latest combined cumulative fix updates"
This is a required task. After you migrate using the Configuration Wizard, you must run two tasks to ensure that all of the combined cumulative fix updates are applied to your system before you complete other post-migration or enablement tasks.

"Administrative tasks" on page 858
To ensure that your new environment functions properly, complete administrative tasks such as enabling automatic synchronization, migrating web server configurations, configuring a federated LDAP user registry, and more.

"Portal tasks" on page 866
Complete the post-migration tasks specific to the way that you use WebSphere Portal Express.

"Development tasks" on page 892
Complete the development tasks that are required after the migration to ensure that your environment functions properly.

"Database tasks" on page 894
There are different post-migration tasks that are required depending on the type of database that is used.

"Add-ons, features, and third-party integration tasks" on page 896
Complete the post-migration tasks that are required based on the way you use your WebSphere Portal Express environment.

Applying the latest combined cumulative fix updates

This is a required task. After you migrate using the Configuration Wizard, you must run two tasks to ensure that all of the combined cumulative fix updates are applied to your system before you complete other post-migration or enablement tasks.

Procedure

1. Stop the target environment:
   - If you migrated a stand-alone server, stop the WebSphere_Portal server.
   - If you migrated a cluster, stop the WebSphere_Portal server and node agent. Then, ensure that the deployment manager is started.
2. Run the **PRE-APPLY-FIX** task from the *wp_profile_root*/ConfigEngine directory:

Linux `./ConfigEngine.sh PRE-APPLY-FIX -DWasPassword=`*password* `-DPortalAdminPwd=`*password*

IBM i `ConfigEngine.sh PRE-APPLY-FIX -DWasPassword=`*password* `-DPortalAdminPwd=`*password*

Windows
`ConfigEngine.bat PRE-APPLY-FIX -DWasPassword=`*password* `-DPortalAdminPwd=`*password*

3. Run the **APPLY-FIX** task from the *wp_profile_root*`/ConfigEngine` directory:

   Important: If you are migrating to Portal 8.5 CF03 or CF04, you must specify the `-DForceRun=true` parameter when you run **APPLY-FIX**.

   Linux `./ConfigEngine.sh APPLY-FIX -DWasPassword=`*password* `-DPortalAdminPwd=`*password*

   IBM i `ConfigEngine.sh APPLY-FIX -DWasPassword=`*password* `-DPortalAdminPwd=`*password*

   Windows
   `ConfigEngine.bat APPLY-FIX -DWasPassword=`*password* `-DPortalAdminPwd=`*password*

4. **Cluster only:** Complete each step on every node in the cluster.

Related tasks:

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

Administrative tasks

To ensure that your new environment functions properly, complete administrative tasks such as enabling automatic synchronization, migrating web server configurations, configuring a federated LDAP user registry, and more.

"Enabling automatic synchronization for a clustered environment" on page 859
Before you started migration, you disabled automatic synchronization to prevent the source and target environments from becoming corrupted. Now that the data migration is complete, it is safe to enable this feature on both the source and target environments.

"Enabling unprotected URI Authentication" on page 859
If Security Assertion Markup Language (SAML) is enabled, or if you plan to enable SAML, you must ensure that LTPA authentication works correctly with the web and proxy servers.

"Federating the LDAP user registry" on page 860
The stand-alone LDAP user registry configuration is deprecated. Instead, configure the federated LDAP user registry. Run the **wp-modify-federated-security** task to change to a federated LDAP user registry.

"Migrating web server configurations" on page 861
Migrate a web server so that it supports the latest version of WebSphere Application Server. The Application Migration Toolkit for WebSphere Application Server supports migrating applications from previous versions of WebSphere Application Server to the latest product version.

"Disabling TAI if disabled previously" on page 864
If TAI was disabled before you began migration, and you had to enable it in

order to run the migration, then you might need to disable TAI on both the source and target environments as a post-migration step.

"Reviewing and updating scheduled tasks" on page 865
After you complete the migration by using the Configuration Wizard, review your scheduled tasks to verify that the tasks are still valid for your current environment.

"Moving the personalization page" on page 865
After you migrate to IBM WebSphere Portal Express Version 8.5 and change your theme to Version 8.5, the Personalization welcome page is in the Content application area. You can run a task to move the Personalization welcome page back to the application level.

Enabling automatic synchronization for a clustered environment:

Before you started migration, you disabled automatic synchronization to prevent the source and target environments from becoming corrupted. Now that the data migration is complete, it is safe to enable this feature on both the source and target environments.

Procedure

1. Start the WebSphere Integrated Solutions Console for the deployment manager.
2. Select **System Administration** > **Node Agents**.
3. Click **nodeagent** for the required node.
4. Click **File Synchronization Service** in the **Additional Properties** section.
5. Ensure that **Enable service at server startup** is selected to enable the synchronization service at startup.
6. Ensure that **Automatic Synchronization** is selected to enable the automatic synchronization feature.
7. Click **OK** and **Save**.
8. Repeat these steps for all remaining nodes.
9. Select **System Administration** > **Node Agents**.
10. Select all nodes that previously had automatic synchronization disabled, and click **Restart**.

Related tasks:

"Disabling automatic synchronization to protect your clustered source environment" on page 817
The target environment initially uses the same ports as the source environment. There are three important steps you must complete to ensure that the source and target environments do not become corrupted.

Enabling unprotected URI Authentication:

If Security Assertion Markup Language (SAML) is enabled, or if you plan to enable SAML, you must ensure that LTPA authentication works correctly with the web and proxy servers.

About this task

The **Use available authentication data when an unprotected URI is accessed** setting must be enabled. Enabling this setting ensures that the security session stays active, if the browser context changes from protected to unprotected, for example from `/myportal` to `/portal`.

Procedure

1. Access the WebSphere Integrated Solutions Console.
2. Go to **Security** > **Global security** > **Web and SIP security** > **General settings**.
3. Select **Use available authentication data when an unprotected URI is accessed**.
4. Click **Apply**.
5. Click **Save**.
6. If you are using a stand-alone environment, restart the server. If you are using a cluster environment, synchronize the nodes, and then restart the servers.

Related information:

 Understanding the WebSphere Application Server SAML Trust Association Interceptor

Federating the LDAP user registry:

The stand-alone LDAP user registry configuration is deprecated. Instead, configure the federated LDAP user registry. Run the **wp-modify-federated-security** task to change to a federated LDAP user registry.

Before you begin

In a stand-alone server environment, you can complete the following task when the servers are either stopped or started. In a clustered environment, start the deployment manager and node agent. Then, verify that they are able to synchronize.

About this task

Remember: Starting with IBM WebSphere Portal Express Version 8.5, the stand-alone LDAP repository is deprecated. Change to the federated LDAP user repository.

Use the `wp_security_federated.properties` helper file that is in the *wp_profile_root*`/ConfigEngine/config/helpers` directory. It ensures that the correct properties are entered. In the following instructions, where the step refers to the `wkplc.properties` file, use your `wp_security_federated.properties` helper file.

Procedure

1. Go to the *wp_profile_root*`/ConfigEngine/properties` directory.
2. Open the `wkplc.properties` file with a text editor.
3. Update the following parameters in the `wkplc.properties` file under the VMM Federated repository properties heading:

   Note: Go to the properties file for specific information about the parameters.

   federated.primaryAdminId

   federated.realm

   federated.serverId

   federated.serverPassword
4. Open a command prompt.
5. Change to the *wp_profile_root*`/ConfigEngine` directory.

6. Run the following task to change the configuration to use a federated repository:
   - Linux : `./ConfigEngine.sh wp-modify-federated-security -DWasPassword=password -Dskip.ldap.validation=true`
   - IBM i: `ConfigEngine.sh wp-modify-federated-security -DWasPassword=password -Dskip.ldap.validation=true`
   - Windows: `ConfigEngine.bat wp-modify-federated-security -DWasPassword=password -Dskip.ldap.validation=true`

Important: If you have WebSphere Portal Express Version 8.5 with a CF05 or later fix pack applied, then you do not have to complete the following steps.

7. Stop and restart the appropriate servers to propagate the changes. For instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.
8. Log in to WebSphere Portal Express as an administrator.
   a. Click **Administration**. Then, click **Virtual Portals** > **Manage Virtual Portals**.
   b. Edit each Virtual Portal using the pencil icon.
   c. Set **User realm** as blank.
   d. Click **OK**.
   e. Edit each Virtual Portal using the pencil icon.
   f. Set **User realm** to match the realm ID that you set for **federated.realm**.
   g. Click **OK**.

Migrating web server configurations:

Migrate a web server so that it supports the latest version of WebSphere Application Server. The Application Migration Toolkit for WebSphere Application Server supports migrating applications from previous versions of WebSphere Application Server to the latest product version.

You can upgrade IBM HTTP Server (IHS) from a previous version in two ways:

- Install the new version in a new directory
- Install the new version in the same directory as the previous version

For information about migrating applications, read more about the Application Migration Toolkit.

Related information:

Configuring a web server plug-in using the pct tool

Installing the new version of IBM HTTP Server in a new directory: Procedure

1. Install the Version 8.5 IBM HTTP Server in a new directory.
2. Optional: If you installed IBM HTTP Server into a new directory and retained your previous version of IBM HTTP Server, by default the administration server and the web server use the same ports as the previous version administration server and web server. If you ever run both versions of the IBM HTTP Server simultaneously, port conflicts occur unless you change the port numbers for one of the server versions. To modify the port numbers for one of the IBM HTTP Servers, edit the server configuration files for that IBM HTTP Server. These files are in the `http_server_install/conf` directory.

3. Optional: Migrate web server definitions. A web server definition is used to manage the web server from a stand-alone profile or the deployment manager.
   - If you updated IBM HTTP Server on the original host but in a new directory, update the path by selecting the web server: **Servers** > **Server Types** > **Web Servers in the WebSphere Application Server administration console**.
   - If the updated IBM HTTP Server is on a new host, follow the procedure in "Selecting a web server topology diagram and roadmap" to create a new web server definition. You can remove the old web server definition when you confirm that the new web server is working properly.
4. Install the latest IBM HTTP Server cumulative fix. Learn more about Recommended fixes for IBM HTTP Server.

Installing the new version of IBM HTTP Server in the same directory as the previous version:
Procedure

1. Migrate web server configurations.
   a. Stop the IBM HTTP Server and the IBM HTTP Server administration server.
   b. Copy the existing installation directory to a new location. This action preserves your configuration, keys, and content.
      - If you are using Linux, use the following command to copy the previous installation:

        ```
        cp –rp current_install_directory new_directory_name
        ```

      - If you are using Windows, use the following command to copy the previous installation:

        ```
        xcopy current_install_directory new_directory_name /s /e /k /i
        ```

   c. Uninstall the previous IBM HTTP Server version.
   d. Remove the previous installation directory. Because the uninstallation leaves behind some files, such as modified and added files, fix pack files, and uninstall files, you must manually remove the previous installation directory to complete the uninstallation process. If you had any uninstallation issues, review and backup the uninstall log files in the `http_server_install/logs/uninstall` directory before proceeding.
      - If you are using Linux, use the following command to remove the installation directory:

        ```
        rm -r  current_install_directory
        ```

      - If you are using Windows, use the following command to remove the installation directory:

        ```
        rd /s current_install_directory
        ```

   e. Install the Version 8.5 IBM HTTP Server in the same directory location as the previous version.
   f. Run the Plug-ins Configuration Tool, the pct tool, to configure your web server plug-ins. Refer to Configuring a web server plug-in using the pct tool for information on running the pct tool.
   g. Restore any custom configurations that were made to your previous version of IBM HTTP Server and IBM HTTP Server administration server.
      - Identify your previous customizations.

        If you used the `httpd.conf` configuration files that are provided with the previous version of IBM HTTP Server as the starting point for your configuration files, compare the content of each configuration file, with its corresponding `.default` file, within the directory that contains your previous IBM HTTP Server installation. For example, if you compare the

content of the `httpd.conf` file with the `httpd.conf.default` file, you can see any customizations that were made to the `httpd.conf` file since the original installation. Then, perform similar comparisons for the other configuration files.

If you did not use the `httpd.conf` configuration files that are provided with the previous version of IBM HTTP Server as the starting point for your configuration files, perform a manual analysis to determine your previous settings. In this situation, you might want to compare the settings in the `httpd.conf.default` file that is provided with the new IBM HTTP Server, with the settings in the `httpd.conf.default` file that is provided with the previous IBM HTTP Server version. Use this comparison to identify configuration differences in the two `httpd.conf.default` files. You can then use this information to modify your customized configuration file to work with the Version 8.5 IBM HTTP Server.

Compare the `bin/envars` file to the `bin/envars-std` file within the directory that contains your previous IBM HTTP Server installation. This comparison identifies any customizations that were made to this file.

- Merge the customizations into the newly installed IBM HTTP Server configuration and `envars` files.

  After you identify the configuration customizations that you made to your previous version of IBM HTTP Server, make these same changes, when applicable, to the configuration files for the Version 8.5 IBM HTTP Server.

  If the configuration files contain WebSphere Application Server plug-in statements from previous versions, remove them to prevent duplicates. If you do not remove these statements, when IBM HTTP Server attempts to start the Version 8.5 plug-in binary module, an error might occur that indicates that the module is already loaded.

  The configuration file might also contain duplicate entries for accessing WebSphere Application Server samples. Remove any aliases for previous versions and retain the Version 8.5 entries.

h. Restore HTML content. If your web page content was previously stored under your IBM HTTP Server installation directory, copy those content files from the directory that contains your prior version of IBM HTTP Server into the installation directory for the new version.

i. Copy any SSL KeyFiles that might be within the installation directory of the previous IBM HTTP Server into the new installation directory

2. Change port assignments for coexisting IBM HTTP Servers. To modify the port numbers for one of the IBM HTTP Servers, edit the server configuration files for that IBM HTTP Server. These files are in the `http_server_install/conf` directory.
3. Upgrade Apache plug-in modules.

   There are no Apache API (application programming interface) changes from the previous major release, so there should be no need to rebuild modules that worked with the previous release. However, if you use modules from third-party vendors, contact your vendors to verify that they support the module with the version of IBM HTTP Server to which you are upgrading.

   Apache plug-in modules from sources other than the Version 8.5 IBM HTTP Server must be built to support Apache 2.2. The distributors of modules that are used with older versions of IBM HTTP Server might need to recompile the modules to support Apache 2.2.

- WebSphere Application Server provides a new plug-in for Apache 2.2 and IBM HTTP Server 8.5.
- If you use modules from third-party vendors, contact your vendor for a version of the module that works with the Apache 2.2 API.
- If you use modules that were developed in-house, you must rebuild your modules to support Apache 2.2. The modules might also require some modifications.

4. Update the IBM HTTP Server service name in the WebSphere Application Server web server definition if the following conditions are true:
   - The server is a Windows server.
   - You installed IBM HTTP Server into the same directory where an earlier version was located.
   - You are using a web server definition from that prior installation.

   For an IBM HTTP Server on a Windows server system, use 'Services' to determine the name that is used for the new IBM HTTP Server service, and then update the web server definition to use this service name.
5. Migrate web server definitions. A web server definition is used to manage the web server from a stand-alone profile or the deployment manager.
   - If you updated IBM HTTP Server on the same host and in the same directory, no action is required. The current web server definition suffices.
   - If the updated IBM HTTP Server is on a new host, follow the procedure in Selecting a web server topology diagram and roadmap to create a new web server definition. You can remove the old web server definition when you confirm that the new web server is working properly.
6. Install the latest IBM HTTP Server cumulative fix. Learn more about Recommended fixes for IBM HTTP Server.

Disabling TAI if disabled previously:

If TAI was disabled before you began migration, and you had to enable it in order to run the migration, then you might need to disable TAI on both the source and target environments as a post-migration step.

Procedure

To disable TAI, complete the following task:

1. Log in to the WebSphere Integrated Solutions Console.
2. Go to **Security** > **Global security**.
3. Select **Enable administrative security** and **Enable application security**.
4. On the Global security page, locate the **Authentication** section and click **Web and SIP security** > **Trust association**.
5. Clear **Enable Trust Assocation**.
6. Click **Apply**.
7. Click **Save**.
8. If you have a cluster environment, synchronize the nodes.
9. Restart the server for a stand-alone environment, and restart all servers for a cluster environment.

Related tasks:

"Verifying that WebSphere Application Server Trust Association Interceptor is enabled" on page 818
The automated migration of the WebSphere Portal Express profile requires that the Trust Association Interceptor (TAI) is enabled so that you can configure content in WebDAV during migration.

Reviewing and updating scheduled tasks:

After you complete the migration by using the Configuration Wizard, review your scheduled tasks to verify that the tasks are still valid for your current environment.

About this task

If you performed a remote migration, then you must update the table prefix and **jdni** name for your schedulers, which are based on the host name. You can complete this update by running the **action-clean-scheduled-tasks** ConfigEngine task, which removes all schedulers. After you run this task, you must restart the Portal server, and then the schedulers are re-created with the correct table prefix and **jdni** name. Learn more about Configuring schedulers using the administration console.

Procedure

1. Review your scheduled tasks:
   a. Start the WebSphere Application Server administrative console.
   b. Select **Resources** > **Schedulers**.
2. Run the following task:
   - Linux : `./ConfigEngine.sh action-clean-scheduled-tasks -DWasPassword=`*`password`* `-Drelease.DbPassword=`*`password`*
   - IBM i: `ConfigEngine.sh action-clean-scheduled-tasks -DWasPassword=`*`password`* `-Drelease.DbPassword=`*`password`*
   - Windows: `ConfigEngine.bat action-clean-scheduled-tasks -DWasPassword=`*`password`* `-Drelease.DbPassword=`*`password`*
3. Restart the Portal server so that the default scheduled tasks that are required for operation are created.

Moving the personalization page:

After you migrate to IBM WebSphere Portal Express Version 8.5 and change your theme to Version 8.5, the Personalization welcome page is in the Content application area. You can run a task to move the Personalization welcome page back to the application level.

About this task

Important: This task is required only for versions before CF03.

Run this task to move the Personalization page to the application level. If you do not run this task, the Personalization welcome page remains in the Content application. If the personalization welcome page remains in the Content application, you cannot access the other existing pages in the Content application area.

Procedure

Run the following task using the ConfigEngine to move the Personalization welcome page:

- Linux : `./ConfigEngine.sh move-pzn-page`
- IBM i: `ConfigEngine.sh move-pzn-page`
- Windows: `ConfigEngine.bat move-pzn-page`

Portal tasks

Complete the post-migration tasks specific to the way that you use WebSphere Portal Express.

"Updating the default theme and skin" on page 867
You can add the Portal 8.5 theme and 8.5 skin to all new pages by default.

"Applying the new theme to your Portal pages" on page 868
Custom portal resources, such as themes and skins, are migrated automatically. The Portal 8.5 theme is also deployed when migration is complete. However, you must complete extra steps to apply the theme to your pages.

"Updating page order" on page 869
If the page order was altered for IBM pages in the Applications, Administration, or Hidden Pages areas on your source environment, then the page order might not be correct on the target environment after migration.

"Web Content Manager" on page 869
Extra migration steps are required for updating Web Content Manager after data migration is complete.

"Virtual Portal tasks" on page 876
If you configured WebSphere Portal Express to use virtual portal, there are some additional steps that are needed to complete the migration.

"Importing search web collections" on page 883
As a part of preparing your source environment, you exported web collections. After you export a search web collection from a source portal, you can import the data into a new, empty collection on the target portal. Importing a web collection retains most of the configuration data such as content sources, schedulers, filters, and language settings. If you configured such settings when creating the new collection, they are overwritten by the imported settings.

"Importing UX Screen Flow Manager dialog definitions" on page 884
If you migrated from Version 8.0.0.1 with the UX Screen Flow Manager (UXFM) enabled, then you exported and removed your dialog definitions before migrating to Version 8.5. Run the following task to import the dialog definitions into your upgraded system.

"WSRP" on page 885
If you use your IBM WebSphere Portal Express as a WSRP Consumer or Producer, you must complete additional steps to complete migration.

"Blogs and wikis" on page 887
After you migrate from WebSphere Portal Express Versions 7.0 or 8.0 to Version 8.5, you must run a configuration task to update the presentation templates that are used by blogs and wikis to apply the latest updates. You must run this task for content in your blogs and wikis to render properly.

"Tag and Search Center pages" on page 887
When you migrate to IBM WebSphere Portal Express Version 8.5, the migration process does not apply the Portal 8.5 theme to all portal pages. For example, this affects the Tag and Search Center pages. To continue to use your Tag

Center and Search Center pages, you must update the theme for the pages to the Version 8.5 theme. You must also update the profile of the pages to the Search and Tag Center profile.

"Removing Person Tag hidden pages" on page 888
Remove Person Tag hidden pages from migrated environments where Search for Portal Site was previously configured. If you migrated or updated to WebSphere Portal Express 8.5 CF04 or later, then you do not need to complete these steps.

"Updating URL mapping for personalization page" on page 889
If you migrated from Version 7.0 to Version 8.5 with a CF03 or earlier fix pack applied, then you must complete the following task to ensure that the Personalization page displays properly.

"Enabling vanity URL support after migration" on page 889
In a new WebSphere Portal Express Version 8.5 installation, vanity URL support is enabled. However, if you upgrade your WebSphere Portal Express from a previous version to Version 8.5, vanity URL support is disabled. You can enable and disable vanity URL support as required by using a portal configuration task.

"Enabling impersonation" on page 890
If you migrate to WebSphere Portal Express Version 8.5 and upgrade to fix pack level CF01-CF03, impersonation might not be enabled. If you plan to use the impersonation feature, which allows a selected user to preview and test new pages or portlets to help identify any potential issues, then you might need to enable this feature.

"Reinstalling the PortalTheme" on page 891
If you migrate from Version 7.0 to Version 8.5, when you apply the latest combined cumulative fix during post-migration, the PortalTheme from earlier WebSphere Portal Express versions is removed. If your system requires this theme, then you can manually reinstall it.

Updating the default theme and skin:

You can add the Portal 8.5 theme and 8.5 skin to all new pages by default.

About this task

After migration, a new theme is available for you to apply to your Portal pages. The name of the new theme is `Portal 8.5`. In addition, four new skins are available for the Portal 8.5 theme. For information about the new skins, see *Skins* in the Developing themes and skins section of the documentation.

Procedure

1. Click the **Administration menu** icon. Then, click **Portal User Interface** > **Themes and Skins**.
2. Select the **Portal 8.5** theme from the **Themes** list in the portlet.
3. Click **Set as default portal theme**.
4. Select the **Portal 8.5 Hidden** skin from the **Skins** list in the portlet.
5. Click **Set as default portal skin**.

Related concepts:

"Skins" on page 2682
You can apply ready-use skins to your portal pages, modify the existing skins, or add your own custom skin to change how your pages display.

Applying the new theme to your Portal pages:

Custom portal resources, such as themes and skins, are migrated automatically. The Portal 8.5 theme is also deployed when migration is complete. However, you must complete extra steps to apply the theme to your pages.

After migration, a new theme is available for you to apply to your Portal pages. The name of the new theme is `Portal 8.5`. In addition, four new skins are available for the Portal 8.5 theme. For information about the new skins, see *Skins* in the Developing themes and skins section of the documentation.

Creating a page and applying the theme

If you prefer to create a new set of pages and apply the theme, you can use the Manage Pages portlet or the XML configuration interface. It is possible that some migrated themes allow for inline page creation, and **Portal 8.5** would be an option to select in the process.

To use the Manage Pages portlet, complete the following steps:

1. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
2. In the **Manage Pages** portlet, select the location in Portal to create the new page.
3. Click **New Page** in the portlet.
4. Define a name for the page and select the **Portal 8.5** option from the **Theme** menu.
5. After you complete these steps, the new page with the Portal 8.5 theme is created.

You can also create a page with the Portal 8.5 theme by using the XML configuration interface. This process is helpful for scripting the process across multiple pages. There are samples for modifying Portal resources with the XML configuration. For more information about using the XML configuration interface, see *Sample XML configuration files*.

When you create a new page and apply the Portal 8.5 theme, the new inline customization features are immediately available for use. Use the inline toolbar to create pages.

Related concepts:

"Skins" on page 2682
You can apply ready-use skins to your portal pages, modify the existing skins, or add your own custom skin to change how your pages display.

Related reference:

"Sample XML configuration files" on page 1111
Sample files are provided for your reference to help illustrate how to use XML configuration for different portal configuration purposes. Before you use them, read the other topics about the XML configuration interface carefully. Many of the samples need to be modified with valid page or user name before they can be used.

Updating page order:

If the page order was altered for IBM pages in the Applications, Administration, or Hidden Pages areas on your source environment, then the page order might not be correct on the target environment after migration.

About this task

You can adjust the page order of your pages using the Portal Administration menu or XMLAccess. If you want your pages to be listed first, then ensure that they have ordinal values less than 100. For more information about using XMLAccess to edit the ordinal values for your pages, see XMLAccess Frequently Asked Questions.

Procedure

1. Click **Administration**.
2. Click **Manage Pages**.
3. Use the **Move up** and **Move down** icons to adjust the order of your pages.

   Note: If you are rearranging the tree structure of your pages, then you can use the **Mark** icon to mark the page, and then move it.

Web Content Manager:

Extra migration steps are required for updating Web Content Manager after data migration is complete.

"Updates for Web Content Manager"
These are additional migration steps required for Web Content Manager after data migration is complete.

"Convert the IBM Portlet API Web Content Viewer to the JSR 286 Web Content Viewer" on page 870
If you use the IBM Portlet API Web Content Viewer portlet in your source environment, you must complete the following task after migration to convert to the JSR 286 Web Content Viewer.

"Convert the IBM Portlet API Remote Web Content Viewer to the JSR 286 Web Content Viewer" on page 871
If you use the IBM Portlet API Remote Web Content Viewer portlet in your source environment, you must complete the following task after migration to convert to the JSR 286 Web Content Viewer.

"Converting an IBM API Web Content Viewer to the JSR 286 API" on page 874
As installed by default, the Web Content Viewer is based on the JSR 286 API. If you have a Web Content Viewer that is based on the older IBM API, you can convert the viewer to the JSR 286 API. Use the **convert-wcm-rendering-portlet** task to convert the IBM API Web Content Viewer settings and instances to the JSR 286 Web Content Viewer portlet.

Updates for Web Content Manager:

These are additional migration steps required for Web Content Manager after data migration is complete.

Authoring portlet preferences

Authoring portlet preferences, such as configured libraries and default rich text editor, are not migrated. You must re-configure your authoring portlet after migration to restore your preferences.

JSP files and Web content plug-ins

- Any JSP files used on your old system must be manually copied to your new system.
- Any Web content plug-ins used on your old system, such as custom workflows, must be manually copied to your new system and enabled.

Syndication

If the hostname or context root of your migrated server is different to the original server, you will need to edit the syndicators or subscribers to use the new hostname or context root.

- Edit the Subscriber URL of any syndicators that are syndicating to the migrated server. See "Cross version syndication" on page 806 for details on what versions of WebSphere Portal support cross version syndication.
- Edit the Syndicator URL of any subscribers that are subscribing to the migrated server. See "Cross version syndication" on page 806 for details on what versions of WebSphere Portal support cross version syndication.

Convert the IBM Portlet API Web Content Viewer to the JSR 286 Web Content Viewer:

If you use the IBM Portlet API Web Content Viewer portlet in your source environment, you must complete the following task after migration to convert to the JSR 286 Web Content Viewer.

About this task

The Web Content Viewer portlet that is based on the IBM Portlet API was removed from IBM Web Content Manager Version 8.5 and is no longer supported. The migration process transfers the portlet and the portlet instances from your portal pages to the target environment. After migration, you must replace the IBM Portlet API Web Content Viewer portlet with the JSR 286 Web Content Viewer portlet. You can complete this task manually or with the conversion task as described in the following procedure.

Procedure

1. Check if you installed the IBM Portlet API Web Content Viewer portlet and identify potential portlet clones.
   a. Click the **Administration menu** icon. Then, click **Portlet Management** > **Web Modules**.
   b. Search for the web module with the file name `ilwwcm-localrendering-portlet.war`.
   c. If the search result is empty, then you do not have the IBM Portlet API Web Content Viewer, and you can skip this task.
   d. If the search result is not empty, then click **ilwwcm-localrendering-portlet.war** > **Web Content Management - Content Viewer**.
2. Check if you installed the JSR 286 Web Content Viewer portlet.

a. Click the **Administration menu** icon. Then, click **Portlet Management** > **Web Modules**.
b. Search for the web module with the file name `ilwwcm-localrenderingportlet-jsr`.
c. If the search result is empty, install the JSR 286 Web Content Viewer portlet from `PortalServer_root/pzn.ext/portlet.localrendering.jsr/localrendering.war/installableApps/ilwwcm-localrenderingportlet-jsr`.

3. To convert the instances of the IBM Portlet API Web Content Viewer portlet to the JSR 286 Web Content Viewer portlet, follow the steps that are given at "Converting an IBM API Web Content Viewer to the JSR 286 API" on page 874.

   Important: If you use clones of the IBM Portlet API Web Content Viewer portlet, you must also convert their instances.
4. After you replace all instances of the IBM Portlet API Web Content Viewer portlet and of its portlet clones, uninstall the `ilwwcm-localrendering-portlet.war` web module.

Convert the IBM Portlet API Remote Web Content Viewer to the JSR 286 Web Content Viewer:

If you use the IBM Portlet API Remote Web Content Viewer portlet in your source environment, you must complete the following task after migration to convert to the JSR 286 Web Content Viewer.

About this task

The Remote Web Content Viewer portlet that is based on the IBM Portlet API was removed from IBM Web Content Manager Version 8.5 and is no longer supported. The migration process transfers the portlet and the portlet instances from your portal pages to the target environment. After migration, you must replace the IBM Portlet API Remote Web Content Viewer portlet with the JSR 286 Web Content Viewer portlet.

Procedure

1. Check if you installed the IBM Portlet API Web Content Viewer portlet and identify potential portlet clones.
   a. Click the **Administration menu** icon. Then, click **Portlet Management** > **Web Modules**.
   b. Search for the web module with the file name `ilwwcm-remoterendering-portlet.war`.
   c. If the search result is empty, then you do not have the IBM Portlet API Remote Web Content Viewer, and you can skip this task.
   d. If the search result is not empty, then click **ilwwcm-remoterendering-portlet.war** > **Web Content Management - Remote Content Viewer**.

      Note: If you do not use clones of the portlet, the displayed table will contain only a single portlet with the title **Remote Web Content Viewer**.
2. If you use the IBM Portlet API Remote Web Content Viewer portlet, choose one of the following methods to deliver your web content after completing the post migration steps:

- Option 1: If you migrate from a source rendering portal that does not include IBM Web Content Manager to a target rendering portal that does not include IBM Web Content Manager, you must use the JSR 286 Web Content Viewer and the WSRP support in the portal.
- Option 2: If you migrate from a source rendering portal that does not include IBM Web Content Manager to a target rendering portal that includes IBM Web Content Manager, the preferred way to display your web content is to locally render it with the JSR 286 Web Content Viewer.
- Option 3: If your source rendering portal includes IBM Web Content Manager, the target rendering portal will also include IBM Web Content Manager. The preferred way to display your web content is to locally render it with the JSR 286 Web Content Viewer.

The following table summarizes the preferred options depending on the source and target rendering portal.

Table 109. Preferred way to display web content in your portal after migration

Source rendering portal	Target rendering portal	Preferred option
Includes the IBM Web Content Manager	Includes the IBM Web Content Manager	Option 3: Local web content rendering with JSR 286 Web Content Viewer
Includes the IBM Web Content Manager	Does not include the IBM Web Content Manager	Unsupported migration path
Does not include IBM Web Content Manager	Includes the IBM Web Content Manager	Option 2: Local web content rendering with JSR 286 Web Content Viewer
Does not include IBM Web Content Manager	Does not include the IBM Web Content Manager	Option 1: Remote web content rendering with JSR 286 Web Content Viewer via WSRP

Option 1: Remote web content rendering with JSR 286 Web Content Viewer via WSRP:

The source rendering portal of the migration does not contain the web content it delivers. Instead the IBM Portlet API Remote Web Content Viewer connects to another portal of your source environment to retrieve the web content.

About this task

As the target rendering portal does not include IBM Web Content Manager, you must continue displaying web content remotely. To perform remote rendering with IBM Web Content Manager version 8.5, you must configure your target environment to use the JSR 286 Web Content Viewer with WSRP.

The target rendering portal that does not include IBM Web Content Manager and still uses the IBM Portlet API Remote Web Content Viewer becomes the WSRP consumer. It consumes the JSR 286 Web Content Viewer from the target authoring portal that includes IBM Web Content Manager and contains the web content.

Procedure

1. To configure remote rendering, follow the instructions at "Enabling remote rendering with WSRP and the Web Content Viewer" on page 2069.
2. On the target rendering portal that acts as WSRP consumer:

a. Add instances of the consumed JSR 286 Web Content Viewer portlet to portal pages that contain instances of the IBM Portlet API Remote Web Content Viewer portlet.
b. Configure the consumed JSR 286 Web Content Viewer portlet instances to match the configuration of the IBM Portlet API Remote Web Content Viewer portlet instances on each of the portal pages.
c. After replacing all instances of the IBM Portlet API Remote Web Content Viewer portlet and of its portlet clones, uninstall the `ilwwcm-remoterendering-portlet.war` web module.

Note: This option has limitations that do not exist when rendering web content locally. For more information, see *Performing remote rendering with WSRP and the web content viewer*.

Options 2 and 3: Local web content rendering with JSR 286 Web Content Viewer:

The source rendering portal of the migration does not contain the web content it delivers. Instead the IBM Portlet API Remote Web Content Viewer connects to another portal of your source environment to retrieve the web content.

About this task

If you choose this option, you will change the web content delivery model from remote rendering to local rendering. After you make the web content libraries available on the target rendering portal, you can use the JSR 286 Web Content Viewer portlet for local rendering.

Procedure

1. To make the web content available on the target rendering portal, syndicate the web content libraries from the target authoring portal to the target rendering portal. For more information, see Syndication.
2. On the target rendering portal:
   a. Check if you installed the JSR 286 Web Content Viewer portlet.
      1) Click the **Administration menu** icon. Then, click **Portlet Management** > **Web Modules**.
      2) Search for the web module with the file name `ilwwcm-localrenderingportlet-jsr.war`.
      3) If the search result is empty, install the JSR 286 Web Content Viewer portlet from `PortalServer_root/pzn.ext/portlet.localrendering.jsr/localrendering.war/installableApps/ilwwcm-localrenderingportlet-jsr.war`.
   b. To convert the instances of the IBM Portlet API Remote Web Content Viewer portlet to the JSR 286 Web Content Viewer portlet, follow the steps given at Converting an IBM API web content viewer to the JSR 286 API.

      Important: If you use clones of the IBM Portlet API Remote Web Content Viewer portlet, you also need to convert their instances.
   c. After replacing all instances of the IBM Portlet API Web Content Viewer portlet and of its portlet clones, uninstall the `ilwwcm-localrendering-portlet.war` web module.

Converting an IBM API Web Content Viewer to the JSR 286 API:

As installed by default, the Web Content Viewer is based on the JSR 286 API. If you have a Web Content Viewer that is based on the older IBM API, you can convert the viewer to the JSR 286 API. Use the **convert-wcm-rendering-portlet** task to convert the IBM API Web Content Viewer settings and instances to the JSR 286 Web Content Viewer portlet.

About this task

The **convert-wcm-rendering-portlet** task converts portlet settings of the IBM API Web Content Viewer to portlet preferences of the JSR 286 Web Content Viewer. The task also converts instances of the IBM API Web Content Viewer to instances of the Web Content Viewer. User customized portlet data that is associated with the portlet instance is converted into portlet preferences.

Procedure

1. Update the file *wp_profile_root*/ConfigEngine/wkplc.properties. Confirm that the user IDs and passwords are set as required, or modify them if necessary.
2. Update or verify the properties in the file *wp_profile_root*/PortalServer/wcm/config/portletconversion.properties.

   Note: If the following conditions are true, no changes are required and you can use the default values in the properties file.
   - There are no clones of the IBM API Web Content Viewer.
   - You want to convert all instances of the portlet on all pages in the default virtual portal.

   For specific situations, you can update the additional properties that are described in the following table.

Scenario	Properties to modify
You want to convert instances of the portlet on specific pages in the default virtual portal.	**pages.uniquename** Specify a list of unique names of pages, separated by commas. If you specify this property, only portlets on those pages and their descendants are converted. If this property is empty or missing, instances of the IBM API Web Content Viewer on all pages are converted.
You want to convert instances of the portlet in a virtual portal that is not the default virtual portal.	**xmlaccess.url** Specify the URL of the virtual portal to the portal XML configuration interface servlet. You can use this property to run conversions for specific virtual portals. If this property is empty or missing, the default portal is used to run the conversion. Example: xmlaccess.url=http://www.example.com:10039/wps/config/vp1

Scenario	Properties to modify
You cloned the Web Content Viewer portlet and want to convert instances of the clone.	Identify the clone by specifying one of the following properties: • **ibmportlet.portletname** • **ibmportlet.uniquename** • **ibmportlet.objectid** Only one of the properties is required to identify the portlet. For a complete list of properties for the portlet conversion task, see *Converting portlet instances and settings from the IBM API to the standard API*.

3. Change to the directory *wp_profile_root*/ConfigEngine.
4. Run the task ConfigEngine convert-wcm-rendering-portlet.

   Windows

   ```
   ConfigEngine.bat convert-wcm-rendering-portlet
   ```

   AIX, HP-UX, Linux, Solaris

   ```
   ./ConfigEngine.sh convert-wcm-rendering-portlet
   ```

   IBM i `ConfigEngine.sh convert-wcm-rendering-portlet`
5. Verify the conversion by reviewing the console. The message Build successful indicates a successful conversion. If the message Build failed is displayed upon completion of the task, review the previous steps.
6. Verify the configuration of the converted Web Content Viewer. For more information about configuring a local Web Content Viewer, see the portlet help.
7. After the successful conversion, you can uninstall the IBM API Web Content Viewer.

What to do next

If you make the web content available on your delivery portal by using syndication, you can also convert instances of the IBM API Remote Web Content Viewer portlet to the JSR 286 portlet. When you make the web content available, you can transform your web content delivery model from remote rendering to local rendering. To convert instances and settings of the IBM API Remote Web Content Viewer portlet to the JSR 286 portlet, proceed by the following steps:

1. Edit the file wp_profile_root/PortalServer/wcm/config/portletconversion.properties.
2. Change the following parameters as described here:

   ibmwebapp.uid
   The default value for this parameter is the unique identifier of the IBM API Web Content Viewer. Change the value to match the unique identifier of the IBM API Remote Web Content Viewer: com.ibm.workplace.wcm.app.ui.portlet.RenderingPortlet.30f9cb100a340018159bfdc6578

   ibmportlet.portletname
   The default value for this parameter is the portlet name of the IBM API Web Content Viewer. Change the value to match the portlet name of the IBM API Remote Web Content Viewer: Remote Web Content Viewer.
3. Perform steps 1 - 5 of the procedure that is given earlier.

4. Verify the configuration of the converted Remote Web Content Viewer portlet. For further information on configuring a local Web Content Viewer portlet, read *Editing the settings of a web content viewer portlet*.
5. After the successful conversion, you can uninstall the IBM API Remote Web Content Viewer portlet.

Note: If you made clones of the IBM API Web Content Viewer portlet or of the IBM API Remote Web Content Viewer portlet, or if you renamed the portlets, you might need to modify the parameters in the file `wp_profile_root/PortalServer/wcm/config/portletconversion.properties` that identify the portlet that you use as the source for the conversion. For a complete reference of all parameters that the portlet conversion task supports, read *Converting portlet instances and settings from the IBM API to the standard API*.

Note: Note: In the version 8.5 user interface, the JSR 286 Web Content Viewer portlet has been renamed to Web Content Viewer.

Related concepts:

"Unsupported and deprecated features for V8.5" on page 17
Review the features that were available in previous versions of IBM WebSphere Portal Express but are no longer available.

Related tasks:

"Converting portlet instances and settings from the IBM API to the standard API" on page 3121
The portal provides a portlet conversion task that allows you to convert the settings and instances of IBM API portlets to the corresponding standard API portlets. This is useful when you intend to replace IBM API portlets by standard API portlets.

Related information:

Editing the settings of a Web Content Viewer Portlet

Virtual Portal tasks:

If you configured WebSphere Portal Express to use virtual portal, there are some additional steps that are needed to complete the migration.

"Updating administration themes in Virtual Portals" on page 877
During migration, the IBM WebSphere Portal Express Version 8.5 theme is deployed. However, only the administration themes on the default virtual portal are updated to the 8.5 version of the administration user interface. For all other virtual portals, you must manually update the administration themes.

"Enabling the 8.5 site toolbar" on page 878
During migration, the IBM WebSphere Portal Express Version 8.5 theme is deployed. The 8.5 theme includes the site toolbar, but the toolbar is enabled only for the default virtual portal. For all other virtual portals, including migrated and newly created virtual portals, you must install the site toolbar separately.

"Updating Web Content Manager pages theme" on page 879
Before using Web Content Manager in virtual portals that are created after migration by using default portal content from an older release, you must update the theme to ⌂Portal 8.5 theme. Otherwise, Web Content Manager related portlets do not work properly. If you upgraded or migrated to WebSphere Portal Express Version 8.5 CF04 or later, then you do not need to complete the following steps.

"Sharing Web Content Manager libraries between Virtual Portals" on page 880
When you created Web Content Manager libraries in the base portal of earlier versions of WebSphere Portal Express, these libraries were available in the virtual portals. In WebSphere Portal Express Version 8.0 and 8.5, a Web Content Manager library that you create in the base portal is not available in the virtual portals.

"Updating scripts and removing deprecated features" on page 880
To create new Virtual Portals with the same content as the portal from which you migrated, you must pre-configure the default content for creating the Virtual Portal. You must also remove or replace references to deprecated features.

"Enabling Tag and Search Center pages for virtual portals" on page 882
When you migrate to IBM WebSphere Portal Express Version 8.5, the migration process does not apply the Portal 8.5 theme to all portal pages for your virtual portals. This affects the Tag and Search Center pages. To use your Tag Center and Search Center pages, you must manually update the theme for the pages to the Version 8.5 theme. You must also update the profile of the pages to the Search and Tag Center profile.

Related tasks:

"Removing Person Tag hidden pages" on page 888
Remove Person Tag hidden pages from migrated environments where Search for Portal Site was previously configured. If you migrated or updated to WebSphere Portal Express 8.5 CF04 or later, then you do not need to complete these steps.

Related information:

"Administering virtual portals" on page 1386
View information to help you scope your WebSphere Portal Express to have multiple virtual portals.

"Tasks for administering virtual portals" on page 1389
Administering virtual portals and their content comprises the tasks described in the following topics.

Updating administration themes in Virtual Portals:

During migration, the IBM WebSphere Portal Express Version 8.5 theme is deployed. However, only the administration themes on the default virtual portal are updated to the 8.5 version of the administration user interface. For all other virtual portals, you must manually update the administration themes.

Procedure

Upgrade the administration pages theme:

1. Run the following commands from the *wp_profile_root*/ConfigEngine directory of your portal installation.

   For a context root based virtual portal:

   - AIX Linux Solaris: `./ConfigEngine.sh action-upgrade-theme-admin-oob -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`* `-DVirtualPortalContext=`*`virtual_portal_context_url`*
   - Windows: `ConfigEngine.bat action-upgrade-theme-admin-oob -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`* `-DVirtualPortalContext=`*`virtual_portal_context_url`*
   - IBM i: `ConfigEngine.sh action-upgrade-theme-admin-oob -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`* `-DVirtualPortalContext=`*`virtual_portal_context_url`*

To find your virtual portal context, run the following command:

- AIX Linux Solaris: `./ConfigEngine.sh list-all-virtual-portals`
- Windows: `ConfigEngine.bat list-all-virtual-portals`
- IBM i: `ConfigEngine.sh list-all-virtual-portals`

2. If IBM Web Content Manager is installed, then run the following command to upgrade the Web Content Manager administration pages theme.

   For a context root based virtual portal:

   - AIX Linux Solaris: `./ConfigEngine.sh action-upgrade-wcm-theme-admin-oob -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`* `-DVirtualPortalContext=`*`virtual_portal_context_url`*
   - Windows: `ConfigEngine.bat action-upgrade-wcm-theme-admin-oob -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`* `-DVirtualPortalContext=`*`virtual_portal_context_url`*
   - IBM i: `ConfigEngine.sh action-upgrade-wcm-theme-admin-oob -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`* `-DVirtualPortalContext=`*`virtual_portal_context_url`*

   To find your virtual portal context, run the following command:

   - AIX Linux Solaris: `./ConfigEngine.sh list-all-virtual-portals`
   - Windows: `ConfigEngine.bat list-all-virtual-portals`
   - IBM i: `ConfigEngine.sh list-all-virtual-portals`

Enabling the 8.5 site toolbar:

During migration, the IBM WebSphere Portal Express Version 8.5 theme is deployed. The 8.5 theme includes the site toolbar, but the toolbar is enabled only for the default virtual portal. For all other virtual portals, including migrated and newly created virtual portals, you must install the site toolbar separately.

About this task

The site toolbar requires a Portal 8.0 theme or a custom Portal 8.0 theme. The theme must be a modularized theme, which supports theme profiles and theme modules. For more information, see *Add the WebSphere Portal Version 8.5 site toolbar to a WebSphere Portal 8.0 theme*.

If you want to use the IBM WebSphere Portal Express Version 8.5 site toolbar on other migrated virtual portals, complete the following configuration task after you remove the old toolbar from your theme.

Cluster only: Complete this step only on the primary node.

Procedure

Go to the *wp_profile_root*`/ConfigEngine` directory of your portal installation and run:

- AIX HP-UX Linux Solaris: `./ConfigEngine.sh install-toolbar -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`* `-DVirtualPortalContext=`*`virtual_portal_context_url`*
- IBM i: `ConfigEngine.sh install-toolbar -DWasPassword=`*`password`* `-DPortalAdminPwd=`*`password`* `-DVirtualPortalContext=`*`virtual_portal_context_url`*

- Windows: `ConfigEngine.bat install-toolbar -DWasPassword=password -DPortalAdminPwd=password -DVirtualPortalContext=virtual_portal_context_url`

Note: Run the **list-all-virtual-portals** ConfigEngine task to find your virtual portal context.

What to do next

You must restart the Portal server after you run the **install-toolbar** task.

For more information about using the 8.5 toolbar, see the site toolbar documentation.

Related concepts:

"Migration - Add the version 8.5 site toolbar to a version 8.0 theme" on page 921
You can easily add the modularized site toolbar of WebSphere Portal Express Version 8.5 to a WebSphere Portal Express 8.0 theme. Or, you can add a toolbar to a custom theme that is derived from the WebSphere Portal Express 8.0 theme. The theme must be a modularized theme, which supports theme profiles and theme modules.

Updating Web Content Manager pages theme:

Before using Web Content Manager in virtual portals that are created after migration by using default portal content from an older release, you must update the theme to ⌂Portal 8.5 theme. Otherwise, Web Content Manager related portlets do not work properly. If you upgraded or migrated to WebSphere Portal Express Version 8.5 CF04 or later, then you do not need to complete the following steps.

Procedure

1. Run the following configuration task from the *wp_profile_root*/ConfigEngine directory:

   Windows
   ```
   ConfigEngine.bat update-wcm-pages -DWasPassword=password
   -DPortalAdminPwd=password
   ```

   AIX Linux Solaris
   ```
   ./ConfigEngine.sh update-wcm-pages -DWasPassword=password
   -DPortalAdminPwd=password
   ```

   IBM i
   ```
   ConfigEngine.sh update-wcm-pages -DWasPassword=password
   -DPortalAdminPwd=password
   ```

2. Change the following pages to use Portal 8.5 Theme:
   - ⌂com.ibm.wps.hiddenpage.wcm.Authoring_Portlet
   - ⌂ibm.portal.Web.Content.Management
   - ⌂ibm.portal.Portal Content

   a. Log in to Portal as an administrator.
   b. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
   c. In the **Search by** menu, select **Unique name contains**.
   d. In the **Search** field, enter the unique name of the page and click **Search**.
   e. Click **Edit Page Properties** for the Web Content Manager page.
   f. On the **Theme** field, select **Portal 8.5**.

g. Click **OK**.

Sharing Web Content Manager libraries between Virtual Portals:

When you created Web Content Manager libraries in the base portal of earlier versions of WebSphere Portal Express, these libraries were available in the virtual portals. In WebSphere Portal Express Version 8.0 and 8.5, a Web Content Manager library that you create in the base portal is not available in the virtual portals.

About this task

If you have managed pages enabled and want to share a Web Content Manager library between virtual portals, use syndication to replicate the library to the virtual portals.

Related concepts:

"Syndication" on page 448
Use syndication to replicate web content library data from one server to another server. Syndication is based on a syndicator and subscriber relationship. The syndicator has the current data. The subscriber received the current data from the syndicator.

Updating scripts and removing deprecated features:

To create new Virtual Portals with the same content as the portal from which you migrated, you must pre-configure the default content for creating the Virtual Portal. You must also remove or replace references to deprecated features.

About this task

When you migrate a Virtual Portal, WebSphere Portal Express treats all pages in the portal as custom, customer-created content. As a result, if the Virtual Portal contains pages that are associated with features that are not available in the new installed version, WebSphere Portal Express migrates those pages regardless.

For example, if you migrate a Virtual Portal that contains a Document Libraries page, WebSphere Portal Express preserves that page. You can remove these pages manually after you migrate the Virtual Portal.

Procedure

1. Register your *filename*.xml file that you used in the previous system to pre-configure the default content for creating the Virtual Portal.

   Note: The name of the file might change based on the release and editions of portal that you are using. The following are examples of file names: "InitVirtualContentPortal.xml, InitAdminVirtualPortal.xml."

Important: If you are migrating to WebSphere Portal Express 8.5 CF04 or later, you can skip the following steps.

2. Use the WebSphere Integrated Solutions Console to update the virtual portal XML scripts to remove references to Dynamic Person Tag portlet.

   Note: Your script can contain references to Dynamic Person Tag portlet. This portlet is no longer available and any reference to this portlet causes your script to fail.

   a. Go to the WebSphere Integrated Solutions Console.

b. Click **Applications** > **Application Types** > **Assets**.
c. Select `VirtualPortal.zip`, and click **Export**.
d. Remove the web-app sequence that includes the opening `<web-app...>` and the closing `</web-app>` tags, and everything in between.

   See the following example to see how the portlet is defined in a web-app sequence:

```
<web-app action="locate" uid="com.ibm.wkplc.people.portal.portlet.dynamicpersontag.web.app"
  <portlet-app action="locate" uid="com.ibm.wkplc.people.portal.portlet.dynamicpersontag.po
    <portlet action="locate" name="Dynamic Person Tag" objectid="3_CGAH47L008DE402BK8543I1C
  </portlet-app>
</web-app>
```

e. Remove the Person Tag page that includes the opening `<content-node...>` and the closing `</content-node>` tags, and everything in between. The object `ids`, `portletrefs`, `sharerefs`, and other references in the example might vary in your installation.

   See the following example of the page that is defined in the content-node sequence.

```
<content-node type="page" uniquename="ibm.portal.Person.Tag">
  <supported-markup markup="html" update="set"></supported-markup>
  <localedata locale="en">
<title>Person Tag</title>
<description>Person Tag portlet, which enables live names and information for names in IBM
  </localedata>
</content-node>
```

f. Select `VirtualPortal.zip`, and click **Update**.
g. Select the default value **Replace entire asset**.
h. Locate the updated file and upload the file.

3. Use the WebSphere Integrated Solutions Console to update the virtual portal XML scripts to remove references to CAI/TAI portlets. For example, the portlets are defined in a web-app sequence like the following examples.

   Note: Your script can contain references to the CAI/TAI portlets. These portlets are no longer available and any reference to these portlets cause your script to fail.

```
<web-app action="locate" uid="com.ibm.workplace.community.portal">
  <portlet-app action="locate" uid="com.ibm.workplace.community.portal.1">
    <portlet action="locate" name="Community" objectid="Z3_CGAH47L0008270I7MOUHLP18C5"/>
  </portlet-app>
</web-app>
```

```
<web-app action="locate" uid="com.ibm.workplace.builder.parameterPortlet.ParamConfigPortlet.40
  <portlet-app action="locate" uid="com.ibm.workplace.builder.parameterPortlet.ParamConfigPort
    <portlet action="locate" name="Parameters" objectid="Z3_CGAH47L0008270I7MOUHLP18O5"/>
  </portlet-app>
</web-app>
```

```
<web-app action="locate" uid="com.ibm.workplace.policystatus.PolicyStatus">
  <portlet-app action="locate" uid="com.ibm.workplace.policystatus.PolicyStatus.1">
    <portlet action="locate" name="Policy Status Portlet" objectid="Z3_CGAH47L0008270I7MOUHLP1
  </portlet-app>
</web-app>
```

```
<web-app action="locate" uid="com.ibm.workplace.builder.propertiesPortlet.portal">
  <portlet-app action="locate" uid="com.ibm.workplace.builder.propertiesPortlet.portal.1">
    <portlet action="locate" name="Properties portlet" objectid="Z3_CGAH47L0008270I7MOUHLP1845
  </portlet-app>
</web-app>
```

```
<web-app action="locate" uid="com.ibm.workplace.builder.manageroles.ManageRoles.50047239651b00181
  <portlet-app action="locate" uid="com.ibm.workplace.builder.manageroles.ManageRoles.50047239651
    <portlet action="locate" name="Roles portlet" objectid="Z3_CGAH47L0008270I7MOUHLP18K5"/>
  </portlet-app>
</web-app>
```

```
<web-app action="locate" uid="com.ibm.workplace.cdo.portal">
      <portlet-app action="locate" uid="com.ibm.workplace.cdo.portal.1">
        <portlet action="locate" name="Application Catalog" objectid="Z3_CGAH47L00OJ790IAH1AFAN10
      </portlet-app>
    </web-app>
```

```
<web-app action="locate" uid="com.ibm.workplace.tai.tc">
      <portlet-app action="locate" uid="com.ibm.workplace.tai.tc.1">
        <portlet action="locate" name="Template Catalog Manager" objectid="Z3_CGAH47L00OJ790IAH1A
      </portlet-app>
    </web-app>
```

The pages that reference these portlets in sequences like the following examples.

```
<component action="update" active="true" deletable="false" domain="rel" modifiable="undefined" ob
  <portletinstance action="update" domain="rel" objectid="Z5_CGAH47L0008270I7MOUHLP18E0" portletr
</component>
```

```
<component action="update" active="true" deletable="false" domain="rel" modifiable="undefined" ob
  <portletinstance action="update" domain="rel" objectid="Z5_CGAH47L0008270I7MOUHLP18I4" portletr
</component>
```

```
<component action="update" active="true" deletable="false" domain="rel" modifiable="undefined" ob
  <portletinstance action="update" domain="rel" objectid="Z5_CGAH47L0008270I7MOUHLP18U2" portletr
</component>
```

```
<component action="update" active="true" deletable="false" domain="rel" modifiable="undefined" ob
  <portletinstance action="update" domain="rel" objectid="Z5_CGAH47L0008270I7MOUHLP18Q1" portletr
</component>
```

```
<component action="update" active="true" deletable="false" domain="rel" modifiable="undefined" ob
  <portletinstance action="update" domain="rel" objectid="Z5_CGAH47L0008270I7MOUHLP1863" portletr
</component>
```

The `portletref` attribute of the `portletinstance` tag matches one of the `objectid` attributes from the portlet tags in the web-app section.

```
<portlet action="locate" name="Roles portlet" objectid="Z3_CGAH47L0008270I7MOUHLP18K5"/>
<portletinstance portletref="Z3_CGAH47L0008270I7MOUHLP18K5" ... />
```

Make sure to remove the web-app sequence that includes the opening `<web-app...>` and the closing `</web-app>` tags, and everything in between. Also, remove the components that reference the portlets. It includes the opening `<component ...>` and the closing `</component>` tags, and everything in between. The `object ids`, `portletrefs`, `sharerefs`, and other references in the example might vary in your installation.

Enabling Tag and Search Center pages for virtual portals:

When you migrate to IBM WebSphere Portal Express Version 8.5, the migration process does not apply the Portal 8.5 theme to all portal pages for your virtual portals. This affects the Tag and Search Center pages. To use your Tag Center and Search Center pages, you must manually update the theme for the pages to the Version 8.5 theme. You must also update the profile of the pages to the Search and Tag Center profile.

About this task

The Search and Tag Center profile is a hidden profile. Therefore, you must set this profile in the page properties.

Procedure

1. Update the theme and profile that is used for the Tag Center pages:
   a. Edit the existing page that is using the older theme.
   b. Go to **Page properties**.
   c. Set the Portal 8.5 theme for the page.
   d. Set the **resourceaggregation.profile** parameter to `profiles/profile_search_tag.json`.
   e. Save your changes.
2. If you created the Search Center pages using the Portal 8.5 theme, update the profile that is used for the Search Center pages:
   a. Go to **Page properties**.
   b. Set the **resourceaggregation.profile** parameter to `profiles/profile_search_tag.json`.
   c. Save your changes.

Importing search web collections:

As a part of preparing your source environment, you exported web collections. After you export a search web collection from a source portal, you can import the data into a new, empty collection on the target portal. Importing a web collection retains most of the configuration data such as content sources, schedulers, filters, and language settings. If you configured such settings when creating the new collection, they are overwritten by the imported settings.

Before you begin

When you import a web collection, a background process fetches, crawls, and indexes all documents that are listed by URL in the previously exported file. Therefore, be aware of the memory and time that is required for crawls. For more information, see *Hints and tips for using Portal search*.

About this task

Complete the following steps on the target portal, using the Manage Search portlet:

Procedure

1. If you are migrating to a remote server, copy the XML file that contains the exported web collection to the server where the new version of WebSphere Portal is installed.
2. In the **Search Collections** box, click **New Collection** to create an empty collection.
3. Specify the required information in the **Location of Collection**, **Name of Collection**, and **Description of Collection** fields, and then click **OK**.
4. To include the security information when you import the search collection, add the **WS_KEY** parameter to the search service that contains the target search collection. Complete the following steps:
   a. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration** > **Manage Search**.
   b. Click **Search Services**.
   c. Click the **Edit** icon for the search service that contains the target search collection.
   d. In the **Parameter key** field, enter `WS_KEY`.

e. In the **New parameter value** field, enter `secret`.
f. Click **Add Parameter**.

Note: If you do not import the security information when you import a search collection, you must manually add the user name and password to each content source after you import the search collection into the target portal.

5. Click the **Import or Export Collection** icon for the collection that you created.
6. In the **Specify Location** field, enter the full directory path and XML file name of the file that you exported.
7. Click **Import**.
8. Edit the Content Source Configuration details to ensure that the settings match your target environment:
   a. Select the imported search collection to edit.
   b. Click the **Edit Content Source** icon.
   c. On the General Parameters tab, update the **Collect documents linked from this URL** field to match your target environment. The update might require changing the name of the target server and the port information. When you import a collection from a version of WebSphere Portal Express before 8.0, you need to add the context root path of the portal application. For example, adding `/wps` to the URL following the port information.
   d. On the Security tab, update the **Security Realms** to match your target environment.
   e. Click **Save**.
9. Click the **Search and Browse the Collection** icon for the collection that you created.
10. Verify that the documents found are the same as the collection that you created on the earlier portal server, and that the links work as expected. Any inconsistencies between the exported document count and the imported document count should be resolved the next time the cleanup daemon runs.

Related tasks:

"Exporting search web collections" on page 823
Use the Manage Search portlet to export search web collections from a source portal. Before you export a collection, make sure that the user who is running the portal application process has write access to the target directory location.

Related reference:

"Hints and tips for using Portal Search" on page 714
View some useful tips for using Portal Search.

Importing UX Screen Flow Manager dialog definitions:

If you migrated from Version 8.0.0.1 with the UX Screen Flow Manager (UXFM) enabled, then you exported and removed your dialog definitions before migrating to Version 8.5. Run the following task to import the dialog definitions into your upgraded system.

Procedure

1. Save the following sample of code as *ImportSampleCode.xml* to use when importing your dialog definitions.

```
<?xml version="1.0" encoding="UTF-8"?>
<request type="update" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="PCM_1.0.xsd">
```

```
    <!-- import all dialog definitions that are currently deployed -->
    <portal action="import">
        <dialog-set>
            <dialog name="*" />
        </dialog-set>
    </portal>
</request>
```

2. Run the following task to import your dialog definitions:
   - Linux : `./xmlaccess.sh -user` *`userID`* `-password` *`password`* `-url` *`url`* `-in` *`ImportSampleCode.xml`* `-in import.xml`
   - IBM i: `xmlaccess.sh -user` *`userID`* `-password` *`password`* `-url` *`url`* `-in` *`ImportSampleCode.xml`* `-in import.xml`
   - Windows: `xmlaccess.bat -user` *`userID`* `-password` *`password`* `-url` *`url`* `-in` *`ImportSampleCode.xml`* `-in` *`ImportSampleCode.xml`* `-in import.xml`

Related tasks:

"Exporting UXFM dialog definitions" on page 826
Export and save your dialog definitions before you migrate to Version 8.5.

WSRP:

If you use your IBM WebSphere Portal Express as a WSRP Consumer or Producer, you must complete additional steps to complete migration.

"Updating a WSRP Producer"
After you complete the migration of your data, perform the following task if you use IBM WebSphere Portal Express as a WSRP Producer.

"Updating a WSRP Consumer" on page 886
After you complete the migration of your data, perform the following tasks if you use IBM WebSphere Portal Express as a WSRP Consumer.

Updating a WSRP Producer:

After you complete the migration of your data, perform the following task if you use IBM WebSphere Portal Express as a WSRP Producer.

Procedure

The WSRP web service security configuration of a Producer is not automatically migrated. After you complete the migration of a Producer portal, you must manually configure the WSRP Producer for security. For more information, see *Securing a WSRP Producer portal*.

Note: WebSphere Portal Express does not provide a default option for the configuration of web service security that uses signed user name tokens anymore. If you have used this security configuration on earlier versions of WebSphere Portal Express, see *Security for WSRP services* for available security options for WSRP.

Related concepts:

"Security for WSRP services" on page 1441
IBM WebSphere Portal Express supports two security mechanisms for WSRP.

Related tasks:

"Securing a WSRP Producer portal" on page 1447
To secure provided portlets, you can configure the WSRP Producer for web service message security, for example, for message authentication. If you configure

message authentication, you must also configure Portal Access Control.

Updating a WSRP Consumer:

After you complete the migration of your data, perform the following tasks if you use IBM WebSphere Portal Express as a WSRP Consumer.

Procedure

- Verify the web service security configuration for the WSRP ports of the Producer definitions. For more information about configuring and securing WSRP on a Consumer portal, see *Configuring security on the Consumer portal*.
- If you configured web services security using LTPA version 1 tokens for a WSRP port of a Producer definition, you must complete more steps. WebSphere Application Server Version 8.5.5 does not support creating LTPA version 1 tokens for immediate use. Your options depend on whether the Producer requires LTPA version 1 tokens or is able to accept LTPA version 2 tokens. For information about your options and the required configuration changes, see *Configuring WSRP Producer ports for Web Service Security on the Consumer portal*.

  Note: WebSphere Portal Express does not provide a default configuration option for web service security that uses signed user name tokens anymore. If you used this security configuration on earlier versions of WebSphere Portal Express, see *Configuring WSRP Producer ports for Web Service Security on the Consumer portal* for available security options for WSRP.

  Note: WebSphere Portal Express does not support to secure WSRP by Secure Socket Layer (SSL) with Client Certificate Authentication anymore. If you used this security configuration on earlier versions of WebSphere Portal Express, see *Security for WSRP services*.
- When you complete the migration of a Producer, verify the corresponding Producer definition on the WSRP Consumer. For details about working with Producer definitions, see *Working with Producer definitions*.

  Note: In particular, verify the WSDL URL of the Producer and the web service security configuration and the endpoint URLs of the Producer ports. If the context root of the Producer was changed, adapt the endpoint URLs of the WSRP ports.
- Configure the JVM setting for retrieving multiple transport headers from a JAX-WS web service response, as described in description of APAR PM91361. See *PM91361: MULTIPLE SET-COOKIE VALUES FROM AN HTTP TRANSPORT HEADER IN A JAX-WS WEB SERVICE RESPONSE ARE NOT RETRIEVED CORRECTLY*.

Related concepts:

"Security for WSRP services" on page 1441
IBM WebSphere Portal Express supports two security mechanisms for WSRP.

Related tasks:

"Configuring security on the Consumer portal" on page 1461
You can configure security for the WSRP Consumer. If you enable security, the WSRP Consumer sends a security token as part of the WSRP request message to the WSRP producer. The security token represents the identity of the user who is logged in to the Consumer Portal. The WSRP Producer uses the security token to process the WSRP requests under the user identity that is represented by the security token.

"Configuring WSRP Producer ports for Web Service Security on the Consumer portal" on page 1464
You can configure each WSRP port of a particular Producer definition for web service security by using LTPA or username tokens.

"Working with Producer definitions" on page 1473
To make a WSRP Producer known to your Consumer portal, you create a Producer definition for that WSRP Producer in your Consumer portal. You can also configure the Producer definition.

Related information:

 PM91361: MULTIPLE SET-COOKIE VALUES FROM AN HTTP TRANSPORT HEADER IN A JAX-WS WEB SERVICE RESPONSE ARE NOT RETRIEVED CORRECTLY

Blogs and wikis:

After you migrate from WebSphere Portal Express Versions 7.0 or 8.0 to Version 8.5, you must run a configuration task to update the presentation templates that are used by blogs and wikis to apply the latest updates. You must run this task for content in your blogs and wikis to render properly.

About this task

If you customized your blogs and wiki, you will lose your customizations and you must reapply those customizations after you run the task.

Procedure

1. Go to the *wp_profile_root*`/ConfigEngine` directory.
2. Run the following command:
   - Linux : `./ConfigEngine.sh configure-blog -DPortalAdminPwd=`*`password`* `-DWasPassword=`*`password`*
   - IBM i: `ConfigEngine.sh configure-blog -DPortalAdminPwd=`*`password`* `-DWasPassword=`*`password`*
   - Windows: `ConfigEngine.bat configure-blog -DPortalAdminPwd=`*`password`* `-DWasPassword=`*`password`*
3. Restart WebSphere Portal Express.

Results

Enabling blogs and wikis by running the **`configure-blog`** task also enables tag and rating widgets. Go to "Enabling the new tag and rating widgets after a portal upgrade" on page 917 to verify and finish enabling tag and rating widgets.

Related reference:

"The tag and rating widgets" on page 1310
The portal provides one widget each for tags and for ratings.

Tag and Search Center pages:

When you migrate to IBM WebSphere Portal Express Version 8.5, the migration process does not apply the Portal 8.5 theme to all portal pages. For example, this affects the Tag and Search Center pages. To continue to use your Tag Center and Search Center pages, you must update the theme for the pages to the Version 8.5 theme. You must also update the profile of the pages to the Search and Tag Center profile.

Before you begin

Ensure that the Portal 8.5 theme is set for your Tag and Search Center pages before completing the following procedure.

About this task

The Search and Tag Center profile is a hidden profile. Therefore, you must set this profile in the page properties.

Procedure

1. To create the Tag Center pages, or apply the correct theme in order to work with these pages, run the following configuration task. Go to the *wp_profile_root*/ConfigEngine directory and enter:
   - HP-UX Linux : `./ConfigEngine.sh cp-setup-tag-center`
   - IBM i: `ConfigEngine.sh cp-setup-tag-center`
   - Windows: `ConfigEngine.bat cp-setup-tag-center`
2. To update the profile for the Search Center pages, complete the following steps:
   a. Go to Page properties.
   b. Set the **resourceaggregation.profile** parameter to `profiles/profile_search_tag.json`.

Removing Person Tag hidden pages:

Remove Person Tag hidden pages from migrated environments where Search for Portal Site was previously configured. If you migrated or updated to WebSphere Portal Express 8.5 CF04 or later, then you do not need to complete these steps.

About this task

You might find errors that are captured in the `systemOut.log` file if the migrated system has Search for Portal Site configured and crawling runs. The following excerpt is provided as an example.

```
00000052 PortletContai E com.ibm.wps.pe.pc.legacy.PortletContainerImpl
         performBeginEvents EJPPG1122E:

         An error occurred during portlet event processing.
             javax.portlet.PortletException: javax.servlet.UnavailableException: SRVE0200E:

         Servlet [com.ibm.wkplc.people.portal.portlet.dynamicpersontag.DynamicPersonTagPortlet]: Could not
         find required class -  class java.lang.ClassNotFoundException: com.ibm.wkplc.people.portal.portlet.
         dynamicpersontag.DynamicPersonTagPortlet

         at com.ibm.wps.pe.pc.legacy.PortletContainerImpl.callPortletMethod(PortletContainerImpl.java:1308)
```

Procedure

Remove the hidden pages to avoid these errors in the `systemOut.log`.

1. Log in to Portal as an Administrator.
2. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
3. Then, click **Context Root** > **Hidden Pages**.
4. Look for **Person Tag** with unique name **ibm.portal.Person.Tag**.
5. Delete this page.

Updating URL mapping for personalization page:

If you migrated from Version 7.0 to Version 8.5 with a CF03 or earlier fix pack applied, then you must complete the following task to ensure that the Personalization page displays properly.

Procedure

1. Log in to IBM WebSphere Portal Express.
2. Click **Administration**. Then, click **Portal Settings** > **URL Mapping**.
3. Delete the URL Mapping for **Personalization**.

   Note: Do not delete the **PZN** entry for PZN mapping on the list.

Enabling vanity URL support after migration:

In a new WebSphere Portal Express Version 8.5 installation, vanity URL support is enabled. However, if you upgrade your WebSphere Portal Express from a previous version to Version 8.5, vanity URL support is disabled. You can enable and disable vanity URL support as required by using a portal configuration task.

About this task

If vanity URL support is not enabled and a user tries to access a vanity URL, the portal gives a 404 return code.

Note: For vanity URLs to work, managed pages also must be enabled as a prerequisite. For more information, read *Enabling managed pages after migration*. If you disable managed pages, this step also disables vanity URLs. If you do not disable vanity URLs and you enable managed pages again, vanity URLs also work again.

Enabling vanity URL support

To enable vanity URL support, you use the configuration task **enable-vanityurl-support**. This task sets a new custom property in the Resource Environment Provider of the WP Configuration Service. The property name is **vanityurl.support.enabled**. When you run the configuration task, the property is set to the value `true`.

Note: It is mandatory to enable managed pages to have vanity URLs enabled. If managed pages support is not enabled, the task **enable-vanityurl-support** fails. In this case, run the task **enable-managed-pages** first.

Disabling vanity URL support

To disable vanity URL support, you use the configuration task **disable-vanityurl-support**. This task deletes the custom property **vanityurl.support.enabled** from the Resource Environment Provider of the WP Configuration Service. If the portal then receives a request to serve a vanity URL, it gives a 404 return code.

Syntax

You call these configuration tasks as follows:

IBM i

Enable: `ConfigEngine.sh enable-vanityurl-support -DPortalAdminPwd=`*`password`* `-DWasPassword=`*`password`*

Disable: `ConfigEngine.sh disable-vanityurl-support -DPortalAdminPwd=password -DWasPassword=password`

Linux

Enable: `./ConfigEngine.sh enable-vanityurl-support -DPortalAdminPwd=password -DWasPassword=password`

Disable: `./ConfigEngine.sh disable-vanityurl-support -DPortalAdminPwd=password -DWasPassword=password`

Windows

Enable: `ConfigEngine.bat enable-vanityurl-support -DPortalAdminPwd=password -DWasPassword=password`

Disable: `ConfigEngine.bat disable-vanityurl-support -DPortalAdminPwd=password -DWasPassword=password`

Remember: After you run the **enable-vanityurl-support** or **disable-vanityurl-support** task, restart the server. Go to "Starting and stopping servers, deployment managers, and node agents" on page 1216 for specific instructions.

Related tasks:

"Enabling managed pages" on page 913
After migration, you must manually enable support for managed pages. Without managed pages support, some features like the project menu are not available on the migrated server.

Enabling impersonation:

If you migrate to WebSphere Portal Express Version 8.5 and upgrade to fix pack level CF01-CF03, impersonation might not be enabled. If you plan to use the impersonation feature, which allows a selected user to preview and test new pages or portlets to help identify any potential issues, then you might need to enable this feature.

About this task

Note: If you applied CF04 or a later fix pack, then you do not need to complete the steps to enable impersonation.

Portal Access Control (PAC) controls the ability to impersonate another user. To impersonate another user, the **Can Run As User** role on the virtual resource users must be assigned. You must first enable the impersonation feature within WebSphere Portal Express. If you are unsure whether the impersonation feature is enabled, you can use the following instructions to verify that you have the correct settings.

Perform the following steps to enable the impersonation feature:

Procedure

1. Log on to the WebSphere Application Server Integrated Solutions Console or Network Deployment Administration Console.
2. Perform the following steps to enable the impersonation feature:
   a. Go to **Resources** > **Resource Environment** > **Resource environment Providers** > **WP Authentication Service** > **Custom Properties**.
   b. Click **New**.
   c. Enter `logout.explicit.filterchain` in the **Name** field.

d. Enter `com.ibm.wps.auth.impersonation.impl.ImpersonationLogoutFilter` in the **Value** field.
e. Click **Apply** , and then click **Save** to save the changes directly to the master configuration.
f. Go to **Resources** > **Resource Environment** > **Resource Environment Providers** > **WP PortletServiceRegistryService** > **Custom Properties**.
g. Click **New**.
h. Enter `com.ibm.wps.portletservice.impersonation.impl.ImpersonationServiceImpl` in the **Name** field.
i. Enter `com.ibm.wps.portletservice.impersonation.impl.ImpersonationServiceImpl` in the **Value** field.
j. Click **Apply** and then click **Save** to save the changes directly to the master configuration.

3. Stop and restart the WebSphere_Portal server.
4. Perform the following steps to assign the **Can Run As User** role to a user.
   a. Log on to WebSphere Portal Express as the administrator.
   b. Click **Administration** in the site toolbar.
   c. Click **Access** > **User and Group Permissions**.
   d. Click **Users**.
   e. Search for the user to which you want to assign the **Can Run As User** role.
   f. Click the **Select Resource Type** icon for the user.
   g. Click **Page Next**, and go to the page that contains the **Virtual Resources** option, and click that link.
   h. Go to the page that contains the **Users** option, and click the **Assign Access** icon.
   i. Select the **Explicitly Assign** check box for the **Can Run As User** role.
   j. Click **OK**.
   k. Verify that the user now has **User** and **Can Run As User** access.

   The users with the **Can Run As User** role can now impersonate another user.

Reinstalling the PortalTheme:

If you migrate from Version 7.0 to Version 8.5, when you apply the latest combined cumulative fix during post-migration, the PortalTheme from earlier WebSphere Portal Express versions is removed. If your system requires this theme, then you can manually reinstall it.

About this task

In Version 8.0, the PortalTheme was removed. Complete the following procedure only if your migrated system requires the PortalTheme from an earlier version.

Procedure

1. Run the following command to install the PortalTheme.
   - Linux : `./ConfigEngine.sh deploy-portal-theme -DPortalAdminPwd=`*`password`* `-DWasPassword=`*`password`*
   - IBM i: `ConfigEngine.sh deploy-portal-theme -DPortalAdminPwd=`*`password`* `-DWasPassword=`*`password`*

- Windows: `ConfigEngine.bat deploy-portal-theme -DPortalAdminPwd=password -DWasPassword=password`

2. Restart your portal server.

Development tasks

Complete the development tasks that are required after the migration to ensure that your environment functions properly.

"Updating portlets URL"
After migrating your portal environment, you must verify and update as needed the portlets URLs. In some cases, portlets have the incorrect URL after migration. URLs can be easily corrected using the XML configuration interface to export, update, and import the configurations.

"Updating custom theme Dojo references" on page 893
The default Dojo context root in WebSphere Portal Express is `/WpsContextRoot/portal_dojo`. You can find the value of `WpsContextRoot` in *wp_profile_root*`/ConfigEngine/properties/wkplc.properties`.

"Enabling and upgrading JavaServer Faces portlet applications" on page 894
You can enable or upgrade portlet projects that use JavaServer Faces 1.x to JavaServer Faces 2.0 Facelet-based portlet projects. Refer to the Rational Application Developer for WebSphere Software documentation for instructions on enabling and upgrading JavaServer Faces portlet applications.

Updating portlets URL:

After migrating your portal environment, you must verify and update as needed the portlets URLs. In some cases, portlets have the incorrect URL after migration. URLs can be easily corrected using the XML configuration interface to export, update, and import the configurations.

About this task

Perform the following steps in the newly migrated environment.

Procedure

Export the portlet configuration.

1. Change directory where the WebSphere Portal Express tools are contained:
   - IBM i Linux : *PortalServer_root*`/bin`
   - Windows: *PortalServer_root*`\bin`
2. Export the configuration using the provided sample file that is named `ExportAllPortlets.xml` by entering the following command:
   - Linux : `./xmlaccess.sh -in` *PortalServer_root*`/doc/xml-samples/ExportAllPortlets.xml -user` *wpsadmin* `-password` *wpsadminpwd* `-url http://`*server.example.com:port*`/wps/config -out` *Server_config.xml*
   - IBM i: `xmlaccess.sh -in` *PortalServer_root*`/doc/xml-samples/ExportAllPortlets.xml -user` *wpsadmin* `-password` *wpsadminpwd* `-url http://`*server.example.com:port*`/wps/config -out` *Server_config.xml*
   - Windows: `xmlaccess.bat -in` *PortalServer_root*`\doc\xml-samples\ExportAllPortlets.xml -user` *wpsadmin* `-password` *wpsadminpwd* `-url http://`*server.example.com:port*`/wps/config -out` *Server_config.xml*

   The exported configuration is stored in the XML file named `Server_config.xml`.

Verify output XML file and update URLs as needed.

3. Open the Server_config.xml file and verify that all URLs are valid. This means that all the files that are listed in the XML file must be present and accessible in the specified locations.
   a. If the URLs do not match the actual location in your environment, update those entries according to the actual location of the files.
   b. If the files specified in the URL field do not exist in the migrated environment or are no longer needed in the configuration, remove those entries.

   Note: If you are making the updates on a clustered environment, modify the XML file to remove the global-settings and services-settings entries if available in the file.

Import XML file with the updated URLs.

4. Change directory where the WebSphere Portal Express tools are contained:
   - IBM i Linux : *PortalServer_root*/bin
   - Windows: *PortalServer_root*\bin
5. Run the following command to import the updated configuration using the Server_config.xml file that you just updated:
   - Linux : ./xmlaccess.sh -in Server_config.xml -user *wpsadmin* -password *wpsadminpwd* -url http://*Server.example.com:port*/wps/config
   - IBM i: xmlaccess.sh -in Server_config.xml -user *wpsadmin* -password *wpsadminpwd* -url http://*Server.example.com:port*/wps/config
   - Windows: xmlaccess.bat -in *Server_config.xml* -user *wpsadmin* -password *wpsadminpwd* -url http://*Server.example.com:port*/wps/config
6. After the request is processed, make sure that the import process displays the following return message:

   ```
   <status element="all" result="ok">
   ```

For clustered environments only.

7. Resync the cluster from the Deployment Manager console.
8. Restart the Enterprise Applications from the Deployment Manager console.
9. Run the following command on the cluster primary node to activate the deployed portlets:
   - Linux : ./ConfigEngine.sh activate-portlets -DWasPassword=*password*
   - IBM i: From the UserData directory, ConfigEngine.sh activate-portlets -DWasPassword=*password*
   - Windows: ConfigEngine.bat activate-portlets -DWasPassword=*password*
10. Restart the portal.

Updating custom theme Dojo references:

The default Dojo context root in WebSphere Portal Express is /WpsContextRoot/portal_dojo. You can find the value of WpsContextRoot in *wp_profile_root*/ConfigEngine/properties/wkplc.properties.

About this task

You might find that migrated themes, including custom themes, have references to /portal_dojo without the WpsContextRoot prefix. You can look for these references in both the WAR file and in the WebDAV storage for the theme, and update it if needed.

See the following procedure for an example of how to search for these references in a Linux environment.

Procedure

1. Open a command line on your server.
2. Enter `WebSphere Portal Install Directory> grep -Hr "\"/portal_dojo".`

Related tasks:

"Using WebDAV with WebSphere Portal Express" on page 1351
WebSphere Portal Express provides a Web-based Distributed Authoring and Versioning (WebDAV) implementation that individual services can use by plugging into. WebDAV is a set of extensions to the HTTP protocol that allows you to collaborate on editing and managing files on remote Web servers. Caching Proxy supports WebDAV methods used by Microsoft Exchange Server, and user-defined (customized) methods. These methods are hard coded and managed by the Enable and Disable directives. Administrators can also use the corresponding method-mask defined in the PROTECT directive to authorize the use of these methods.

Enabling and upgrading JavaServer Faces portlet applications:

You can enable or upgrade portlet projects that use JavaServer Faces 1.x to JavaServer Faces 2.0 Facelet-based portlet projects. Refer to the Rational Application Developer for WebSphere Software documentation for instructions on enabling and upgrading JavaServer Faces portlet applications.

Go to Enabling and upgrading JavaServer Faces portlet applications for step-by-step instructions.

Database tasks

There are different post-migration tasks that are required depending on the type of database that is used.

"Updating the isolation level for SQL server databases"
If you are using an SQL Server, you must update the isolation level of your release, community, customization, and JCR databases to improve concurrency and efficiency of the databases.

"Updating DB2 self-tuning memory manager (STMM) settings" on page 895
With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

"Oracle: Enabling the auto space advisor background task" on page 896
If you use an Oracle database, you must complete a post-migration task to enable the Oracle background task called "Auto Space Advisor" after your run the **upgrade-profile** task during migration using the Configuration Wizard.

Updating the isolation level for SQL server databases:

If you are using an SQL Server, you must update the isolation level of your release, community, customization, and JCR databases to improve concurrency and efficiency of the databases.

Before you begin

Ensure that the database is not in use.

Procedure

Run the command: `ALTER DATABASE database_name SET READ_COMMITTED_SNAPSHOT ON;`.

Updating DB2 self-tuning memory manager (STMM) settings:

With the Version 8.5 release, a number of the settings have been changed to be managed by the self-tuning memory manager (STMM) engine. Change your settings to the recommended values, if there are no specific needs for the current values.

Procedure

1. Stop the portal server.
2. Change your settings to the recommended values:

   UPDATE DBM CFG USING sheapthres 0;

   For each Portal database (release, community, customization, jcr, feedback, likeminds):

   - UPDATE DB CFG FOR *dbname* USING applheapsz automatic;
   - UPDATE DB CFG FOR *dbname* USING stmtheap automatic;
   - UPDATE DB CFG FOR *dbname* USING dbheap automatic;
   - UPDATE DB CFG FOR *dbname* USING locklist automatic;
   - UPDATE DB CFG FOR *dbname* USING avg_appls automatic;
   - UPDATE DB CFG FOR *dbname* USING PCKCACHESZ automatic;
   - UPDATE DB CFG FOR *dbname* USING AUTO_MAINT on;
   - UPDATE DB CFG FOR *dbname* USING SHEAPTHRES_SHR automatic;
   - UPDATE DB CFG FOR *dbname* USING SORTHEAP automatic;
   - UPDATE DB CFG FOR *dbname* USING SELF_TUNING_MEM ON;
   - Required only for the database that contains the JCR Domain: UPDATE DB CFG FOR *dbname* USING logfilsiz 16000;
   - Required only for the database that contains the JCR Domain: UPDATE DB CFG FOR *dbname* USING logprimary 20;
   - Required only for the database that contains the JCR Domain: UPDATE DB CFG FOR *dbname* USING logsecond 50;
   - Required only for the database that contains the JCR Domain: UPDATE DB CFG FOR *dbname* USING logbufsz 500;
3. Connect to your database.

   `db2 connect to dbdomain.DbName user dbdomain.DBA.DbUser using dbdomain.DBA.DbPassword`

   Note:

   - For *dbdomain.DbName*, enter the name of the portal domain database.
   - For *dbdomain.DBA.DbUser*, enter the name of the database administrator user ID for privileged access operations during database creation and setup.
   - For *dbdomain.DBA.DbPassword*, enter the database administrator password for privileged access operations during database creation.
4. Change your settings to the recommended bufferpool values:
   - ALTER BUFFERPOOL ICMLSFREQBP4 SIZE automatic;
   - ALTER BUFFERPOOL ICMLSVOLATILEBP4 SIZE automatic;

- ALTER BUFFERPOOL ICMLSMAINBP32 SIZE automatic;
- ALTER BUFFERPOOL CMBMAIN4 SIZE automatic;

Oracle: Enabling the auto space advisor background task:

If you use an Oracle database, you must complete a post-migration task to enable the Oracle background task called "Auto Space Advisor" after your run the **upgrade-profile** task during migration using the Configuration Wizard.

Procedure

Enter the following SQL:

```
BEGIN
 dbms_auto_task_admin.enable(
  client_name => 'auto space advisor',
  operation  => NULL,
  window_name => NULL);
END;
/
```

What to do next

To ensure that you enabled the "Auto Space Advisor" task, you can check whether the task is enabled or disabled by entering the following command from SQL Plus:

```
SQL> select status from dba_autotask_client where client_name = 'auto space advisor';
```

Related tasks:

"Oracle: Disabling the auto space advisor background task" on page 839
To prevent deadlocks during migration, you must complete a task to disable the Oracle background task called "Auto Space Advisor" before your run the **upgrade-profile** task during migration using the Configuration Wizard. After you complete migration, you can enable "Auto Space Advisor" as a post-migration task.

Add-ons, features, and third-party integration tasks

Complete the post-migration tasks that are required based on the way you use your WebSphere Portal Express environment.

"Content Template Catalog" on page 897
Content Template Catalog versions 3.x, 4.0.x, and 4.1.x are migrated along with all Web Content Manager data. These are additional migration steps that are required for Content Template Catalog after data migration is complete.

"Migrating Security Access Manager" on page 899
The IBM WebSphere Portal Express migration process migrates the security configurations. However, there is no provision for the automatic migration of any junction definitions that exist for the previous version of WebSphere Portal Express in WebSEAL. You must replace the old junction definitions with the new virtual host junction definitions.

"Migrating your PAA content" on page 901
If you installed a Portal Application Archive (PAA) file on a previous version of IBM WebSphere Portal Express, you can migrate the content to the current version of WebSphere Portal Express.

"Creating the analytics tag root label " on page 902
If you use site analytics on your migrated WebSphere Portal Express, you need to create the analytics tag root label.

Content Template Catalog:

Content Template Catalog versions 3.x, 4.0.x, and 4.1.x are migrated along with all Web Content Manager data. These are additional migration steps that are required for Content Template Catalog after data migration is complete.

Important post migration steps for Content Template Catalog versions 3.x, 4.0.x, and 4.1.x

As part of your migration to WebSphere Portal version 8.5 you must do all the migration steps and post-migration steps.

Using the WebSphere Portal version 8.5 default theme

If you want to use the new WebSphere Portal version 8.5 default theme, including all the new features that are included with that theme, such as the updated toolbar, you need to download and install Content Template Catalog version 4.2 or higher. Follow the upgrade procedures in the Content Template Catalog version 4.2 documentation.

Upgrading the Content Template Catalog 3.x theme to IBM WebSphere Portal 8.5

If you upgraded from IBM WebSphere Portal Express 7 to IBM WebSphere Portal Express 8.5, content and pages created with Content Template Catalog 3.x are migrated to WebSphere Portal Express 8.5 automatically.

The version of Content Template Catalog 3.x that ran on WebSphere Portal Express 7 was based on the PageBuilder2 theme. To run on WebSphere Portal Express 8.5, install the Mashup Integration `.ear` file, which installs additional required modules. Then, use the XML Access script that is provided to configure the Content Template Catalog 3.x theme on the WebSphere Portal Express 8.5 server.

1. On the WebSphere Portal Express 8.5 server, install the Mashup Integration `.ear` file by running the ConfigEngine batch or script file with the following arguments:

   Windows

   ```
   ConfigEngine.bat action-create-ear-wp.mmi.deploy
   ```

   AIX Linux

   ```
   ./ConfigEngine.sh action-create-ear-wp.mmi.deploy
   ```

2. Copy the following XML Access script and save it as an XML file:

```
<?xml version="1.0" encoding="UTF-8"?>
 <request type="update" version="8.5.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd">
   <portal action="locate">
    <theme action="update" active="true" context-root="/wps/PageBuilder2" default="false"
     domain="rel" resourceroot="PageBuilder2" uniquename="CTCTheme"/>
   </portal>
 </request>
```

3. Log in to WebSphere Portal Express as an administrator.
4. Click the **Administration menu** icon. Then, click **Portal Settings** > **Import XML**.
5. Import the file that is saved in Step 2.

Enabling inline editing for Content Template Catalog version 3.x

Inline editing for Content Template Catalog version 3.x is not enabled during migration. To enable inline editing:

1. Open the authoring portlet and go to the Content Template Catalog version 3.x Design library.
2. Open the Inline Editing folder and edit all the toolbar components.
3. Change the "On Editing" setting to "Edit live content".
4. Save each component.

Tagging and rating

If you use tagging and rating, then you must enable the new tag and rating widgets after migration. For more information, see "Enabling the new tag and rating widgets after a portal upgrade" on page 917.

Migrating Security Access Manager:

The IBM WebSphere Portal Express migration process migrates the security configurations. However, there is no provision for the automatic migration of any junction definitions that exist for the previous version of WebSphere Portal Express in WebSEAL. You must replace the old junction definitions with the new virtual host junction definitions.

About this task

The migration process migrates security configurations such as the user registry, VMM settings, and the IBM WebSphere Application Server security setup, including any Trust Association Interceptor (TAI) configurations. You must install the latest version of TAI on the current version of WebSphere Portal Express server. This installation configures the new WebSphere Portal Express instance for integration with Security Access Manager WebSEAL.

The WebSphere Portal Express migration process cannot change the junction definitions in WebSEAL to point to the new server. It cannot switch from standard non-transparent or transparent junctions to the new virtual host junctions. You must manually run these tasks within Security Access Manager. The Security Access Manager administrator staff often runs these tasks, which might be separate from the WebSphere Portal Express administrative staff.

Tip: Complete the following steps with the instructions in the *Security Access Manager eBusiness WebSEAL Administrative Guide*.

Attention: These steps describe that you create the new virtual host junction before you delete the old junctions. This approach assumes that there are no detected conflicts to prevent the new junction from coexisting with the old junctions. A conflict might arise if the **vhost_label** value is the same between the new and old junctions. Try to avoid or work around these conflicts. If you cannot avoid the conflict, delete the old junction before you create the new virtual host junction. Create a backup copy of the WebSEAL configuration file first so you can refer to it if necessary.

Procedure

1. Complete the following steps on the previous instance of WebSphere Portal Express:
   a. Open the WebSEAL configuration file.
   b. Search the file for stanzas that define the junctions. For example: `[junction:junction_name]`.
   c. Record the configuration value for each junction for future reference.
   d. Save a backup copy of the WebSEAL configuration file.
2. Create the new virtual host junctions that are based on the junctions from the previous instance:

   The general format for the `pdadmin` command to create a virtual host junction is

   ```
   pdadmin> server task WebSEAL-instance_name-webseald-WebSEAL-HostName virtualhost create -t typ
   ```

The following information describes the mandatory parameters in the `pdadmin` command:

- The *WebSEAL-instance_name*-webseald-*WebSEAL-HostName* has three parts, as documented in the WebSEAL Administration Guide:
  a. The configured name of a single WebSEAL instance, for example *web 1*
  b. The literal string `-websealed-`
  c. The host name, for example, *webseal.yourco.com*

  The resulting combination would be *web 1*-websealed-*webseal.yourco.com*. You can use the `pdadmin server list` command to display the correct format of the server name.
- The virtual host label (`vhost-label`) is the name for the virtual host junction.
  – Virtual host junctions are always mounted at the root of the WebSEAL object space.
  – You can refer to a junction in the `pdadmin` utility with this label.
  – The virtual host junction label must be unique within each instance of WebSEAL.
  – Because the label represents virtual host junctions in the protected object space, the label name must not contain the forward slash character (/).
- **`-t type`**: This parameter defines whether the junction is encrypted (`-t ssl`) or not encrypted (`-t tcp`). This parameter is mandatory when you create a virtual host junction. For more information about other possible values, see the *WebSEAL Administration Guide*.
- **`-h hostname`**: This parameter defines the backend server to which the junction connects. In most situations, the host name is the HTTP server that sits in front of WebSphere Portal Express. This parameter is mandatory when you create a virtual host junction.

The *[options]* includes the following parameters:

- **`-p port`**: This parameter defines the port number for the backend server to which the junction connects. If not specified, the default value is `80` for HTTP or `443` for HTTPS. It is best to specify this value explicitly in the junction creation command even if the default values are in use.
- **`-v vhost_name[:port]`**: This parameter is the virtual host name and port number that defines the junction. WebSEAL maps incoming requests to this host name and port to this junction. If not specified, the values default to the **`-h hostname`** and **`-p port`** values.
- **`-c header_type`**: This parameter inserts the Security Access Manager client identity in HTTP headers across the junction. The **`header_type`** argument can include any combination of the following Security Access Manager HTTP header types:
  – `{iv_user|iv_user-l}`
  – `iv_groups`
  – `iv_creds`
  – `all`

  The header types must be comma-separated, and cannot have a space between the types. For example: **`-c iv_user,iv_groups`**. Specifying **`-c all`** is the same as specifying **`-c iv_user,iv_groups,iv_creds`**. This parameter is valid for all junctions except for the type of local. The setting here depends on how you want your TAI running within WebSphere Application Server to operate. In certain modes, the TAI might be looking for the presence of one or more of these headers. The TAI looks for these headers to know that it

must claim the request when interrogated by WebSphere Application Server security. This setting must be set to match what the TAI is looking for. Consult your WebSphere system administrator if you are in doubt as to how the TAI is configured.

- **-b**: This option controls how WebSEAL passes authentication information to the backend server. Usually this setting depends on how you want the TAI to be configured in WebSphere to validate a trust relationship with WebSEAL. The usual option that is chosen is **-b supply**. For more information, see the *WebSEAL Administration Guide* or the *ETAI installation and configuration* documentation.
- **-k**: This option controls whether WebSEAL includes its own session cookie in the request to the backend server. In some situations, sending the WebSEAL session cookie to the backend server is necessary. This action is necessary to support single sign-on from WebSphere Portal Express to other backend services where WebSEAL also protects those backend services.
-

  Note: Junctions to WebSphere Portal Express whether direct or through an HTTP server does not support the **-q option** the query_contents function. Query_contents is not possible on WebSphere Portal Express

The following information is a sample command to create a virtual host TCP junction, on the *web 1* WebSEAL instance that is running on a host *webseal.yourco.com*, for the virtual host name *portalvhost.yourco.com* running on port 80 that requires a TAI in WebSphere Application Server. The virtual host junction is labeled *vhost_junction_portal_1*. The virtual host junction host name must be mapped in DNS to the WebSEAL server. The portal or http server is running on host *portal.yourco.com* and is using port 8080:

```
pdadmin> server task web1-webseald-webseal.yourco.com virtualhost create -t tcp -v portalvhost
```

3. Delete the old junctions with the appropriate administration commands. For example: `server task` *`instance_name`* `-webseal-`*`host_name`* `delete` *`junction_point`*

Related tasks:

"Configuring Security Access Manager for authentication only" on page 1631
IBM WebSphere Portal Express and IBM WebSphere Application Server support the Trust Association Interceptors (TAI) that IBM Security Access Manager provides. If you use Security Access Manager for authorization, you must also use Security Access Manager for authentication. Using Security Access Manager only for authorization is not supported.

"Configuring Security Access Manager for authentication, authorization, and the Credential Vault" on page 1644
You can configure Security Access Manager for authentication, authorization, and the vault adapter with one task.

Migrating your PAA content:

If you installed a Portal Application Archive (PAA) file on a previous version of IBM WebSphere Portal Express, you can migrate the content to the current version of WebSphere Portal Express.

About this task

Migration is supported from WebSphere Portal Express Version 7.0 to 8.5 and from Version 8.0 to 8.5.

Procedure

1. If you are migrating from Version 7.0 to Version 8.5, copy the PAA content from the directory of the previous release to the directory in the current release.

   Note: If you are migrating from Version 8.0.0.1 to Version 8.5, you can skip this step.

   Tip: Starting with version 8.0, the PAA directory is in the *wp_profile_root*/paa directory.

   Remember: Select only the expanded PAA file directories that you want to migrate to the new release.
2. Open a command prompt and change to the following directory:
   - IBM i: *wp_profile_root*/ConfigEngine
   - Windows: *wp_profile_root*\ConfigEngine
3. Run the following task to migrate the PAA content:

   This task deletes the auto-generated code and then re-creates it in the config/includes directory for each component. The task then registers the assembly and components with the current version ConfigEngine.
   - IBM i: ConfigEngine.sh migrate-paa -DWasPassword=*password* -DPortalAdminPwd=*password*
   - Windows: ConfigEngine.bat migrate-paa -DWasPassword=*password* -DPortalAdminPwd=*password*

Creating the analytics tag root label :

If you use site analytics on your migrated WebSphere Portal Express, you need to create the analytics tag root label.

About this task

To do this, proceed as follows in the newly migrated environment:

Procedure

1. Change to the following directory that contains the WebSphere Portal Express tools:
   - Linux : *PortalServer_root*/bin
   - IBM i: *PortalServer_root*/bin
   - Windows: *PortalServer_root*\bin
2. Create the analytics tag root label by running the following command:
   - Linux :

     ```
     ./xmlaccess.sh
     -url http://example_server.com:port/wps/config
     -user wpsadmin -password wpsadminpwd
     -in PortalServer_root/base/wp.asa.server.impl/config/templates/create_asa_tag_root.xml
     -out results.xml
     ```
   - IBM i:

     ```
     ./xmlaccess.sh
     -url http://example_server.com:port/wps/config
     -user wpsadmin -password wpsadminpwd
     -in PortalServer_root/base/wp.asa.server.impl/config/templates/create_asa_tag_root.xml
     -out results.xml
     ```
   - Windows:

```
xmlaccess.bat
-url http://example_server.com:port\wps\config
-user wpsadmin -password wpsadminpwd
-in PortalServer_root\base\wp.asa.server.impl\config\templates\create_asa_tag_root.xml
-out results.xml
```

As an alternative, you can also use the portal administration portlet **Import XML** to perform this import.

3. After the request has been processed, make sure that the import process has returned the following message: `<status element="all" result="ok">`

Results

Your site analytics tags are now ready for you to work.

Updating the web application bridge:

After you migrated to Version 8.5 and successfully tested it, update your migrated web application bridge content.

Procedure

1. Log in to IBM WebSphere Portal Express Version 8.5.
2. Click **Administration**. Then, click **Portlet management** > **Virtual Web Application Manager**.
3. View the user interface that lists the previous virtual web applications.
4. Click **Show Virtual Web Application** to display the list of previous virtual web applications.
5. Select the one of the previous virtual web applications that you want to create in the current version.
6. Click **Apply**. The screen to create the content provider profile displays.
7. Create the content provider profile.
8. Click **Next**. The screen to create the content provider profile policy displays.
9. Create the content provider profile policy.
10. Repeat these steps until all of the previous virtual web applications are converted to the current version.
11. Create the **Web Dock Application**.

Running post migration steps for the multilingual solution:

When you migrate the multilingual solution, you need to merge your configuration settings.

About this task

Note: If your current multilingual system uses synchronized publishing, it is recommended to move to the project-based synchronizing publishing extension. If you have existing documents in the pending-publish stage, then you temporarily leave the pending publish stage in your workflow until those items are published. The presence of the pending publish stage does not affect new items that use projects.

Procedure

1. Ensure that the **`WasPassword`** and **`PortalAdminPwd`** passwords are set in the `wkplc.properties` file.

2. Run the following registration command from the *wp_profile_root*/ConfigEngine directory:

   Windows
   ```
   ConfigEngine.bat register-wcm-mls
   ```

   Linux
   ```
   ./ConfigEngine.sh register-wcm-mls
   ```

   IBM i
   ```
   ConfigEngine.sh register-wcm-mls
   ```

3. Run the following deployment command from the *wp_profile_root*/ConfigEngine directory:

   Windows
   ```
   ConfigEngine.bat deploy-wcm-mls
   ```

   Linux
   ```
   ./ConfigEngine.sh deploy-wcm-mls
   ```

   IBM i
   ```
   ConfigEngine.sh deploy-wcm-mls
   ```

4. If your server contains virtual portals, you must also run the following task for each virtual portal on your server:

   Windows
   ```
   ConfigEngine.bat import-wcm-mls-data
   -DVirtualPortalHostName=VirtualPortalHostName
   -DVirtualPortalContext=virtual_portal_context_url
   ```

   Linux
   ```
   ./ConfigEngine.sh import-wcm-mls-data
   -DVirtualPortalHostName=VirtualPortalHostName
   -DVirtualPortalContext=virtual_portal_context_url
   ```

   IBM i
   ```
   ConfigEngine.sh import-wcm-mls-data
   -DVirtualPortalHostName=VirtualPortalHostName
   -DVirtualPortalContext=virtual_portal_context_url
   ```

5. Restart WebSphere Portal.
6. Repeat these steps on every server and cluster node.

Social Media Publisher:

The Social Media Publisher for Web Content Manager is an extension to Web Content Manager that allows businesses to promote their web content on social networks, and provide some basic statistics about the promoted content. When migrating the Social Media Publisher, you need to merge your configuration settings.

About this task

Procedure

1. Ensure that the **WasPassword** and **PortalAdminPwd** passwords are set in the wkplc.properties file.
2. Edit *wp_profile_root*/PortalServer/wcm/social/smp.properties to ensure that the configuration is correct for the current server or cluster node. Specifically, AUTHOR_DB_URL, AUTHOR_DB_NAME, AUTHOR_DB_SCHEMA and AUTHOR_DB_TYPE.

   Note: The parameter settings in smp.properties are case-sensitive.
3. Run the following registration command from the *wp_profile_root*/ConfigEngine directory:

   Windows
   ```
   ConfigEngine.bat register-wcm-social
   ```

Linux `./ConfigEngine.sh register-wcm-social`

IBM i `ConfigEngine.sh register-wcm-social`

4. Run the following deployment command from the *wp_profile_root*/ConfigEngine directory:

   Windows
   `ConfigEngine.bat deploy-wcm-social`

   Linux `./ConfigEngine.sh deploy-wcm-social`

   IBM i `ConfigEngine.sh deploy-wcm-social`
5. Restart WebSphere Portal.
6. Repeat these steps on every server and cluster node.

Social Lists:

After you migrate from a previous WebSphere Portal Version to Version 8.5, you must run a configuration task. The configuration task is run to deploy the new web content library and templates before you can use the Social Lists features. If you already used the social rendering feature from 8.0.0.1, your existing web content libraries and all portlet clones that were created during the enablement on 8.0.0.1 is not changed.

Before you begin

If you migrated from 7.0 to 8.5, then you must run the **action-install-wcm-localrender-portlet** task before you start the following procedure:

- Linux : `./ConfigEngine.sh action-install-wcm-localrender-portlet -DPortalAdminPwd=`*password* `-DWasPassword=`*password*
- IBM i: `ConfigEngine.sh action-install-wcm-localrender-portlet -DPortalAdminPwd=`*password* `-DWasPassword=`*password*
- Windows: `ConfigEngine.bat action-install-wcm-localrender-portlet -DPortalAdminPwd=`*password* `-DWasPassword=`*password*

Procedure

1. Go to the wp_profile_root/ConfigEngine directory. Run the **deploy-social-rendering** task:
   - Linux : `./ConfigEngine.sh deploy-social-rendering -DPortalAdminPwd=`*password* `-DWasPassword=`*password*
   - IBM i: `ConfigEngine.sh deploy-social-rendering -DPortalAdminPwd=`*password* `-DWasPassword=`*password*
   - Windows: `ConfigEngine.bat deploy-social-rendering -DPortalAdminPwd=`*password* `-DWasPassword=`*password*

   By default, the task is run on the base portal. To run this task on a different virtual portal, identify the virtual portal by adding one of the following parameters to the command line. Each parameter requires the prefix `-D` on the command line

   VirtualPortalHostName
   Specify the host name of the virtual portal. For example, `vp.example.com`

   VirtualPortalContext
   Specify the virtual portal context that identifies the virtual portal. For example, `vp1`.

   Example Commands:

```
./ConfigEngine.sh deploy-social-rendering -DPortalAdminPwd=password
-DWasPassword=password -DVirtualPortalContext=vp1
```

Note: If you plan to continue to use the old toolbar, you can use the following configuration task to deploy new social portlets to that toolbar.
```
./ConfigEngine.sh action-deploy-social-shelf-category-and-portlets
-DPortalAdminPwd=password -DWasPassword=password
```

2. Verify that the web content library Social Lists 1.1 was created. Click the **Administration menu** icon. Then, click **Portal Content** > **Web Content Libraries**.

Related concepts:

"Enabling social rendering in a virtual portal" on page 2133
Before you use social rendering in a virtual portal, you must deploy the new web content library and templates. The version and fix pack of your IBM WebSphere Portal Express determines how you do so.

Deploying and updating sample web content template items:

The sample web content template items are not installed or updated during migration. This sample content includes examples of web content template pages and predefined content items that you can add to pages to render content. You can add or update these items manually after migration.

Before you begin

Important: These instructions apply only if you installed an offering that includes the sample web content template items. These steps do not work with an offering that does not include the sample content.

About this task

When you add the sample content, the following artifacts are created:

- The `Template Page Content 3.0` and `Web Content Templates 3.0` web content libraries.
- The **Web Content** category in the site toolbar.
- The "Image," "Rich Text," "List of Articles," and "Article" content items. These content items are available from **Create** > **Content** tab of the site toolbar.
- The "Articles" web content page template.

Note: The "Articles" page template can be deployed only if the Portal 8.5 theme is available in your portal.

Procedure

1. Go to the *wp_profile_root*`/ConfigEngine` directory.
2. To install the sample web content template items, choose one of the following options that is based on your environment:
   - If the Portal 8.5 theme is available in your portal, run the following command to install the library, content items, and Articles sample page:
     - AIX Linux Solaris: `./ConfigEngine.sh deploy-content-templating-ui -DPortalAdminId=user_name -DPortalAdminPwd=password -DWasUserid=user_name -DWasPassword=password`

- Windows: `ConfigEngine.bat deploy-content-templating-ui -DPortalAdminId=user_name -DPortalAdminPwd=password -DWasUserid=user_name -DWasPassword=password`
- IBM i: `ConfigEngine.sh deploy-content-templating-ui -DPortalAdminId=user_name -DPortalAdminPwd=password -DWasUserid=user_name -DWasPassword=password`

- If the Portal 8.5 theme is not available in your portal, run the following command to install the library and the content items:
  - AIX Linux Solaris: `./ConfigEngine.sh deploy-content-templating-library -DPortalAdminId=user_name -DPortalAdminPwd=password -DWasUserid=user_name -DWasPassword=password`
  - Windows: `ConfigEngine.bat deploy-content-templating-library -DPortalAdminId=user_name -DPortalAdminPwd=password -DWasUserid=user_name -DWasPassword=password`
  - IBM i: `ConfigEngine.sh deploy-content-templating-library -DPortalAdminId=user_name -DPortalAdminPwd=password -DWasUserid=user_name -DWasPassword=password`

3. Optional: The sample web content template items include tagging and rating components. These components are functionally available from the web content libraries that are provided with the blogs and wikis feature of the portal. If you intend to use tagging and rating with the templating sample content, you must ensure that the blogs and wikis libraries are installed. For details on installing these libraries, see *Blogs and wikis* in the migration section of the documentation.

Related concepts:

"Creating content with sample web content template items" on page 2026
To illustrate how page templates, web content viewers, and content associations work together, IBM Web Content Manager provides sample web content. The sample content includes examples of web content template pages and predefined portlets that you can add to pages to render content.

Related tasks:

"Blogs and wikis" on page 887
After you migrate from WebSphere Portal Express Versions 7.0 or 8.0 to Version 8.5, you must run a configuration task to update the presentation templates that are used by blogs and wikis to apply the latest updates. You must run this task for content in your blogs and wikis to render properly.

Default changed for JavaServer Faces implementation:

The default JavaServer Faces (JSF) implementation has changed starting in WebSphere Application Server Version 8.

When you are migrating JSF portlets from an earlier version of IBM WebSphere Portal Express, be aware that WebSphere Application Server has changed the default JSF implementation starting in WebSphere Application Server Version 8. For more information, see *JavaServer Faces migration* in the WebSphere Application Server documentation.

Related information:

JavaServer Faces migration

Portlets no longer available:

Some portlets that were available on previous releases of WebSphere Portal Express are no longer including in Version 8.5. These portlets are not migrated as part of the WebSphere Portal Express migration process.

Choose one of the following methods to correct any references to unsupported portlets:

- Many of the portlets are now available for download from the IBM Lotus and WebSphere Portal Business Solutions Catalog. If these portlets are still required, installation and deployment instructions are provided with the portlet download.
- Some of the portlets have replacements that are installed during the migration, and the references to the old portlets can be updated to refer to the replacements.
- If you need to retain the original functionality, copy the war file to *wp_profile_root*`/PortalServer/deployed/archive/`, and follow the instructions from *Updating Portlets URLs* in the post-migration activities section.
- If the functionality is no longer needed, delete those portlet references or pages that contain the portlets.

See the What's new section for details on what changed and what features are deprecated in this release.

Related concepts:

"What's changed" on page 16
WebSphere Portal Express includes changes to existing features.

"Unsupported and deprecated features for V8.5" on page 17
Review the features that were available in previous versions of IBM WebSphere Portal Express but are no longer available.

Related tasks:

"Updating portlets URL" on page 892
After migrating your portal environment, you must verify and update as needed the portlets URLs. In some cases, portlets have the incorrect URL after migration. URLs can be easily corrected using the XML configuration interface to export, update, and import the configurations.

Related information:

IBM Lotus and WebSphere Portal Business Solutions Catalog

Mashup integration:

The mashup integration feature was removed in WebSphere Portal Express Version 8.0 and is not included in later releases. If you used this feature in a previous release and you want to continue using it in WebSphere Portal Express Version 8.5, you need to manually enable mashup integration after migration.

About this task

In previous versions of WebSphere Portal Express, the `MashupMaker_Integration.ear` was registered with the system in all cases. Even when it was only required in cases where the mashup integration feature was used actively in the system. To remove system overhead, the `.ear` file was removed from the portal configuration during migration.

Procedure

To enable the mashup integration feature, register the `.ear` file with the following command:

- Linux : `./ConfigEngine.sh action-create-ear-wp.mmi.deploy`
- IBM i: `./ConfigEngine.sh action-create-ear-wp.mmi.deploy`
- Windows: `ConfigEngine.bat action-create-ear-wp.mmi.deploy`

Navigation with consecutive labels is dynamic:

The secondary navigation is dynamic, and shows only two levels of navigation at a time. When a page on the last level of navigation is selected, only that level of navigation is displayed. It is possible to select a page where both the parent level of navigation and the grandparent level of navigation are both labels. In this case, the sibling navigation for the parent level is not accessible.

Related concepts:

"Side navigation" on page 2715
The Portal 8.5 theme includes a side navigation template that can be applied to render pages at the secondary level in a list with the main content. By default, this template is applied to the Administration section of your portal.

Enabling new functionality in a migrated portal

The migration process collects configuration data and applications from an earlier installed version of IBM WebSphere Portal Express and merges them into the newer installed version so that the new environment is identical to the earlier environment. Taking advantage of new functionality that was not available in the earlier portal requires additional attention after migration is complete.

"New Web Content Manager features"
You might need to update your old web content to take advantage of the new Web Content Manager features.

"Enabling managed pages" on page 913
After migration, you must manually enable support for managed pages. Without managed pages support, some features like the project menu are not available on the migrated server.

"Changing from Ajax proxy to outbound HTTP connection" on page 916
WebSphere Portal Express Version 8.5 provides a migration process for the change from the Ajax proxy of previous portal versions to the new outbound HTTP connection.

"Enabling the new tag and rating widgets after a portal upgrade" on page 917
If you upgrade your IBM WebSphere Portal Express from an earlier version to Version 8.5 and want to use the new tag and rating widgets, you must first enable blogs and wikis. Then, complete the following task to ensure that tag and rating widgets are enabled.

"Enabling new functionality in migrated themes" on page 918
When you migrate your themes, migration moves your existing themes to the new server. Migration does not upgrade themes from an earlier version to use new functions that are introduced in more recent versions of the product.

New Web Content Manager features

You might need to update your old web content to take advantage of the new Web Content Manager features.

Approver role

The Approver role is replaced by the Reviewer role and Draft Creator role. All users assigned the Approver role are automatically reassigned to the Reviewer and Draft Creator roles during migration to ensure that the behavior of the workflowed items remains unchanged on the new system. These new roles support inheritance and propagation, removing the need to set them explicitly on each item in the workflow stage.

The users that are assigned to the Reviewer role and Draft Creator role should be reviewed post migration because not all users need to be assigned to both roles. Enabling inheritance for these roles is also recommended, not just to reduce the effort of maintaining permissions on individual items, but to also improve overall system performance due to the reduction of explicit role assignments. The simplest way to enable inheritance is to assign users to the Reviewer role and Draft Creator role at the library level.

Restart workflow

The minimum resource access that is required for the restart workflow function has been raised from edit access to the library, to manager access to the library. If you need to maintain the old access level, then you can change the following property in the `WCM WCMConfigService` service to "false" in the WebSphere Integrated Solutions Console:

```
workflowrestart.requires.manager=false
```

Web content configuration changes

Note: This was new in Web Content Manager version 8.0. No action is required if migrating from Web Content Manager version 8.0 or higher.

You no longer use the `WCMConfigService.properties` file to update configuration settings for Web Content Manager. Instead, you now use the `WCM WCMConfigService` service to update configuration settings with the WebSphere Integrated Solutions Console. For more information, see “Setting service configuration properties” on page 283.

Web content tag format changes

Note: This was new in Web Content Manager version 8.0. No action is required if migrating from Web Content Manager version 8.0 or higher.

Web content tags now use brackets. This allows you to add web content tags directly into rich text fields.

For example, in previous versions the component tag was written as:

```
<component name="componentname" />
```

The component tag now uses brackets:

```
[component name="componentname" ]
```

Note:

- All web content tags are converted to the new format during migration.
- After migration, if a user enters a web content tag with the old format, it will be converted to the new format when saved.

New web content property tag

Note: This was new in Web Content Manager version 8.0. No action is required if migrating from Web Content Manager version 8.0 or higher.

The `IDCmpnt`, `HistoryCmpnt`, `ProfileCmpnt`, `WorkflowCmpnt`, and `SecurityCmpnt` tags are no longer supported. These features are retained and consolidated into a new property tag.

```
[Property field=" " context=" " type=" " name=" " key=" " format=" " link=" " separator=" "
htmlencode=" " awareness=" " ifEmpty=" " include=" " restrict=" " resolve=" "
start=" " end=" " ]
```

Note:

- These tags are converted to the new property tag during migration.
- After migration, if a user enters one of the deprecated tags, it will be converted to the new property tag when saved.

Autofill parameter in web content tags

Note: This was new in Web Content Manager version 8.0. No action is required if migrating from Web Content Manager version 8.0 or higher.

The behavior of the **`autofill`** parameter that is used in some web content tags is updated. If used in an item where **`autofill`** is not applicable, the tag instead uses the context of the current item.

If you do not want this behavior to be used, you can add the following property in the `WCM WCMConfigService` service in the WebSphere Integrated Solutions Console

```
renderAutoFillTagsAsCurrent=false
```

Federated content component and element

Note: This was new in Web Content Manager version 8.0. No action is required if migrating from Web Content Manager version 8.0 or higher.

The federated content component and element are no longer supported in version 8.0. To reference federated content in your website, use one of the following features:

- The Enterprise Content Manager window
- The Web Content Integrator
- The federated document feature of Personalization

Federated content components and elements are migrated from previous versions and are visible in the authoring portlet. These items can still be rendered in web pages, but you cannot create new federated content components or elements unless you enable this function.

To maintain older systems, enable federated content components and elements in your new system by adding the following property in the `WCM WCMConfigService` service in the WebSphere Integrated Solutions Console:

```
federatedcontent.enabled=true
```

Web content viewer portlet display title options

The standard Web Content Viewer portlet supports extra portlet and page display title options. You can now configure web content viewers to use the value of an element of the displayed content item as portlet and page display title.

Links and metadata for remote content

Note: This was new in Web Content Manager version 8.0. No action is required if migrating from Web Content Manager version 8.0 or higher.

The federated documents feature of Web Content Manager is used to insert links to content from a remote content system or document repository. Examples of supported repositories include IBM Content Manager, IBM FileNet Content Manager, and Microsoft SharePoint. This capability is provided by the rich text editors in Version 8.5:

- The default and advanced rich text editors include a toolbar button named **Insert Link to Remote Document**.
- In addition to the toolbar button, the advanced rich text editor includes a menu item named **Insert Link to Remote Document**.

You can also use the federated documents feature with Personalization to create selection rules. Personalization is used to retrieve metadata about documents that are stored in external content management systems or document repositories. With personalization components in Web Content Manager, you can display the metadata and create links to the documents in your web content. To select remote content in the Personalization Editor, the federated documents feature provides a wizard.

In Version 8.5, the capabilities of the portal theme determine whether the user interface features for Web Content Manager and Personalization are available.

To enable the **Insert Link to Remote Content** function after migration, ensure that the `wp_federated_documents_picker` theme module is available on the migrated portal. Any page that contains the authoring portlet must use a theme that integrates this theme module. In addition, to ensure that the module is loaded, the module profile that is used by the page must include the `wp_federated_documents_picker` theme module. If the feature is disabled when you use inline editing of web content, complete these steps:

- Apply a theme to the hidden authoring page that contains the `wp_federated_documents_picker` theme module. For example, you can apply the Portal 8.5 theme.
- Apply a module profile to the hidden authoring page that contains the `wp_federated_documents_picker theme` module. For example, you can apply the deferred profile of the Portal 8.5 theme.
- Apply a skin without decorations to the reserved authoring portlet instance on the hidden authoring page. For example, you can apply the "Portal 8.5 - No Skin" skin.

To enable the folder selection wizard in Personalization, ensure that the `wp_federated_documents_picker` theme module is available to the page that contains the Personalization Editor.

Related concepts:

Supported specifications for inserting links to remote content
The federated documents feature enables you to retrieve information about documents in a remote repository and insert links to those documents in your web content. Several types of feeds are supported for accessing remote content.

Related tasks:

"Configuring the reserved authoring portlet" on page 431
The reserved authoring portlet is essential to the proper operation of web content pages and the web content viewer, so it is important that the configuration of the reserved authoring portlet is similar to the configuration of other instances of the authoring portlet.

"Personalizing federated documents" on page 1827
Portal Personalization provides the federated documents feature to retrieve metadata about documents that are stored in external content management systems or document repositories. Examples of these systems include IBM Content Manager, IBM FileNet Content Manager, and Microsoft Sharepoint. You can use a personalization component in IBM Web Content Manager to display metadata from federated documents and to create links to download or open the documents.

Enabling managed pages

After migration, you must manually enable support for managed pages. Without managed pages support, some features like the project menu are not available on the migrated server.

About this task

When you enable managed pages after migration, several considerations apply:

- After migration, managed pages are initially disabled on the migrated server. In this case, there is a single workspace for storing IBM Web Content Manager content. All virtual portals on the server share this workspace. Although this single workspace simplifies the sharing of content between virtual portals, it can also lead to dependencies between content libraries and different virtual portals. Depending on your content environment, these dependencies can make it difficult for an administrator to determine which libraries to select for syndication to a syndicator.
- When you enable managed pages, a workspace is created for each virtual portal. This separation ensures that there are no cross-references between content items in different virtual portals. In addition, by selecting all libraries that are visible in a workspace, references are guaranteed to be resolved during syndication.

  Because of this separation of workspaces, extra syndication steps might be required on the migrated server for web content libraries:

  Libraries that are used only by a specific virtual portal
  You must syndicate the libraries from the default virtual portal to the specific virtual portal.

  Libraries that are shared by multiple virtual portals
  You must syndicate the shared libraries from the default virtual portal to each specific virtual portal.

  After you syndicate the libraries that are unique to specific virtual portals, you can delete the libraries from the default virtual portal.
- When supported, managed pages is enabled for a virtual portal, all pages in the virtual portal are copied into the Portal Site library in IBM Web Content Manager. However, the following pages are not treated as managed pages and are not copied:

- Administration pages, as identified by the label `ibm.portal.Administration` and its child pages
- Private pages

Each virtual portal has its own Portal Site library.

- In scenarios where the configured user realm does not contain the domain administrator, an alternative user must be provided. If you have a multi-realm configuration, see "Defining alternative administrators for multi-realm configurations" on page 436.

Note: To take advantage of the features available to managed pages in the user interface, your pages must use the Portal 8.5 theme.

Procedure

1. Start the portal server.
2. To enable support for managed pages, run the **enable-managed-pages** task from the *wp_profile_root*/ConfigEngine directory.

   Windows
   ```
   ConfigEngine.bat enable-managed-pages -DPortalAdminPwd=password
   -DWasPassword=password
   ```

   Linux
   ```
   ./ConfigEngine.sh enable-managed-pages -DPortalAdminPwd=password
   -DWasPassword=password
   ```

   IBM i
   ```
   ConfigEngine.sh enable-managed-pages -DPortalAdminPwd=password
   -DWasPassword=password
   ```

   After you run the **enable-managed-pages** task for the first time, the property **managed.pages** is created in the portal WP Configuration Service. The value of the property is set to `true`.
3. Restart the portal server.
4. To populate web content libraries with information about virtual portals in the system, run the **create-virtual-portal-site-nodes** task from the *wp_profile_root*/ConfigEngine directory. For each virtual portal, this task creates a library and a site area that is called `lost-found` for resources that cannot be properly located. If the library or site area exist, the task exits. By default, the task runs on all virtual portals in the system.

   Windows
   ```
   ConfigEngine.bat create-virtual-portal-site-nodes
   -DPortalAdminPwd=password -DWasPassword=password
   ```

   Linux
   ```
   ./ConfigEngine.sh create-virtual-portal-site-nodes
   -DPortalAdminPwd=password -DWasPassword=password
   ```

   IBM i
   ```
   ConfigEngine.sh create-virtual-portal-site-nodes
   -DPortalAdminPwd=password -DWasPassword=password
   ```
5. To populate web content libraries with information about the portal pages in the system, run the **create-page-nodes** task from the *wp_profile_root*/ConfigEngine directory.

   This task can also be used when portal pages and managed pages artifacts in Web Content Manager are not synchronized. In this case, the task attempts to resynchronize the portal artifacts and web content artifacts, giving precedence to the portal artifacts.

   Performance note: Depending on the amount of information in the system, the **create-page-nodes** task can take a long time to run. Because of the database

load of the task, do not run the task frequently. The initial run of the task requires the most time, while subsequent runs typically require less time.

Note: If you have many pages, then it might be necessary to increase the soap client timeout. Edit the *wp_profile_root*/properties/soap.client.props file to change the **com.ibm.SOAP.requestTimeout** to 60000.
Attention: If your virtual portals have different administrative accounts, you cannot run the **create-page-nodes** task directly. You must run the task for every virtual portal, including the base virtual portal. Use the **VirtualPortalHost** or **VirtualPortalContext** parameter with the **create-page-nodes** task. Run the **list-all-virtual-portals** task to get a list of all your virtual portals. When you run the create-page-nodes task on the base virtual portal, set the **VirtualPortalContext** value to __NO__VP__ID__.

Windows
```
ConfigEngine.bat create-page-nodes -DPortalAdminPwd=password
-DWasPassword=password
```

Linux
```
./ConfigEngine.sh create-page-nodes -DPortalAdminPwd=password
-DWasPassword=password
```

IBM i
```
ConfigEngine.sh create-page-nodes -DPortalAdminPwd=password
-DWasPassword=password
```

By default, this task is run on all pages in all virtual portals. To limit this task to a specific virtual portal, identify the virtual portal by adding one of the following parameters to the command line. Each parameter requires the prefix -D on the command line.

VirtualPortalHost
Specify the host name of the virtual portal. For example, vp.example.com.

Important: If the host name of the virtual portal is the same as the host name of the default virtual portal, you must also specify the **VirtualPortalContext** property. You can specify the **VirtualPortalHost** property by itself only if the host name is unique.

VirtualPortalContext
Specify the virtual portal context that identifies the virtual portal. For example, vp1.

You can customize the task with the following optional parameters on the command line. Each parameter requires the prefix -D on the command line.

RunParallel
Indicate whether you want the task to run with multiple threads. A value of false indicates a single thread and is the default setting.

A value of true indicates multiple threads, as specified by the work manager **wpsJcrSyncWorkManager** in the WebSphere Integrated Solutions Console. Each thread requires a database connection. For optimal performance, ensure that your database connection pool supports at least as many connections as there are threads in the pool.

Excluded
Specify a list of unique names of page nodes to exclude from the creation process. Excluding a page also excludes its child pages. By default, the portal administration pages (ibm.portal.Administration) are excluded.

6. Optional: If you used web content pages before you enabled managed pages, you can transfer the content for those pages to the Portal Site library. If you plan to use the default page templates and store your web content in the Portal Site library, transfer the content for the template pages to the Portal Site library. For more information, go to *Transferring content associations to the Portal Site library*.
7. Optional: Ensure that users have appropriate access to the Web Content Manager REST virtual resource so they can use **Edit mode**. For example, they have `user` access.

What to do next

After you migrate from Version 7.0 or Version 8.0, vanity URLs are not available. After you enable managed pages, you can enable vanity URL support. If you migrated from Version 7.0 or 8.0, go to "Enabling vanity URL support" on page 2100.

Related tasks:

"Transferring content associations to the Portal Site library" on page 380
When you enable manage pages, any web content pages that you have are converted to managed pages and added to the Portal Site library. However, the content that is associated with the web content pages remains in the original libraries. You can transfer this associated content to the Portal Site library with the `internalize-content-mappings` task.

"Enabling vanity URL support" on page 2100
In a new WebSphere Portal Express Version 8.5 installation, vanity URL support is enabled. If you upgrade your WebSphere Portal Express from a previous version to Version 8.5, vanity URL support is disabled. You can enable and disable vanity URL support as required by using a portal configuration task.

"Synchronizing the vanity URL database" on page 2104
Vanity URLs are stored as part of the page in the JCR database in the portal page site area of Web Content Manager. For performance reasons, the data is also stored in the WebSphere Portal Express database. When the data is modified, the portal synchronizes the data between both sides. However, under certain circumstances it can happen that the data is not synchronized. For such cases, the portal provides a configuration task that synchronizes the data.

Changing from Ajax proxy to outbound HTTP connection

WebSphere Portal Express Version 8.5 provides a migration process for the change from the Ajax proxy of previous portal versions to the new outbound HTTP connection.

If you upgrade your WebSphere Portal Express from a previous version to Version 8.5, the outbound HTTP connection infrastructure attempts to migrate existing Ajax proxy configuration settings to the outbound HTTP connection service configuration. It moves all settings from the Ajax proxy configuration into database settings for the outbound HTTP connection service. This migration takes place as follows:

Migration of global configuration settings:

In previous WebSphere Portal Express versions, global proxy configuration settings were in the `proxy-config.xml` file of the Ajax Proxy Configuration web module. The outbound HTTP connections infrastructure migrates global proxy configuration settings when the portal Version 8.5 outbound

HTTP connection service is accessed for the first time after the portal upgrade. The global configuration settings are updated in the system configuration profile.

Migration of customized global configuration settings:
In previous WebSphere Portal Express versions, custom global proxy configuration settings were in the WP Configuration Service Resource Environment Provider (REP) property named `proxy.config.file`. The outbound HTTP connections infrastructure migrates these custom global configuration settings when the portal Version 8.5 outbound HTTP connection service is accessed for the first time after the portal upgrade. Custom global configuration settings are updated into the global configuration profile.

Migration of application-specific proxy configurations
Application-specific configurations are imported into the outbound HTTP connection configuration when the web module for the application is deployed. The deployment program scans for the file `/WEB-INF/proxy-config.xml`. If this file exists, the migration program creates a scoped configuration profile that relates to the web module that is created.

Important: In previous WebSphere Portal Express versions, the `proxy-config.xml` file held the configuration of the Ajax proxy. Changes that you make to this file became effective when you restarted the web module. In contrast, the outbound HTTP connection infrastructure makes only the necessary updates to the `proxy-config.xml` file. Changes to this file do not become effective until one of the following events occur:

- For global or customized global configuration settings: The outbound HTTP connection service is started for the first time after the portal upgrade.
- For application-specific proxy configurations: The web module that contains the `proxy-config.xml` is deployed.
- One of the portal configuration tasks `create-outbound-http-connection-config` or `update-outbound-http-connection-config` is started.

Related concepts:

"Outbound HTTP connection" on page 2984
Applications in your IBM WebSphere Portal Express and the related user activities can require outbound HTTP connections to remote computer systems. The outbound HTTP connection service provides an administration infrastructure with a central point of administration for all outbound HTTP connections that are defined in the portal environment.

Related tasks:

"Configuring outbound HTTP connections by using configuration tasks" on page 3013
Programmers can create, read, update, or delete settings of the outbound HTTP connection by using the appropriate portal configuration engine tasks.

Enabling the new tag and rating widgets after a portal upgrade

If you upgrade your IBM WebSphere Portal Express from an earlier version to Version 8.5 and want to use the new tag and rating widgets, you must first enable blogs and wikis. Then, complete the following task to ensure that tag and rating widgets are enabled.

Before you begin

You must have blogs and wikis enabled before proceeding. Go to "Blogs and wikis" on page 887 for more information.

Procedure

1. Verify that the IBM Web Content Manager components are updated.
   a. Open the applications menu and go to **Content** > **Web Content Authoring** > **Libraries** > **Web Resources v70** > **Components**. If the Web Resources v70 library is not listed by default, add it. Open the applications menu and go to **Content** > **Web Content Authoring** > **Preferences** > **Edit Shared Settings** > **Library selection**. Select **Web Resources v70** and click **Add** and **OK**.
   b. Verify that the components **HTML-Tagging Widget Light- Tags** and **HTML-Rating Widget Light- Stars** are listed. If they are not listed, check the logs to determine whether the configuration task might have resulted in errors.
2. Turn on **Edit Mode**.
3. For the tag and rating widgets to render properly, go to **Menu** > **Edit Page Properties** and select the Portal 8.5 theme for your portal theme and click **Save**. The Portal 8.5 theme is now enabled.

   Note: Apply the Portal 8.5 theme to the page on which you want to use the tagging and rating widgets.
4. Go to **Edit Page Properties** and select the "Basic Content" profile for your portal profile.

Results

You and your portal site visitors can now use the new enhanced tag and rating widgets.

Enabling new functionality in migrated themes

When you migrate your themes, migration moves your existing themes to the new server. Migration does not upgrade themes from an earlier version to use new functions that are introduced in more recent versions of the product.

When migrating modular themes, the modular theme architecture makes it easy to enable new functions in your themes by turning on and off the modules or versions of the modules. For example, if you want to update your theme to use the current version of Dojo, replace the `dojo`*`xx`*`.json` file in the capabilities folder, where *xx* is the version number. Your Dojo meta-modules load the newer version of Dojo as specified in the `dojo`*`xx`*`.json` file.

WebSphere Portal Express provides an optimized, modular theme architecture from Version 7.0.0.2 onward. Non-modular Version 6.1, such as Portal, PortalWeb2, and Tab Menu - Page Builder themes were deprecated in Version 8.0 and are no longer supported in Version 8.5.

Non-modular version 7.0 (Page Builder) themes are still supported on version 8.5 but are also deprecated and will be unsupported in the next version of WebSphere Portal Express. It is either required (6.1 themes) or recommended (7.0 themes) that you upgrade to a modular theme. If you are starting a new project, you must start with a copy of the 8.5 theme. If you must migrate a previous project, the older

non-modular themes can physically be migrated. But, you must then manually update your themes to the modular architecture, which allows you to enable any new features.

"Device class support in a migrated theme"
You can use the new device classes and equation support to modify your themes from previous versions so they are more efficient.

"Migration - Removing the site toolbar in an 8.0 theme" on page 920
To use the new site toolbar of WebSphere Portal Express 8.5 within a WebSphere Portal Express 8.0 theme, you must first remove the existing toolbar that comes with your WebSphere Portal Express 8.0 theme.

"Migration - Add the version 8.5 site toolbar to a version 8.0 theme" on page 921
You can easily add the modularized site toolbar of WebSphere Portal Express Version 8.5 to a WebSphere Portal Express 8.0 theme. Or, you can add a toolbar to a custom theme that is derived from the WebSphere Portal Express 8.0 theme. The theme must be a modularized theme, which supports theme profiles and theme modules.

"Updating your theme to use simple context menus" on page 924
The theme context menu framework that is provided with the module `wp_theme_menus` in IBM WebSphere Portal Express 8.0 and earlier is relabeled and improved in WebSphere Portal Express 8.5. The new Simple Menu Framework is compatible with all previous versions of the `wp_theme_menus`.

Device class support in a migrated theme:

You can use the new device classes and equation support to modify your themes from previous versions so they are more efficient.

With the two new features, you can remove or modify certain aspects of a prior theme that you migrated. These changes are optional, but these new features are more powerful and simpler to use.

This version of Portal supports `ios`, `android`, `blackberry`, and `worklight` in addition to the `smartphone` and `tablet` themes it already supports. You can also use equation support to combine individual device classes to check for multiple client conditions.

For example, if you were manually integrating with MobileFirst in a previous version, and manually created some combination device classes, such as `smartphone-ios`, `tablet-ios`, `smartphone-android`, `tablet-android`, your combination device classes are no longer needed. The individual device classes can be combined in equations, such as `smartphone+ios`, `tablet+ios`. If you had a combination device class such as `hires-tablet-ios`, you can change to define the singular `hires` device class. Then, use equations to combine with the other default singular device classes, such as `hires+tablet+ios`. In these equations, use `+` for AND, `/` for OR, `!` for NOT and parentheses to group items together. For more information, see *Device Class Equations*.

In your `.jsp` logic, for example, you can change the previous version syntax with `<c:if>` against `deviceClass`. Here is an example of code from a previous version:

```
<c:set var="deviceClass" scope="request" value="${wp.clientProfile['DeviceClass']}" />
<c:set var="isMobile" scope="request" value="${deviceClass == 'tablet' || deviceClass == 'smartpho
<c:if test="${isMobile}">
...
</c:if>
```

You can change that to a new equation syntax with `<portal-logic:if>` with the new **deviceClass** parameter. For example:

```
<portal-logic:if deviceClass="tablet/smartphone">
...
</portal-logic:if>
```

You can now use device class equations in module subcontributions in theme contributions `.json` files. For example:

```
}, {
     "value":"/css/my_css_smartphone_ios.css",
     "type":"smartphone+ios"
    }, {
```

Dynamic content spots in theme templates can now also be varied entirely with device class equations. For example:

```
mvc:res:/hello.jsp,smartphone+ios@res:/hello_smartphone_ios.jsp,(smartphone/tablet)+android@res:/hel
```

In this example, you can use `res:/hello.jsp` as the default URI, `res:/hello_smartphone_ios.jsp` as the URI for iOS smartphones, and `res:/hello_mobile_android.jsp` as the URI for Android smartphones and tablets.

Migration - Removing the site toolbar in an 8.0 theme:

To use the new site toolbar of WebSphere Portal Express 8.5 within a WebSphere Portal Express 8.0 theme, you must first remove the existing toolbar that comes with your WebSphere Portal Express 8.0 theme.

About this task

All of the theme modules providing functions for editing and managing the site, which are also available in the new site toolbar, can be removed when you remove the toolbar. You cannot add the WebSphere Portal Express 8.5 site toolbar to a WebSphere Portal Express 8.0 theme while keeping the version 8.0 toolbar.

Procedure

1. Remove the theme modules from the WebSphere Portal Express 8.0 site toolbar from the theme profiles. Go through the theme profiles of your Portal 8.0 theme and remove the theme modules that are listed.
   - `wp_theme_edit`
   - `wp_project_menu`
   - `wp_preview`
   - `wp_toolbar`
   - `wp_project_menu_edit`
   - `wp_preview_menu`
   - `wp_pagebuilder_controls`
   - `wp_pagebuilder_dnd`
   - `mm_new_page_dialog`
   - `mm_builder_wiring`
   - `mm_move_page`
   - `mm_delete_page`
   - `mm_delete_control`
   - `mm_page_sharing_permission`

2. Remove the dynamic content spots of the WebSphere Portal Express 8.0 site toolbar from the theme templates. Edit the theme HTML templates of your WebSphere Portal Express 8.0 theme and remove the dynamic content spots with the following IDs.
   - `80theme_preview`
   - `80theme_projectMenu`
   - `80theme_toolbar`
   - `80theme_pageModeToggle`
3. For each of the dynamic content spots, look for HTML anchor elements that reference the spots with its ID in your templates.

   ```
   <a rel="dynamic-content" href="dyn-cs:id:spot-id"></a>
   ```
4. Remove the HTML anchor element from the template and save the template file. Depending on your theme, check if there is other decorating HTML markup which can be removed in addition to the anchor elements referencing the dynamic content spots. Verify that the changes are applied to the theme template of each locale.
5. In the default version 8.0 theme, the names of the theme template files are `theme.html` and `theme_sidenav.html`. The localized versions of `theme.html` can be found in the locale subfolder, `nls`. For example `theme_en.html`. The theme template `Plain.html` does not need to be edited.
6. In the `menuDefinitions` directory, remove `shelfActions.json` and `moreActions.json`.
7. In the `menuDefinitions` directory, you can optimize the menu definition `skinActions.json` by removing the menu contributions in `mm_builder_wiring`, `wp_pagebuilder_controls`, and `mm_delete_control`.
8. In the `system` directory, keep `layouts.json` and `styles.json`. Remove all other files.

Migration - Add the version 8.5 site toolbar to a version 8.0 theme:

You can easily add the modularized site toolbar of WebSphere Portal Express Version 8.5 to a WebSphere Portal Express 8.0 theme. Or, you can add a toolbar to a custom theme that is derived from the WebSphere Portal Express 8.0 theme. The theme must be a modularized theme, which supports theme profiles and theme modules.

You can add the site toolbar of WebSphere Portal Express Version 8.5 to an existing modularized theme. You can also remove the WebSphere Portal Express 8.0 site toolbar from it. In the WebSphere Portal Express Version 8.5 theme, the WebSphere Portal Express Version 8.5 site toolbar is available by default. If you migrate your system to WebSphere Portal Express Version 8.5, you need to install the site toolbar first by running a configuration task. For more information, *Enabling the 8.5 site toolbar*.

Note: You must add the Portal 8.5 toolbar to a migrated custom theme that you use. Adding the toolbar enables your content creators to take advantage of updates to the site manager and other updates in the maintenance of your portal including creating, reusing, deleting, copying, and moving pages and content items.

There are two ways to embed a WebSphere Portal Express Version 8.5 toolbar into your WebSphere Portal Express 8.0 theme.

1. You can embed the site toolbar by adding dynamic content spot to your theme HTML template. You must be familiar with theme templates and dynamic content spots to use this option. For more information, see *Dynamic content spots*.
2. You can embed the site toolbar dynamically without adding dynamic contents spots to your theme HTML. You must be familiar with theme modules and theme profiles to use this option. For more information, see *The module framework*.

Both options are valid. The dynamic content spots option provides a more customizable user experience. There are no flickering effects because of the dynamic content spot. However, the dynamic option offers integration without editing the theme HTML template. It is easier to implement. Both options require that you add a certain theme module to the theme profiles of your theme.

"Migration - Embedding a toolbar into a theme by adding a dynamic content spot to an HTML template"
Embedding a toolbar into a theme with a dynamic content spot provides an optimized user experience when compared to embedding it dynamically.

"Migration - Embedding the version 8.5 site toolbar dynamically without a dynamic content spot" on page 924
Embedding a toolbar into a theme dynamically integrates the toolbar without editing the theme HTML template.

Related concepts:

"Dynamic content spots" on page 2666
The static template files use dynamic content spots to reference JSP files or other dynamic resources. The dynamic resources are stored in a WAR file.

"The module framework" on page 2526
The module framework allows extensions to contribute to different areas of a page to provide flexibility, enhance the user experience, and maximize performance.

"Understanding the Portal Version 8.5 modularized theme" on page 2521
Modern websites and browsers enable incredible new capabilities that can greatly enhance your user's web experiences. However, these capabilities are not without cost in terms of large page sizes and more processing in the browser when each page is rendered. These capabilities are worth it when you need them, but removing them for an entire site or including them only on pages that take advantage of these capabilities provides for more flexibility.

"Customizing the theme" on page 2658
The module framework allows themes to be customized in order to provide flexibility, enhance the user experience, and maximize performance.

Related tasks:

"Enabling the 8.5 site toolbar" on page 878
During migration, the IBM WebSphere Portal Express Version 8.5 theme is deployed. The 8.5 theme includes the site toolbar, but the toolbar is enabled only for the default virtual portal. For all other virtual portals, including migrated and newly created virtual portals, you must install the site toolbar separately.

Migration - Embedding a toolbar into a theme by adding a dynamic content spot to an HTML template:

Embedding a toolbar into a theme with a dynamic content spot provides an optimized user experience when compared to embedding it dynamically.

Procedure

1. Configure the site toolbar. To add the site toolbar theme modules to the theme profiles of your theme, choose one of the three different theme modules.

   wp_toolbar_host
   The first level module of the WebSphere Portal Express 8.5 site toolbar. It groups all of the resources that are required to run the toolbar in your theme. It supports view mode and edit mode.

   wp_toolbar_host_view
   Contains all toolbar resources that are needed for view mode.

   wp_toolbar_host_edit
   Contains all toolbar resources that are needed for edit mode.

2. Use `wp_toolbar_host` alone, or combine `wp_toolbar_host_view` with the `wp_toolbar_host_edit` module.
   - If you have a theme profile which does not have a deferred section, you need to add the `wp_toolbar_host` theme module.

     ```
     {
     "moduleIDs" : [
     "getting_started_module",
     "wp_toolbar_host",
     ...
     ]
     ...
     }
     ```

   - If you have a theme profile with a non-deferred and a deferred section, it is recommended to combine the `wp_toolbar_host_view` with the `wp_toolbar_host_edit` module. In that case you must add the `wp_toolbar_host_view` to the non-deferred section and `wp_toolbar_host_edit` to the deferred section.

     ```
     {
     "moduleIDs" : [
     "getting_started_module",
     "wp_toolbar_host_view",
     ...
     ],
     "deferredModuleIDs" : [
     "wp_toolbar_host_edit",
     ...
     ],
     ...
     }
     ```

     The theme module `wp_toolbar_host_edit` will not be loaded until you enter edit mode.

3. Add the dynamic content spot of the version 8.5 toolbar to the theme templates. To integrate the new toolbar to your theme, you must add a dynamic content spot to the theme HTML templates of your theme. The ID of the dynamic content spot is `85toolbar`.
4. To add this dynamic content spot, create a HTML anchor element which references this dynamic content spot. Embed the dynamic content spot in the header of your page

   ```
   <div class="wpthemeFrame">
   <header role="banner">
   <a rel="dynamic-content" href="dyn-cs:id:85toolbar"></a>
   ...
   ```

5. Add this dynamic content spot to the locale-specific versions of your theme templates.

6. In the default version 8.0 theme, the names of the theme template files are `theme.html` and `theme_sidenav.html`. The localized versions of `theme.html` can be found in the locale subfolder, `nls`. For example `theme_en.html`.
7. After you embed a toolbar, you must either restart your portal server. Or, you must use the WebSphere Integrated Solutions Console to invalidate the resource aggregator cache. To invalidate your cache, click **Theme Analyzer** > **Utilities** > **Control Center** > **Invalidate Cache**. CF06 Auto invalidation recognizes your changes automatically for WebDAV based themes. No further action is required. For more information, see Utilities.

Migration - Embedding the version 8.5 site toolbar dynamically without a dynamic content spot:

Embedding a toolbar into a theme dynamically integrates the toolbar without editing the theme HTML template.

Procedure

1. Add `wp_toolbar_host_dynamic` to the non-deferred section of your theme profiles. This theme module contains both the resources for view mode and edit mode.

```
{
"moduleIDs" : [
"getting_started_module",
"wp_toolbar_host_dynamic",
...
]
...
}
```

2. See *Adding or removing a ready-to-use module to a theme* to add a module to your profile.
3. If you are not using WebDAV to connect to your theme resources, see *Setting a profile override on a page*.
4. After you embed a toolbar, you must either restart your portal server. Or, you must use the WebSphere Integrated Solutions Console to invalidate the resource aggregator cache. To invalidate your cache, click **Theme Analyzer** > **Utilities** > **Control Center** > **Invalidate Cache**. CF06 Auto invalidation recognizes your changes automatically for WebDAV based themes. No further action is required. For more information, see Utilities.

Updating your theme to use simple context menus:

The theme context menu framework that is provided with the module `wp_theme_menus` in IBM WebSphere Portal Express 8.0 and earlier is relabeled and improved in WebSphere Portal Express 8.5. The new Simple Menu Framework is compatible with all previous versions of the `wp_theme_menus`.

About this task

If you want to use the WebSphere Portal Express simple context menus, you must complete the following steps on your migrated theme.

Procedure

1. Update the existing module `wp_theme_menus` in *`WebDAV`*`/themes/`*`your_theme`*`/contributions/theme.json` and replace it with:

```
{
 "id":"wp_theme_menus",
 "prereqs":[{
  "id":"wp_simple_contextmenu_main"
}]
}
```

2. Copy the following artifacts from the Portal 8.5 theme to your migrated theme in the same directory:
   - *WebDAV*/themes/Portal 8.5/contributions/simple_contextmenu.json
   - *WebDAV*/themes/Portal 8.5/css/wp_simple_contextmenu.css
   - *WebDAV*/themes/Portal 8.5/css/wp_simple_contextmenu.css.uncompressed.css
   - *WebDAV*/themes/Portal 8.5/css/wp_simple_contextmenuRTL.css
   - *WebDAV*/themes/Portal 8.5/css/wp_simple_contextmenuRTL.css.uncompressed.css
   - *WebDAV*/themes/Portal 8.5/css/default/contextmenu.css
   - *WebDAV*/themes/Portal 8.5/css/default/contextmenu.css.uncompressed.css
   - *WebDAV*/themes/Portal 8.5/css/default/contextmenuCommon.css
   - *WebDAV*/themes/Portal 8.5/css/default/contextmenuCommon.css.uncompressed.css
   - *WebDAV*/themes/Portal 8.5/css/default/contextmenuRTL.css
   - *WebDAV*/themes/Portal 8.5/css/default/contextmenuRTL.css.uncompressed.css
   - *WebDAV*/themes/Portal 8.5/menuDefinitions/templates/simpleMenuTemplate.html
3. Click the **Administration menu** icon. Then, click **Portal Analysis** > **Theme Analyzer**. Then, click **Utilities** > **Control Center** > **Invalidate Cache** to invalidate the theme cache. You must invalidate the cache so that your profile and module changes are picked up by the Portal server. CF06 Auto invalidation recognizes your changes automatically for WebDAV based themes. No further action is required. For more information, see Utilities.

Chapter 10. Integrating

Integrate IBM WebSphere Portal Express with software such as IBM Sametime to enable your users to collaborate more easily. You can also use the unified task list portlet, available on the WebSphere Portal Express Solution Catalog to integrate WebSphere Portal Express with your backend business process software, such as IBM WebSphere Process Server.

"Integrate with collaboration software "
IBM WebSphere Portal Express integrates with collaboration software to provide you with more effective and cost-efficient ways of accessing information, sharing ideas, communicating and working together. Key software that WebSphere Portal Express integrates with includes IBM Domino, IBM Sametime, and IBM Connections.

"Integrating business processes" on page 967
Access relevant tasks and activities for multiple business process management solutions from a single user interface. The Unified Task List portlet provides a single point of integration between multiple systems and displays tasks that WebSphere Portal Express users must complete to advance workflows. The topics in this section provide information and instructions for developing and customizing the Unified Task List.

"Integrating with web applications" on page 977
The web application bridge uses reverse proxy technology to integrate web-based content providers, such as the Microsoft SharePoint server, with IBM WebSphere Portal Express. Administrators must first define the virtual web applications or content providers. A lightweight iFrame portlet renders the content from the backend applications. Users can then access the iFrame on a page without requiring direct network access to the backend application. A special engine maps Uniform Resource Identifier (URIs) on the iFrame portlet to real URIs from the content providers.

"Integrating with SAP NetWeaver Portal" on page 986
You can use IBM WebSphere Portal Express Integrator for SAP to integrate content from an SAP NetWeaver Portal into your IBM WebSphere Portal Express. You can integrate navigational structures and single content pieces.

"Integrating with IBM MobileFirst" on page 1002
You can integrate WebSphere Portal Express with MobileFirst to provide multi-channel support to your web communities. You can create a hybrid application that adds native device functions and a unified web experience on mobile device browsers and in mobile device native applications. You can use MobileFirst to create a hybrid application that adds native device functions to your portal.

"Integrating with Brightcove" on page 1032
If you are using videos as part of your website, integrate your portal with the Brightcove video streaming server.

Integrate with collaboration software

IBM WebSphere Portal Express integrates with collaboration software to provide you with more effective and cost-efficient ways of accessing information, sharing ideas, communicating and working together. Key software that WebSphere Portal Express integrates with includes IBM Domino, IBM Sametime, and IBM Connections.

"Finding users"
Use PeopleFinder or Directory Search to locate other WebSphere Portal users. The People Finder portlet can be added to a page through the Content menu, so that users can find people in any context. Directory Search is available to users as part of a larger task, such as sending mail. Directory Search also enables users to search for user groups in addition to individual users.

"Planning for collaborative servers and portlets" on page 943
Setting up a site with IBM Domino integration requires decisions about user directories, security, authentication, and performance. View some use cases that may help you make decisions leading to a successful Domino integration.

"Integrating with IBM Sametime" on page 953
Configuring the Sametime Community Server and Sametime portlets requires a set of tasks performed on both the Sametime Community Server and the portal server.

"Collaborative Services environment properties" on page 962
If your collaborative site requires optional configuration that is not accomplished by the tasks that you run on the IBM WebSphere Portal Express server to integrate IBM Domino and the collaboration products, you can modify the operation of the collaborative servers and portlets in various ways by manually editing the Lotus Collaborative Services environment properties file (`CSEnvironment.properties`) on the WebSphere Portal Express server.

Finding users

Use PeopleFinder or Directory Search to locate other WebSphere Portal users. The People Finder portlet can be added to a page through the Content menu, so that users can find people in any context. Directory Search is available to users as part of a larger task, such as sending mail. Directory Search also enables users to search for user groups in addition to individual users.

"People Finder"
The **People Finder** portlet provides both quick search and advanced search options for locating people and information about people. Once found, a person is visible to other users as a person link that indicates online presence and displays a menu of instant messaging and other options.

"Directory Search" on page 938
The Directory Search or "People Picker" portlet is a common embedded component that allows users to search for and select names of people (individual users) and groups for which the portal is configured.

People Finder

The **People Finder** portlet provides both quick search and advanced search options for locating people and information about people. Once found, a person is visible to other users as a person link that indicates online presence and displays a menu of instant messaging and other options.

Hovering over the person link provides an option to display the person's business card (also known as person card). To view the profile, select **Profile** on the business card. For more information on the business card, see the section "Person card" in the People Awareness topic.

You can add the People Finder portlet to a page in the same manner that you add any portlet. People Finder is then immediately available when the page is rendered. If you add **People Finder** to a page, you must configure the profile parameter of that page before you add the portlet.

1. Turn **Edit Mode** on.

2. Click **Page**.
3. On the **General** tab, click **Edit Page Properties**.
4. Click the **Advanced** tab.
5. Scroll to the **Theme Settings** section.
6. Select `Basic Content with Dojo` from the **Profile** menu.
7. Click **Save**.
8. Add the **People Finder** portlet to the page.

The **People Finder** portlet appears when users click the **Find a User** link that appears at the end of any IBM WebSphere Portal Express page when the back-end server is J2EE.

People Finder is supported as a remote WSRP service.

People Finder supports the Java Portlet Specification. For more information, read the information about developing portlets.

For information about the **People Finder** portlet, view the help that is available when you select Configure from the drop-down menu on the portlet title bar.

Using the People Finder portlet

Users interact with the People Finder portlet through the following features, which appear in one or more views of the portlet:

- Quick Search and Quick Search Results, including Business Card fields
- Advanced Search and Advanced Search Results
- Profile page, including Business Card fields
- Organization view, including Business Card fields

Setting up People Finder

Use the configuration mode to specify which fields to display in the **People Finder** portlet. The fields that are available for selection when configuring the portlet are based on a mapping of attributes from the user repository to the Member Manager component of WebSphere Portal Express. People Finder requires a mapping for the `uid` and `mail` attributes. To learn more about attribute mapping or how to determine which attributes are defined to the portal, see the section "Adding more attributes to VMM."

For People Finder, make sure your site meets the following client and server requirements.

Client requirements

This portlet supports browsers capable of rendering HTML markup.

Table 110. Minimum requirements for HTML

Item	Description
Markup level	HTML 4.01 Transitional
Java applet	n/a
JavaScript	n/a
<iframe>	No

Table 110. Minimum requirements for HTML (continued)

Item	Description
Style sheets	Portal styles only
Software	The People Finder portlet is compatible with specific Web browser software releases. For details, see the *WebSphere Portal Express hardware and software requirements*.
Accessibility	Yes

Server requirements

There are no special server requirements for the People Finder portlet.

Deployment/Installation

This portlet is installed automatically as part of WebSphere Portal Express installation.

Configuration parameters

Config_ResultSetLimit is the only parameter that you can edit. Use the **Configure** command. Do not modify any configuration parameter values for the **People Finder** portlet in the **Portlet Management**. Click the **Administration menu** icon. Then, click **Portlet Management** > **Portlets**. Use this interface only for viewing parameters when troubleshooting portlet configuration.

Table 111. Configuration parameters

Parameter	Value
ConfigHelpURI	The location and name of the JSP containing topics launched from the Help command when configuring People Finder. Default: `/help/pfind_config.jsp`
SearchHelpURI	The location and name of the JSP containing topics launched from the Help command in the Profile page of the People Finder. Topics in this JSP include Help for person links, Quick Search, Quick Search Results, Advanced Search, and Advanced Search Results. Default: `/help/pfind_view_search_for_people.jsp`
PersonRecordHelpURI	The location and name of the JSP containing topics launched from the Help command in the Profile page and Organization view of the People Finder. Default: `/help/pfind_view_display_person_info.jsp`
Config_PageIndexLimit	Specifies the maximum number of search results to be returned per page by Quick Search and Advanced Search. Default: 5

Table 111. Configuration parameters (continued)

Parameter	Value
Config_RecursionSetLimit	Specifies the maximum number of recursions performed by search and retrieval on the *manager* attribute for a given person. The limit of recursions performed on the *manager* attribute determines the number of managers responsible for the found person that appear in the Organization View.
Config_ActiveFields **Config_Active_Queries** **Config_VisualSets** **Config_Views** **Config_QuerySets** **Config_QueryViews** **Config_UpdateInfo** **Config_Complete**	These parameters appear in the portlet after fields are specified in configuration mode. Do not modify these parameters.

Related concepts:

"Developing portlets" on page 2931
Get an overview of the process of creating portlets, learn about the concepts of the APIs used to develop portlets, and view the samples to get you started. Also, learn about integrating features such as single sign-on, cooperative sharing of information using the property broker, and migrating Struts applications to the portlet environment.

"People awareness" on page 959
People awareness makes people's names appear as hyperlinks that users can click to display information about the individual and gain access to actions for contacting and working with him or her. If the administrator has configured an IBM Sametime server to work with WebSphere Portal Express, users can see each other's online presence in their person links according to the status options they have set in their Sametime client (for example, whether the person is active, away, offline, or does not want to be disturbed). The person's online status appears only if Sametime is enabled.

Related tasks:

"Adding more attributes to VMM" on page 568
After you install IBM WebSphere Portal Express and configuring your LDAP user registries, you must adapt the attribute configuration to match the configured LDAP servers and your business needs. However, do not complete these steps if you configured only a database user registry or the default federated file-based repository for out-of-box installations.

Related information:

"System requirements" on page 103
Before you install IBM WebSphere Portal Express, review the hardware and software requirements to ensure that you have the supported versions of prerequisite and corequisite software and the required hardware.

People Finder configuration reference:

The default fields of the People Finder portlet correspond to attributes defined by Member Manager, and enable the display of information about people in several views. As an administrator, you can determine the layout and content of portlet views by selecting and ordering the fields that appear in each view.

Note: If Member Manager is configured with Microsoft Active Directory, remove the fields *carLicense*, *secretary*, and *pager* from the People Finder configuration.

The configuration settings are stored in the file `apPDirConfig.xml`. You can return People Finder to its default configuration by selecting **Configure** from the portlet drop-down menu, and under XML Configuration, clicking **Reload Settings**.

For additional information on setting up the views of the People Finder portlet, while configuring People Finder, select Help from the drop-down menu on the portlet title bar.

Business Card fields

The following table lists the Member Manager attributes that appear as fields in the unlabelled Business Card section of the Profile in People Finder. By default, the Business Card section displays the following fields as lines in descending order. Business Card fields do not display labels, and fields that contain no information are not displayed. For each attribute listed, the field label and display type are identified, except for Photograph, whose attribute you can select from a separate list and is *jpegPhoto* by default. Unless noted, the default display type of an attribute is a text string.

Table 112. Member Manager attributes and Business Card fields

Member Manager attribute	Field label	Display type
jpegPhoto	Photo	N/A
cn	Name	Person Link
telephoneNumber	Phone number	N/A
ibm-primaryEmail	Email address	Email Address
ibm-jobTitle	Job title	N/A
localityName	City	N/A

Contact Information fields

The following table lists the Member Manager attributes that appear as fields in the **Contact Information** section of the Profile. For each attribute listed, the field label and display type are identified. Unless noted, the default display type of an attribute is a text string.

Table 113. Member Manager attributes and Contact Information fields

Member Manager attribute	Field label	Display type
ibm-personalTitle	Title	N/A
cn	Name	Person Link
uid	User ID	N/A
employeeNumber	Employee number	N/A

Table 113. Member Manager attributes and Contact Information fields (continued)

Member Manager attribute	Field label	Display type
ibm-primaryEmail	Email address	Email Address
telephoneNumber	Phone number	N/A
mobile	Mobile phone number	N/A
pager	Pager Number	N/A
postalAddress	Address	N/A
roomNumber	Room number	N/A
street	Street	N/A
localityName	City	N/A
stateOrProvinceName	State or province	N/A
postalCode	Postal Code	N/A
countryName	Country	N/A

Current Job fields

The following table lists the Member Manager attributes that appear as fields in the **Current Job** section of the Profile. For each attribute listed, the field label and display type are identified. Unless noted, the default display type of an attribute is a text string.

Table 114. Member Manager attributes and Current Job fields

Member Manager attribute	Field label	Display type
ibm-jobTitle	Job title	N/A
departmentNumber	Department Number	N/A
businessCategory	Business	N/A
employeeType	Employee Type	Member Link

Background fields

The following table lists the Member Manager attributes that appear as fields in the **Background** section of the Profile. For each attribute listed, the field label and display type are identified. Unless noted, the default display type of an attribute is a text string.

Table 115. Member Manager attributes and Background fields

Member Manager attribute	Field label	Display type
preferredLanguage	Preferred language	N/A
labeledURI	Personal web page	Web Page Link

Advanced Search Queries

Advanced Search criteria, by default, include most of the same fields that are available for the Profile. In general, only fields corresponding to attributes of data type *String* are available as search criteria. Attributes of data type *Numeric*, *Object*, or *MemberLink* are not available for search.

The following fields are not available for selection as Advanced Search criteria in the People Finder. Users cannot search for people using these fields because the attributes are not type String.

manager (data type = MemberLink)

seeAlso (data type = MemberLink)

secretary (data type = MemberLink)

Advanced Search Results fields

The following table lists the Member Manager attributes that appear as fields in the Advanced Search Results in People Finder. For each attribute listed, the field label and display type are identified. Unless noted, the default display type of an attribute is a text string.

Table 116. Member Manager attributes and Advanced Search Results fields

Member Manager attribute	Field label	Display type
cn	Name	Person Link
telephoneNumber	Phone number	N/A
ibm-jobTitle	Job title	N/A

Quick Search Queries

The following table lists the Member Manager attributes that appear by default in the **Search By** list in Quick Search. For each attribute listed, the field label and display type are identified. Unless noted, the default display type of an attribute is a text string.

Table 117. Member Manager attributes and Search By fields

Member Manager attribute	Field label	Display type
byName	Name	N/A
uid	User ID	N/A
ibm-primaryEmail	Email address	N/A
telephoneNumber	Phone number	N/A
ibm-jobTitle	Job title	N/A

Quick Search Results fields

The following table lists the Member Manager attributes that appear as fields in the Quick Search Results. For each attribute listed, the field label and display type are identified. Unless noted, the default display type of an attribute is a text string.

Table 118. Member Manager attributes and Quick Search Results fields

Member Manager attribute	Field label	Display type
cn	Name	Person Link
telephoneNumber	Phone number	N/A

Organization View fields

The following table lists the Member Manager attributes that appear as fields in the Organization View. For each attribute listed, the field label and display type are identified. Unless noted, the default display type of an attribute is a text string.

Table 119. Member Manager attributes and Organization fields

Member Manager attribute	Field label	Display type
cn	Name	Person Link
telephoneNumber	Phone number	N/A
ibm-jobTitle	Job title	N/A

Paginating search results in the People Finder:

You can configure the People Finder portlet to display a maximum number of results found per page.

Before you begin

About this task

Procedure

1. Select **Configure** from the People Finder drop-down menu.
2. Click **Configuration Basics**.
3. Type the value you want in the **Maximum number of items shown on a page in search results** field.
4. Click **OK**.
5. Click **Apply Changes Now**.

Example

What to do next

Enabling People Finder for anonymous users:

If you grant access to anonymous users for any page that contains the **People Finder** portlet, you must also grant them access to a hidden page, and to a dynamic person tag portlet that support awareness in **People Finder**. In addition, you must enable session use for anonymous users.

About this task

Perform the following steps:

Procedure

1. Log in to the portal as an Administrator.
2. Click the **Administration menu** icon. Then, click **Access** > **Resource Permissions**.
3. In the **Resource Types** list, click **Pages**.
4. Use Search to locate the page in your site where you added the **People Finder** portlet.
5. Click the key icon under **Assign Access**.

6. For the role User, click the pencil icon, and then click**Add**.
7. Select **Anonymous Portal User**, and then click **OK** as necessary to save the change.
8. Use Search to locate the **People Finder** portlet.
9. Repeat steps 6 - 8 to assign access for the portlet.
10. Click **Logout**.
11. Set the following property in the `WP NavigatorService` from the WebSphere Integrated Solutions Console. For more information, see the topic on setting configuration properties, and the Navigator Service section in the topic on Portal configuration services.

    ```
    public.session = true
    ```
12. Add the anonymous portal user to the user role on the USERS Virtual Resource:
    a. Click the **Administration menu** icon. Then, click **Access** > **Resource Permissions**.
    b. Under Resources Types, select **Virtual Resources**, locate **USERS** in the Resources list and then click the **Assign Access** icon.
    c. For the User role, click the **Edit Role** (pencil) icon.
    d. Click **Add**, then select **Anonymous Portal User** and click **OK** to save the change.

People Finder interaction with other portlets:

Other portlets can communicate with People Finder using URL addressability.

The Person tag uses URL addressability to communicate with the People Finder if People Finder is not on the current page. Other collaboration portlets can use URL addressability to programmatically integrate People Finder features. Portlets can display a link for the **Profile** action and thereby access People Finder functionality in their application contexts. Clicking **Profile** displays the Profile inside the People Finder.

Table 120. Parameters and values for implementing URL actions in portlets

URL Parameter	Possible Values
URLAction	URLShowPersonRecord, URLShowOrgView
URLMemberID	Member Manager memberUniqueIdentifier of the person whose Profile or Organization View is to be displayed. Alternatively, the following Member Manager attributes: • memberDN • any valid input for *byName* query • a valid entry of ibm-PersonAwarenessIdentifier attribute.

Note: In IBM WebSphere Portal Express, People Finder is a standard portlet and therefore does not support cooperative portlet methods.

Member Manager and People Finder:

Member Manager is the component of IBM WebSphere Application Server and IBM WebSphere Portal Express that provides the common schema of attributes used by People Finder for people and their Profile pages.

Prerequisites

To understand how Member Manager works, you should have a thorough understanding of WebSphere Portal Express security and authentication concepts, including user registries and user repositories.

Member Manager data types

You can select the display format of a People Finder field if the data type of the corresponding attribute in Member Manager is *string*. Attributes with data type *string* can have one of the following display formats:

- String (default)

  The People Finder field appears as text.
- Person link

  The People Finder field appears as a person name showing online presence and a person menu of actions.
- Email address

  The People Finder field appears as a mailto: link that launches an email message to the person.
- Web page link

  The People Finder field appears as a link to a web page

People Finder fields that correspond to Member Manager attributes that have data types other than *string* appear as fixed text. People Finder supports the following data types, but you cannot choose the display format for fields that correspond to attributes with these data types.

- Integer, Long, Double

  Display format is always Numeric.

  Examples:

  ibm-firstDayOfWeek

  ibm-firstWorkDayOfWeek
- MemberIdentifier

  Display format is always Member Link.

  Examples:

  manager (Employee's manager. Required to build Organization view.)

  secretary (Name of the person's secretary or assistant)

  seeAlso (Person who can be contacted when this person is not available)
- ByteArray

  Display format is always Image.

  Example: *jpegPhoto* (A jpeg format photograph)
- Object

  Display format is always Object.

Other data types such as Timestamp are not supported by People Finder.

People Finder attribute mapping:

The collaboration integration relies on a predefined set of Virtual Member Manager (VMM) user and group attributes to function properly, while your LDAP server may use a different set of predefined user and group attributes. If a portal attribute

is available under a different name on the LDAP server, you can map the portal attribute to the corresponding LDAP attribute. If you want to use an attribute as search attribute or you want to see its value in the search result, you must to map the attribute. Portal attributes that do not correspond to an LDAP attribute should be flagged as unsupported.

Before you begin

The collaboration products use these attributes:

- **displayName**
- **ibm-primaryEmail**
- **givenName**
- **dn**
- **cn**
- **sn**
- **uid**

About this task

Mapping the following attributes ensures that users see the appropriate values when People Finder displays an individual's User Profile, or when they search by attribute (for example, to find all users who have the same preferred language or whose telephone number starts with the same area code).

```
# name of the portal attribute
federated.ldap.attributes.mapping.portalName=ibm-primaryEmail,ibm-jobTitle,
stateOrProvinceName,countryName,localityName,street,employeeNumber,roomNumber,
preferredLanguage,labeledURI,ibm-personalTitle

# name of the LDAP attribute
federated.ldap.attributes.mapping.ldapName=mail,title,st,c,l,OfficeStreetAddress,
EmployeeID,physicalDeliveryOfficeName,preferredLanguage,url,
personalTitle
```

For more information, see *Adding more attributes to VMM*.

Directory Search

The Directory Search or "People Picker" portlet is a common embedded component that allows users to search for and select names of people (individual users) and groups for which the portal is configured.

Directory Search is different from People Finder in two ways:

- Users use Directory Search as part of a task, such as sending a document link to other people with access to the same document library. Users can search for user or group names and select them. Directory Search then tries to find the selection so that it can be used in the next step of the larger task.
- Users can search for groups and people.

Directory Search is supported as a remote WSRP service, and is available from all calling applications.

Using the Directory Search portlet

Directory Search opens in a window that is set to use the "no skin" skin so that it displays without a title bar. The portlet provides common controls for searching for people and groups.

Table 121. Portlet controls

Control	Description
Search for	The field where users enter all or part of a name for retrieval from the directory. Users can search for person names only, group names only, or person names and group names.
Search results	The list of names (persons and groups) that match the search text. By default, the application context and the type of directory determine the information that is displayed in the columns of the search results list. You can configure the search results to show information in 3 columns. Users can use the third column to distinguish between people whose name information in the first and second columns is identical.
Show details	Displays profile information for the selected name. If the search results include multiple users with the same displayed name, viewing a selected name's profile lets you check whether it is the one you want. **Tip:** As an alternative, you can hover over the selected name to see the profile information.

Starting Directory Search Portlet

Starting the Directory Search portlet requires the use of the IBM WebSphere Portal Express urlGeneration tag. This tag creates a URL to the Directory Search portlet. The following parameters are used with urlGeneration tag. This tag uses **`openModalDialog`** to pass a callback method, which is called when the portlet is closed, and contains the resulting people or groups that were selected:

- contentNode="PeopleConstants.IBM_PORTAL_DIRECTORY_SEARCH_PAGE"
- compositionNode="PeopleConstants.IBM_PORTAL_DIRECTORY_SEARCH_CONTROL"
- portletWindowState="solo"
- newWindow="true"

The parameters that are mentioned in the following table are used with the urlGeneration tag. The **`Lable`** and **`buckets`** parameters are required, while the other parameters are optional. Except for the following parameters, the urlGeneration tag accepts customized parameters.

Table 122.

Parameter	Possible values	Description
Label	default when parameter not specified: `Selected names:` `picker.mail.label=Recipients:` `str.calendar.label=Recipients:`	Resource key for the text string that the calling portlet requires; these resource keys are stored in the people picker property file. For example, the default is `Selected names` but the Mail portlet needs it to be `Recipients`.
dirs	default when parameter not specified: `WMM` `WMM=Organization Directory Adapter`	A comma delimited list that consists of the directory adapters that the picker should search.
searchScope	`peopleOnly` `groupsOnly` `all` If parameter is not specified, people and groups are searched (all)	Determines whether you can search for people only, groups only, or both people and groups. Also determines whether the **Show Group Member** button is displayed. If the value for this parameter is either `groupsOnly` or all, the **Show Group Member** button is displayed.
requireEmail	`true/false`	Determines whether the user can select the name of a person who does not have an email address. If the value for this parameter is true and the user tries to select a person who does not have an email address, an error message

Table 122. (continued)

Parameter	Possible values	Description
buckets	default when parameter not specified: Only one Search results list box is displayed, and the user can select only 1 name in it. `picker.mail.bucket1,picker.mail.bucket2,` `picker.mail.bucket3=To,cc,bcc`	The value is `0` or a comma-delimited list of resource keys for the text strings that the calling portlet requires. These resource keys are stored in the people picker property file. Determines the following: Whether the user can select only 1 name or more than 1 name. Whether a second list box is displayed. If the user can select multiple names, a second list box that holds the selected names is displayed. If the user can select only one name, the second list box bucket is not displayed. The number of **Add** buttons that are displayed, and the number of **buckets** in the second list box. Example: the default is one button that is labeled **Add** and no **buckets** in the second list box, but the Mail portlet has three buttons and buckets: To, cc, and bcc.

Setting up Directory Search

For Directory Search, make sure that your environment meets the following client and server requirements.

Note: The ability to search for users and groups in the Directory Search portlet requires the USER role on the USERS and USER GROUPS virtual resources.

Client requirements

This portlet supports browsers capable of rendering HTML markup. The following table provides detailed information.

Table 123. Minimum requirements for HTML

Item	Description
Markup level	HTML 4.01 Transitional
Java applet	No
JavaScript	Yes
<iframe>	No
Style sheets	Portal styles only
Software	The Directory Search portlet is compatible with specific web browser software releases. For details, refer to the *WebSphere Portal Express hardware and software requirements*.
Accessibility	Yes

Server requirements

There are no special server requirements for the Directory Search portlet.

Deployment/Installation

This portlet is installed automatically as part of WebSphere Portal Express installation.

Note: This portlet is supported as a remote WSRP service.

Configuring Directory Search

When you are searching for people and groups, Directory Search searches the repositories that are defined in the Member Manager component of the portal.

These repositories can include the default file repository, LDAP user registry, property extension database, database user registry, or custom user registry. For more information about configuring Member Manager, see the topic on User registry considerations.

The directory applies the `byName` query that is defined in Member Manager when retrieving person or group names. The `byName` query uses *cn* when it is retrieving a group name, and at least one of the following attributes when it is retrieving a person name:

- *cn*
- *givenName*
- *sn*
- *DisplayName*

For detailed information, refer to the attribute definition and mapping files and the configuration property files that are described in Member Manager documentation.

"Configuring display attributes in the directory search portlet" on page 942
You can configure the display attributes in the directory search dialog.

"Configuring search attributes in the directory search portlet" on page 942
You can configure the search attribute in the directory search dialog.

"Configuring the wildcard support in directory search queries" on page 943
You can use the wildcard character (*) in directory search queries.

Related information:

WebSphere Portal detailed system requirements

Changing the number of search results found:

By default, Member Manager limits the number of entries that are returned to 50 when you use the Directory Search window to search the organization directory for names. By modifying the `pickersettings.properties` file, you can change the number of search results returned.

Procedure

1. Edit the `pickersettings.properties` file in the following location:
   - Windows: `wp_profile\PortalServer\config\lotusworkplacelib\pickersettings.properties`
   - Linux : `wp_profile/PortalServer/config/lotusworkplacelib/pickersettings.properties`
   - IBM i: `wp_profile/PortalServer/config/lotusworkplacelib/pickersettings.properties`
2. Change the 50 at the end of the following content to the number of search results you want to display:

   ```
   people.view.1
   =WMM.1,com.ibm.workplace.people.picker.data.value.service,
   str.directory.name.wmm,com.ibm.wkplc.people.picker.workspace.
   WmmPickerAdapterFactory,50,2
   ```
3. Save the properties file.
4. Restart the IBM WebSphere Portal Express server.

Changing the minimum number of characters in names for searching:

By default, Member Manager requires users to enter a minimum of two characters when they use the Directory Search window to search the organization directory for names. This limit is not appropriate for all languages. By modifying the `pickersettings.properties` file, you can reduce the limit to one character.

Procedure

1. Edit the `pickersettings.properties` file in the following location:
   - Windows: `wp_profile\PortalServer\config\lotusworkplacelib\pickersettings.properties`
   - Linux : `wp_profile/PortalServer/config/lotusworkplacelib/pickersettings.properties`
   - IBM i: `wp_profile/PortalServer/config/lotusworkplacelib/pickersettings.properties`
2. Change the 2 at the end of the following content to a 1:

   ```
   people.view.1
   =WMM.1,com.ibm.workplace.people.picker.data.value.service,
   str.directory.name.wmm,com.ibm.wkplc.people.picker.workspace.
   WmmPickerAdapterFactory,50,2
   ```
3. Save the properties file.
4. Restart the IBM WebSphere Portal Express server.

Configuring display attributes in the directory search portlet:

You can configure the display attributes in the directory search dialog.

Procedure

1. Log on to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Find and click the **WP PeopleService** resource environment provider. The resource environment provider name is case sensitive.
4. Click **custom properties**.
5. Click the custom properties page and configure the values for following property:

   pickerDisplayAttribute

   Governs the display attributes for the Portal People Picker UI.

   Use two attributes at a time, supported by both users and groups.

   default value : `cn,displayName`
6. Restart IBM WebSphere Portal Express server to reflect the changes.

Configuring search attributes in the directory search portlet:

You can configure the search attribute in the directory search dialog.

Procedure

1. Log on to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Find and click the **WP PeopleService** resource environment provider. The resource environment provider name is case sensitive.

4. Click **custom properties**.
5. Click the custom properties page and configure the values for the following properties:

   pickerPeopleSearchAttribute
   Governs the Portal People Picker search attributes for a user. Use four attributes at a time.

   default value : `cn,displayName,sn,givenName`

   pickerGroupSearchAttribute
   Governs the Portal People Picker search attributes for a group. Use one attribute at a time.

   default value : `cn`
6. Restart IBM WebSphere Portal Express server to reflect the changes.

Configuring the wildcard support in directory search queries:

You can use the wildcard character (*) in directory search queries.

Procedure

1. Log on to WebSphere Integrated Solutions Console.
2. Go to **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Find the **WP PeopleService** resource.
4. Click the **WP PeopleService** resource link.
5. Click **Custom properties**.
6. Configure the value for the following property:

   pickerWildCardSearchEnable
   Controls whether to append the wildcard character (*) to search query or not. The default value is `true`.
7. Restart the IBM WebSphere Portal Express server to reflect the changes.

Planning for collaborative servers and portlets

Setting up a site with IBM Domino integration requires decisions about user directories, security, authentication, and performance. View some use cases that may help you make decisions leading to a successful Domino integration.

Performance considerations

When integrating Domino into your portal environment, consider performance when deciding how many and which servers you need.

For example, to use a Domino LDAP server as the user directory (repository) for the portal, install portal on a separate machine from the Domino LDAP server configured to support collaborative features in the portlets. The Domino LDAP server for the portal user directory should reside on a machine that is dedicated to serving the portal environment and all its users.

IBM i: It is recommended that a specific Domino server be created to run the collaborative components, and that it should reside on the same IBM i server as WebSphere Portal Express.

"Domino server installation and server setup reference" on page 944
View a roadmap to instructions for installation and first set up of an IBM

Domino server, including installation of the Domino Administrator client software that you can use to administer the server.

"Planning names for servers and users in a Domino site" on page 946
While installing and integrating Domino and collaboration products, you must decide on a naming scheme for your servers, plan for and create several important administrative users, and create passwords for those users. Learn about server naming and a table of identities you can use while installing and configuring servers, with recommendations for which names need to match for better performance.

"Determining the needs of your portal site" on page 949
The following general use cases are intended only to provide some recommendations for your decisions about directories when integrating collaboration product.

"Platform and user directory considerations" on page 951
There are unique considerations for operating systems that you should be aware of before you proceed with integrating collaboration software with WebSphere Portal Express. Also, in the context of collaboration integration, there are two types of user directories, Domino directory and then all other supported user directories.

"Security and user authentication considerations" on page 952
Correctly configuring authentication and security includes configuring single sign-on and setting up SSL.

Domino server installation and server setup reference

View a roadmap to instructions for installation and first set up of an IBM Domino server, including installation of the Domino Administrator client software that you can use to administer the server.

About Domino servers

A Domino server has several purposes in an IBM WebSphere Portal Express site. It can be a Domino Directory (LDAP) user registry server, a messaging/application server housing source data for Domino-based portlets. Or it can be a prerequisite server on an IBM Sametime server.

Because WebSphere Portal Express is compatible with many releases of Domino server software, this topic contains no step-by-step procedures for installation. Instead, you can access the following links to locate the correct procedure for your platform and for the release of Domino you intend to install. Documentation for specific releases and platforms is available over the Internet on the IBM website, for download and browsing. For information about the releases of Domino supported for use with WebSphere Portal Express, see the detailed system requirements.

The installation documentation covers the following basic tasks to complete a Domino server installation:

- Install the Domino server software.
- Run the Domino Server Set up program (with a wizard).
- Install the Domino Administrator client software. You can use this software for important tasks on the server.

  Tip: You can also want to install the Notes Client and Notes Designer clients for email and application development tasks.
- Set up the Domino Administrator client software (with a wizard).

- If this server is the first Domino server in your portal site, register ids for subsequent servers.

Usage restrictions: You are authorized to install and use Domino Directory solely and exclusively in connection with your use of Sametime. Consult the product license for details.

Where to find installation procedures

The IBM Lotus Documentation website lists all documentation for all releases of Domino.

The following table lists installation documentation pieces for the Domino 8.5.2 release. The documentation website is subject to change, so if any of these pieces are unavailable, check the Domino product pages for updates for your platform.

Table 124. Installation documentation

Platform	Documentation	Important topics	Comments
Windows	Lotus Domino 8 Administrator Help	Under *Installation*, *Server installation*,Installing Domino on Windows systems, and The Domino Server Set up program.	You can either view this documentation online or download a file.
Linux	Lotus Domino 8 Administrator Help	Under *Installation*, *Server installation*,Installing Domino on UNIX systemsInstalling Domino on Linux on zSeries systems, and The Domino Server Set up program.	You can either view this documentation online or download a file.
IBM i	Installing and Managing Domino 8 for System i	*Chapter 3 Installing Domino on your system*, *Chapter 6 Setting Up a First Domino Server*, and *Chapter 8 Setting Up an Additional Domino Server*.	You can either view this documentation online or download a file.

Critical choices for Domino in a portal site

The following table lists choices common to Domino installation on most or all platforms, with recommendations for use of Domino with the portal. Because specific installation steps differ between platforms, the descriptions of the choices are general. Do not expect to see the exact wording of the choice when installing for your platform and release. The options highlighted in this table are a subset of all choices you make during installation; in general, if not specified, accept default options.

Before installation, see the topic on planning for collaborative servers and portlets, which describes common types of collaborative portal sites, and includes considerations for platform, user directories, security, and performance.

Tip: Be sure to keep a record of all names and passwords you specify, and all directories where the installation programs installation software.

Table 125. Key decisions for Domino installation

Procedure	Decision point	Recommendations
Domino Server Set up program	First server versus additional server	Your Domino LDAP server must be a first server, and messaging/application servers or underlying Domino servers for Sametime are additional.
Domino Server Set up program	Organization name and certifier	See "Planning names for servers and users in a Domino site." Use the certifier id you create on the first Domino server to register subsequent Domino servers in the portal site.
Domino Server Set up program	Administrator name	See "Planning names for servers and users in a Domino site."
Domino Server Set up program	Services (servers)	Enable HTTP (or web browsers) and DIIOP on all Domino servers in your site, and LDAP on the Domino LDAP server. The DIIOP service can be with a Customize option in the server setup program.
Domino Administrator client software setup (wizard)	Administrator name	Use the same name you did for Domino server setup. See "Planning names for servers and users in a Domino site."

After you install and set up the first Domino server in your site, you must create certifier IDs to register subsequent Domino servers. For information, see the Lotus Domino 8 Administrator Help topic on server registration.

Related information:

IBM Support

IBM Publication Center - IBM Lotus Domino 7.0.2 Release Notes

WebSphere Portal detailed system requirements

Lotus Domino Administrator Help, Server registration

Planning names for servers and users in a Domino site

While installing and integrating Domino and collaboration products, you must decide on a naming scheme for your servers, plan for and create several important administrative users, and create passwords for those users. Learn about server naming and a table of identities you can use while installing and configuring servers, with recommendations for which names need to match for better performance.

About server naming:

When your portal site contains multiple IBM Domino servers, their names should be reasonably short and should contain no spaces. The server names will be seen by your users, so make them descriptive when possible. If the Domino server name is not the same as the physical server name, you must make sure that the name is resolvable through DNS. For example, you could name a hub server in Chicago `acmehub`, which, combined with the server's domain, could yield a fully qualified host name of `acmehub.chicago.acme.com`. You would configure an entry in DNS for `acmehub.chicago.acme.com` to point to the physical IP address of the server. It is not a requirement to make the Domino server name the same as the physical server name, but if it is not, Server Connection documents are required in all other Domino servers and the Lotus Notes or Domino Administrator client software running on them.

Table of user identities

Some names in the following table are specified during the Domino Server Setup procedure you perform after installing the Domino LDAP server for the first time. If you later perform an upgrade installation, configuration choices from the first-time Server Setup will be used; you will not see screens presenting these choices again.

Table 126. User identities

Identity	Description	Recommendation
Organization name for a Domino LDAP server	This name is specified during the Domino Server Setup procedure.	See *About server naming*. Example: `dom_hub/chicago/renovationscorp`
Administrator of a Domino LDAP server	This administrative user identity is created during the Domino Server Setup procedure.	**Tip:** For convenience, you could make this the same as the user name of an administrator in WebSphere Portal Express (**`PortalAdminId`** in the `wkplc.properties` file)Example: `wpsadmin`
Bind user OR IBM WebSphere Application Server administrator OR both	This identity is used by IBM WebSphere Portal Express to access the LDAP directory. Both LDAP directory and security configuration for WebSphere Portal Express involve modifying values in the `wkplc.properties` file. This user identity is created during the Server Setup procedure.	Should be the same as the user ID of an administrator for the WebSphere Application Server (**`WasUserID`** in the `wkplc.properties` file)Example: `wpsbind` For information on how to create the Bind user or Admin user, use Search to find the topic on preparing a Domino Directory server on <*platform*> (for example, Preparing a Domino Directory server on Windows).

Table 126. User identities (continued)

Identity	Description	Recommendation
WebSphere Portal Express administrators group	You should manually edit the group `wpsadmins` to `wpsadmins/yourorgname`. This edit creates a fully distinguished LDAP name of `cn=wpsadmins/o=yourorgname`. This change must made when using a Domino LDAP directory, because Domino does not store groups in the hierarchical format that WebSphere Portal Express expects.	Should be the same as the group name of an administrator for all administrators for the WebSphere Portal Express server (**`PortalAdminGroupId`** in the `wkplc.properties` file)Example: `wpsadmins/renovationscorp` **Note:** In the ACL of the Domino Directory this group should have Author or Editor access, and the Role Types. These settings allow the administrator group to write and edit Person documents in the Domino Directory; these are necessary tasks in a portal that uses subscriber management.
Sametime server administrator	This user name has administrative access to the Sametime server and can modify Web pages on the server.	Example: `stadmin`
Sametime Web Conferencing administrator	This user name is created in the Domino Directory (`names.nsf`) on the Sametime Web Conferencing server and is used only for integration of Sametime. **Note:** In the ACL of the `STConfig.nsf` database, this user name is a Person/Manager, and has, at minimum, role(s) equivalent to those specified for the servlet entry of the meeting API in the `servlets.properties` file on the Domino server.	Example: `st_webconf_admin` Recommended: At least the [SametimeAdmin] role

Related tasks:

"Preparing a Domino Directory server" on page 159
If you plan to use a Domino Directory as an LDAP user registry, you must install and set up the server so that it communicates with IBM WebSphere Portal Express.

Related information:

Domino Administrator Help, Server document - Security tab

Domino Administrator Help, Server registration

Determining the needs of your portal site

The following general use cases are intended only to provide some recommendations for your decisions about directories when integrating collaboration product.

To identify your use case, ask yourself two questions about your site:

1. What directory service are you already using, or do you want to use, for the user directory for WebSphere Portal Express? Possible answers are:
   A: LDAP directory other than Domino
   B: Domino LDAP directory
2. What directory service are you already using, or do you want to use, for the user directories for collaboration products, such as IBM Sametime? Possible answers are:
   A: LDAP directory other than Domino
   B: Domino LDAP directory
   C: Native (non-LDAP) Domino directory

Table 127. Use case: 1A+2A

Single directory (LDAP other than Domino) site	**Your decisions about Domino integration**
• You have WebSphere Portal Express installed and in active use. • Your portal site is configured with an LDAP directory other than Domino (for the purposes of these scenarios, assume IBM Directory Server, but any other LDAP has the same considerations) with a substantial user repository in active use. • You intend to integrate collaborative portlets • You want the Domino portlets to have online awareness features. • You want users to be able to work in portlets without authentication other than logging into the portal (that is, you need the single sign-on feature). In fact, you may already have single sign-on enabled on your portal server. • You do not yet have any collaboration products or servers installed, or if you have them, they are not yet configured for use with the portal. **Note:** If you have an existing Domino server you intend to integrate, make sure that its release is supported before you attempt to use it with the portal. See the Software for collaboration section in the *WebSphere Portal Express hardware and sofware requirements.* If the release is not supported, you must upgrade the Domino server before you can use it with the portal.	Your environment is typical of most portal customers. You must install and set up a Sametime server to support awareness. We recommend that you configure the server to authenticate against the LDAP directory already configured with your portal site. To enable single sign-on, configure it as a last task after installing and configuring new servers for collaboration products, to include all the new servers. Support for key features in the collaborative portlets such as auto-detection of users' mail files requires additional configuration in this environment.

Table 128. Use case: 1B+2B

Single directory (Domino LDAP) site	Your decisions about Domino integration
• You have installed WebSphere Portal Express • You have no LDAP user directory configured yet. • You intend to integrate collaborative portlets • You want the portlets to have online awareness features, and you want users to be able to work in portlets without authentication other than logging into the portal (that is, you need the single sign-on feature)	Your environment is recommended, especially for new portal sites, if you intend to make full use of Domino integration. Install and configure Domino as your LDAP directory for the portal. It is a best practice to use the directory configured for Sametime as the directory configured for the portal, and Domino LDAP is the best choice for Sametime; therefore, in a new site we recommend using Domino LDAP as the single directory.

Table 129. Use case: 1A + 2B

Dual directory-type site (LDAP other than Domino for portal with Domino LDAP for Sametime user directory)	Your decisions about Domino integration
• You already have a mature installation of Domino servers including Sametime or iNotes. Your Domino servers are upgraded to a release supported by WebSphere Portal Express. • You have newly installed WebSphere Portal Express or have the intention to deploy it. You may even have a mature portal site, but have not yet attempted to integrate it with your Domino installations. • You intend to integrate collaborative portlets, especially messaging portlets to support your existing Domino mail and calendar users. • You want the portlets to have online awareness features (your Domino users are accustomed to Sametime instant messaging), and you want users to be able to work in portlets without authentication other than logging into the portal (that is, you need the single sign-on feature).	Your environment is typical of many portal customers who have investments in both directories that must be maintained. See the following topics for tasks specific to reconciling directories: • "Auto-detecting user mail information from a secondary LDAP server" on page 964 • "People awareness" on page 959

Table 130. Use case: 1A + 2B + 2C

Multiple directory-type site (LDAP other than Domino for portal with a combination of other directories, most likely native Domino directory for Sametime	**Your decisions about Domino integration**
• You already have a mature installation of Domino servers including Sametime or iNotes. Your Domino servers are upgraded to a release supported by WebSphere Portal Express. • You have newly installed WebSphere Portal Express or have the intention to deploy it. You may even have a mature portal site, but have not yet attempted to integrate it with your Domino installations. • You have a native Domino Directory (non-LDAP) in active use. Sametime uses a native Domino Directory. • You intend to integrate collaborative portlets, especially messaging portlets, to support your existing Domino mail and calendar users. • You want the portlets to have online awareness features (your users are accustomed to Sametime instant messaging), and you want users to be able to work in portlets without authentication other than logging into the portal (that is, you need the single sign-on feature).	Your environment is typical of many customers with mature Domino installations and an investment in an extensive native Domino directory who want to integrate portal. To support SSO, you must reconcile authentication between user identifications in your native Domino directory and the portal LDAP directory. See the following topics for tasks specific to reconciling directories: • "Auto-detecting user mail information from a secondary LDAP server" on page 964 • "People awareness" on page 959

Related information:

WebSphere Portal hardware and software requirements

Platform and user directory considerations

There are unique considerations for operating systems that you should be aware of before you proceed with integrating collaboration software with WebSphere Portal Express. Also, in the context of collaboration integration, there are two types of user directories, Domino directory and then all other supported user directories.

Platform considerations

Depending upon platform, Domino servers in your environment have slightly different task and/or registry requirements:

- **All platforms:** Domino IIOP is used to pre-populate drop-down lists shown when users personalize the collaborative portlets.
- Windows: Any Domino data source servers must have HTTP, LDAP, and Domino IIOP enabled.
- Linux: Any Domino data source servers must have HTTP, LDAP, and Domino IIOP enabled.
- IBM i: Any Domino data source servers must have HTTP and Domino IIOP enabled, and must use an LDAP user registry.

User directory considerations

Directory considerations for Domino LDAP:

From the portal perspective, there are two types of Domino servers: the Domino server as a user repository (Domino Directory server as an LDAP server), and any Domino server that acts as a Domino data source for portlets: such a server is called a messaging/application server.

Because WebSphere Portal Express supports the use of Domino Directory as an LDAP server, you can set up the portal to use a Domino server as the user repository for users who access both the portal and any portlets that access Domino and collaboration products.

You can use a Domino server with LDAP enabled both as the user repository for the portal and for auto-detection of users' mail files, unless your portal user repository is so large that you want to use separate machines for performance reasons (see *Performance considerations*).

Directory considerations for Sametime :

If you will be using portlets for Domino and Sametime, the Sametime user directory can be any supported LDAP (including Domino) directory, or a native Domino directory. But it is recommended that Sametime use the same directory as the one configured for the portal, to avoid the additional configuration necessary to support both directories.

Security and user authentication considerations

Correctly configuring authentication and security includes configuring single sign-on and setting up SSL.

About security through SSL and other features

Whether your site includes single, dual, or multiple types of user directories, SSL is recommended, and you enable it the same way.

If your site will use IBM Security Access Manager or Computer Associates eTrust SiteMinder for additional security, set up such protection on servers in the following order: WebSphere Portal Express, Sametime, and then Domino servers. In addition, if you use eTrust SiteMinder, portlets such as Lotus Notes View will be unable to take advantage of features supported by DIIOP.

If your site will use Security Access Manager or another reverse proxy, or a load balancer, when installing Sametime, select the option "Allow HTTP Tunneling on a Sametime server with a single IP address." With this option selected, all Sametime client data, except A/V data, is tunneled to the Sametime server via HTTP on port 80. You also may need to enable this option if Sametime clients must connect to the server through a network that blocks TCP communications on ports 8081 and 1533.

About user authentication through Single Sign-On (SSO)

Single sign-on between the Domino environment and the portal environment allows users to log in to the portal, and then work in any of the collaborative portlets without having to authenticate a second time. Although enabling single

sign-on is not required to use all the collaborative portlets, it is strongly recommended as a way of improving the user experience. Lotus Notes View and iNotes require single sign-on support.

To support single sign-on, a Web SSO configuration document must exist for each Domino domain that includes Domino servers. The Web SSO configuration document is a domain-wide configuration document stored in the Domino Directory. This document, which you can replicate to all servers participating in the single sign-on domain, is encrypted for participating servers and administrators, and contains a shared secret key used by servers for authenticating user credentials.

In addition to the Web SSO configuration document for Domino servers, you must create, save, and export an LTPA key from WebSphere Application Server, and then import that WebSphere LTPA key into the Domino domain or domains. For each Domino domain that is set up for use with the portal, the same WebSphere LTPA key must be imported to support single sign-on. Verify that automatic LTPA key generation is disabled on each node of the single sign-on domain.

A best practice is to install and configure all servers prior to enabling single sign-on. For example, install and configure Sametime before you enable single sign-on.

If you complete the required single sign-on configuration between the Domino environment and portal environment, there is no procedure to disallow automatic login for a specific user. For example, if user A logs in to the portal, user A will always be logged in to the Domino environment.

Tip: Managing Single Sign-On and awareness when there are multiple types of directories. If there is an LDAP directory server other than Domino in place, for example IBM Directory Server, you could employ several strategies to integrate it with a native Domino Directory and therefore achieve single sign-on (SSO) and awareness across any collaborative portlets your organization uses. The Domino Directory Assistance functionality may provide a solution for name mapping across LDAP directories. Even when your organization, as a matter of policy, manages modifications primarily through an existing non-Domino LDAP directory, schema in the non-Domino directory can be customized and then work in concert with Directory Assistance, which can manage the name mapping for collaborative applications. For a number of creative multi-directory solutions, including information on supporting single-sign on for awareness through the Sametime servers if your organization uses it, see the IBM developerWorks article *Single Sign-on in a Multi-Directory World.*

Related information:

Single Sign-on in a Multi-Directory World

Integrating with IBM Sametime

Configuring the Sametime Community Server and Sametime portlets requires a set of tasks performed on both the Sametime Community Server and the portal server.

About this task

Important: Configuring a Sametime Community Server with secure socket layers (SSL) is not recommended when using a multiplexer server to manage traffic on the Sametime Community Server. If you configure SSL on the multiplexer,

WebSphere Portal Express will no longer be able to communicate through the multiplexer. For additional information, see Technote 1086354.

This topic lists the required procedures for configuring the Sametime Web 2.0 Contact List portlet to work with both the Sametime Community Server and the portal server:

Unless otherwise noted, all the procedures are required for all portlets.

1. "Sametime server installation reference"
   View a roadmap to instructions for installing an IBM Sametime server.
2. "Configuring Sametime Proxy" on page 957
   If IBM WebSphere Portal Express and IBM Sametime are both authenticating with the same LDAP server, SSO configuration is simple.
3. "People awareness" on page 959
   People awareness makes people's names appear as hyperlinks that users can click to display information about the individual and gain access to actions for contacting and working with him or her. If the administrator has configured an IBM Sametime server to work with WebSphere Portal Express, users can see each other's online presence in their person links according to the status options they have set in their Sametime client (for example, whether the person is active, away, offline, or does not want to be disturbed). The person's online status appears only if Sametime is enabled.

Related information:

Technote 1086354

Sametime server installation reference

View a roadmap to instructions for installing an IBM Sametime server.

About the Sametime server

A Sametime server is an IBM Domino server that has the Sametime server software installed and set up on it.

IBM WebSphere Portal Express is compatible with several releases of Sametime server software. Use the following links to locate the correct procedure for your platform and for the release of Sametime you intend to install. Documentation for specific releases and platforms is available on the IBM website for download and browsing. For information about the releases of Sametime supported for use with WebSphere Portal Express, see the *IBM WebSphere Portal Express detailed system requirements.*

The installation documentation covers the following basic tasks that you perform to complete a Sametime server before you begin to integrate it with your site:

- Install and set up the prerequisite underlying Domino server and its Domino Administrator client software.
- Install the Sametime server software.
- Set up the Sametime server software using a browser on the server home page.

Usage restrictions: You are authorized to install and use Domino Directory solely and exclusively with your use of Sametime. Consult the product license for details.

Where to find installation procedures

The IBM Lotus Documentation website lists all documentation for all releases of Sametime.

Table 131. Installation resources for recent Sametime releases

Platform	V8	V8.5	V9
Windows Linux	In the *IBM Sametime 8 documentation*, see Installing a Sametime server on Windows, AIX, Linux, or Solaris.	In the *IBM Sametime 8.5 documentation*, see Installing on AIX, Linux, Solaris, and Windows.	In the *IBM Sametime 9 documentation*, see the Deploying section. **Note:** Refer to the topics available in *Deploying instant messaging*. Details for each specific platform are available for each server installation option.
IBM i	In the *IBM Sametime 8 documentation*, see Installing Sametime server on i5/OS.	In the *IBM Sametime 8.5 documentation*, see Installing on IBM i.	

To install a Sametime proxy server, see *Installing a Sametime Proxy Server* in the Sametime documentation.

Critical choices for Sametime V8 in a WebSphere Portal Express site

The following table lists choices common to Sametime installation on most or all platforms, with recommendations for use of Sametime with WebSphere Portal Express. Because specific installation steps differ between platforms, the descriptions of the choices are general. Do not expect to see the exact wording of the choice when installing for your platform and release. The options that are highlighted in this table are a subset of all choices you make during installation; in general, if not specified, accept default options.

In addition, before installation, see the topic on planning for collaborative servers and portlets, which describes several common types of collaborative sites, and includes considerations that are related to platform, user directories, security, and performance.

Tip: Be sure to keep a record of all names and passwords you specify, and all directories where the installation programs installation software.

Table 132. Key decisions for Sametime V8 installation

Procedure	Decision point	Recommendations
Sametime server software installation	**Setting up the Directory Type** dialog box You can choose Domino Directory (this selection is known as *native*) or an LDAP server (which can be either a separate Domino LDAP server or other non-Domino LDAP server) as the repository for user authentication and management.	The people awareness features and the authentication features behave differently, depending on the directory you choose. • If you select Domino Directory, people awareness can display names in both the common name format and hierarchical name format. Configuring Sametime to use the native Domino Directory allows the most flexibility for people awareness. By default, Sametime searches the native Domino Directory first for user names. • If you choose an LDAP server, Sametime installs a Directory Assistance database (`da.nsf`). If directory assistance is used, multiple LDAP directories can be searched after Sametime queries the native Domino Directory. In addition, because Sametime searches the native Domino Directory first and external LDAP servers second, the Domino Directory on the Sametime server should contain only users that are for administering Sametime, not WebSphere Portal Express users. The Directory Assistance feature is useful in environments where the same users are not listed in both the LDAP directory and Domino Directory.

Sametime 8.5 requires the use of an LDAP directory for user authentication.

Related information:

IBM Support

Sametime Product Page

WebSphere Portal detailed system requirements

Installing a Sametime Proxy Server

Configuring Sametime Proxy

If IBM WebSphere Portal Express and IBM Sametime are both authenticating with the same LDAP server, SSO configuration is simple.

Before you begin

1. Install WebSphere Portal Express and configure it with an LDAP.
2. Install and configure the IBM Domino 8.5.2 server.
3. Install and configure the Lotus Notes Administrator client.
4. Install the IBM Sametime server with the IBM Domino 8.5.2 server and configure the same LDAP you used with WebSphere Portal Express.
5. Install the Sametime Proxy server with the Sametime server. Configure it to work with the WebSphere Portal Express server.
6. Reboot the Sametime Proxy sever and the WebSphere Portal Express server.

Note: Only users in the LDAP will have awareness functions.

About this task

Use the following steps to configure WebSphere Portal with the Sametime Proxy server.

Procedure

Configuring Sametime single sign on to work with WebSphere Portal Express.

1. Install WebSphere Portal Express Version 8.5.
2. Log in to the IBM WebSphere Application Server Integrated Solutions Console and click **Security** > **Global Security**.
3. Click **Web and SIP security** > **Single Sign-on SSO**.
4. Set the single sign-on domain and save it to Master Configuration.
5. From the main Global Security page, click **LTPA**.
6. Enter a password for the token, then enter a path for the token.
7. Click **Export Keys** and then **OK**.
8. Save to Master Configuration.
9. Copy the key to the Sametime Proxy server machine.
10. Open the **Address Book** for the domain.
11. Open the **Web** > **Web Configuration** section. Open the **Web SSO Configuration** twistie and delete the existing token.
12. Click the **Configuration** tab. Click **Server** > **All Server documents**.
13. Open **Web** > **Web SSO Configuration**. The Web SSO Configuration for: page opens.
14. Open **Keys** > **Import WebSphere LTPA Key**.
15. Enter the path and password for the key you created. The key imports.
16. Set the domain starting with a period. For example, `.rtp.yourco.com`.

    Note: There must be a period at the beginning of the domain.

17. Choose the Domino server name from the twistie. Choose the **Domino Address Book** as the source.
18. Set the token format to `LtpaToken` and `LtpaToken 2`.
19. Click **Save and Close** to save the token.

Creating the Resource Environment Providers

20. Log in to the IBM WebSphere Application Server Integrated Solutions Console.
21. Click **Resources** > **Resource Environment Providers**.
22. Open the **WP CommonComponentConfigService** provider.
23. Create the following custom properties if the are not already created:

    cc.sametime.proxy.enabled
    Set the value to `true`.

    cc.sametime.proxy.scheme
    Set the value to `http` or `https`. It must match the way your Sametime Proxy Server is accessed.

    cc.sametime.proxy.host
    Set the value to the name of your server. For example, `hostname.domainname.com`.

    cc.sametime.proxy.port
    Set the value to the port of your server.

    cc.sametime.connect.client
    Set the value to `false`. If you set the value to true Sametime Proxy uses the Sametime connect client which is installed on Sametime Proxy server machine.

    cc.sametime.proxy.version
    Set the value to `9.0`.

    cc.sametime.proxy.includedock
    Set the value to `true` to show the Sametime web client dock.
24. Save to Master Configuration.
25. Log in to WebSphere Portal Express as a Sametime user.
26. Navigate to the page with the Sametime Web 2.0 Contact List portlet.
27. Edit the page properties and set the profile to the custom profile you created.
28. Log in as a user in the LDAP and open the page with the Sametime Web 2.0 Contact List portlet. Click **Applications** > **Collaboration** > **IBM Sametime**. The portlet shows that the user is online and you can use other Sametime Web 2.0 Contact List functions to see other online users.

Related tasks:

"Serving HTTP OPTIONS requests to the server context root by WebDAV clients" on page 1358
Some WebDAV clients send an HTTP OPTIONS request to the server context root (/) to check whether the server supports WebDAV. To support these clients, the portal provides a web application called `wp.webdav.options.war` that you can enable. This application responds to such requests with a confirmation that the portal supports WebDAV.

Related information:

Exporting Lightweight Third Party Authentication keys

People awareness

People awareness makes people's names appear as hyperlinks that users can click to display information about the individual and gain access to actions for contacting and working with him or her. If the administrator has configured an IBM Sametime server to work with WebSphere Portal Express, users can see each other's online presence in their person links according to the status options they have set in their Sametime client (for example, whether the person is active, away, offline, or does not want to be disturbed). The person's online status appears only if Sametime is enabled.

Note: Users who are not in an LDAP user registry do not have awareness and cannot see if other users are online. This can happen if you install your portal and then enable a Federated LDAP or Federated DB repository that does not contain that user. Also, users who sign up using the Self Care portlet do not have awareness.

User impersonation and people awareness: When a user who is enabled for impersonation impersonates other users, the people awareness feature is disabled for the entire session for which that user is authenticated.

Person card

Move the cursor over an active (underlined) name to see the **Click here for Person Card** option. The Person card displays business card details such as the email address, job title, and so on. Available actions display following the business card section.

Click **Profile** to display full information about the person including (by default) business card information, contact information, current job, and background. The information in the business card section is distinct from the set of information you can configure to appear from a person link, so you may choose to display different information about people in each of these contexts.

You can also customize how much information is displayed. For example, you can show just the business card fields by default and let users choose whether to expand the information shown to include contact information, job, and background fields. You can also configure how long the Person card displays.

Additional actions that are available on the Person card depend on whether Domino and collaboration products, the Lotus Collaborative Services, or both, are configured to work with WebSphere Portal Express. These actions can include:

- **Send Mail**

  Opens a new message in the preferred email client as specified in browser and operating system. This option appears only if the user has an email address.
- **Chat**

  Appears only if Sametime is enabled. This action is not available if the person is offline or has set status to "Do not disturb me."
- **Add as Sametime Contact**

  Appears only if Sametime is enabled. Action displays a window where a user can add the person to the contact list, as a member of a new or existing personal group. For more information about Sametime client features, see the Sametime documentation.

If you are using a screen reader, you can press Shift + Enter to see the Person card, and then press Tab to navigate through the available actions.

People awareness and the Person tag

People awareness in the form of online presence (names displayed as hyperlinks) is supported by Collaborative Services. In contexts where the Person card is available, the Person JSP tag provides contextual collaboration functionality that is related to a named person. The tag generates the HTML that renders both the actions to display on the Person card and the online presence state to display for that person, taking into account the Domino and collaboration product servers that are installed and enabled in the portal environment.

For more information on configuring a collaborative portal, see the topic on integrating Domino and the Extended Product Portlets into WebSphere Portal Express. When configuring collaboration, pay special attention to the Collaborative Services environment properties file, `CSEnvironment.properties`, which supports people awareness.

For details on the implementation of the Person tag, and instructions on customizing the tag for WebSphere Portal Express applications that you develop, see the Collaborative Services API topic.

"Configuring contact information on person links"
When users click a person link, the Person card displays contact information for the selected person. You configure this information by modifying custom properties in the WebSphere Integrated Solutions Console, specifying the Member Manager attributes that correspond to the fields you want to display in the contact information and the order in which you want the information to appear.

Related tasks:

"Setting display duration for the Person card" on page 3107
You can configure the Person card to display longer than the default number of milliseconds by modifying the `personTagTimeout` custom property in the WebSphere Integrated Solutions Console.

Related reference:

"People Finder" on page 928
The **People Finder** portlet provides both quick search and advanced search options for locating people and information about people. Once found, a person is visible to other users as a person link that indicates online presence and displays a menu of instant messaging and other options.

Related information:

Sametime documentation

Configuring contact information on person links:

When users click a person link, the Person card displays contact information for the selected person. You configure this information by modifying custom properties in the WebSphere Integrated Solutions Console, specifying the Member Manager attributes that correspond to the fields you want to display in the contact information and the order in which you want the information to appear.

Procedure

1. Select the appropriate console, depending on your environment:

- If you are running stand-alone, use the local WebSphere Integrated Solutions Console.
- If you are running in a cluster, use the console of the Deployment Manager.

2. Start the WebSphere Integrated Solutions Console by entering the URL in the location field of a web browser:

   `http://example.com:admin_port/ibm/console`

   where *example.com* is the name of your server and *admin_port* is the port that is assigned to the WebSphere Integrated Solutions Console.
3. In the navigation, click **Resources** > **Resource Environment** > **Resource Environment Providers**.
4. Locate and click the resource **WP PeopleService**.
5. Under Additional Properties, click **Custom Properties**.
6. Locate the custom properties **collapsedBCardItems** and **expandedBCardItems** and set each property's value to the Member Manager attributes that you want to display. The order in which you specify the attributes determines the order in which the corresponding information displays.

   collapsedBCardItems
   Identifies the items that the Person card displays when business card details are hidden from view. For example: `ibm-jobTitle`, `telephoneNumber`

   expandedBCardItems
   Identifies the items that the Person card displays when business card details are visible. For example: `ibm-primaryEmail`, `homePostalAddress`, `stateOrProvinceName`, `postalCode`, `countryName`

   The following Member Manager attributes are valid for specifying person link contact information:
   - businessCategory
   - carLicense
   - cn
   - countryName
   - departmentNumber
   - description
   - displayName
   - employeeNumber
   - employeeType
   - facsimileTelephoneNumber
   - givenName
   - homePostalAddress
   - ibm-gender
   - ibm-hobby
   - ibm-jobTitle
   - ibm-otherEmail
   - ibm-personalTitle
   - ibm-primaryEmail
   - ibm-regionalLocale
   - ibm-timeZone
   - Initials

- localityName
- manager
- mobile
- pager
- postalAddress
- postalCode
- preferredLanguage
- roomNumber
- secretary
- seeAlso
- sn
- stateOrProvinceName
- street
- telephoneNumber
- uid

7. Click **Apply** and then save the settings.
8. Restart the portal server.

Collaborative Services environment properties

If your collaborative site requires optional configuration that is not accomplished by the tasks that you run on the IBM WebSphere Portal Express server to integrate IBM Domino and the collaboration products, you can modify the operation of the collaborative servers and portlets in various ways by manually editing the Lotus Collaborative Services environment properties file (`CSEnvironment.properties`) on the WebSphere Portal Express server.

The file is installed in the following directory:

Windows
wp_profile_root`\PortalServer\config\config`

Linux *wp_profile_root*`/PortalServer/config/config`

IBM i *wp_profile_root*`/PortalServer/config/config`

This file contains the following information about the portal environment:

- A flag to indicate whether the Collaborative Services are being used within the portal context
- Location, protocol, port, and version of Domino Directory server
- Location, protocol, port, and version of IBM Sametime server for Sametime 8.5.1 or earlier
- Configuration and performance tuning settings specific to Collaborative Services
- A flag to indicate the type of token that is being used: **`lpta token`** or **`ltpa token2`**

"Editing the CSEnvironment.properties file" on page 963
To modify any Lotus Collaborative Services environment properties, you must stop the portal server, locate the file in a location specific to your platform, back up the file before editing it, use a text editor to open and modify it, and then restart the server.

"Auto-detecting user mail information from a secondary LDAP server" on page 964

You can set the Lotus Collaborative Services in the portal to detect users' mail file information from an additional (secondary) non-Domino LDAP user directory. For example, you may need to configure two directories if your organization has one for customers and one for employees.

"Customizing Collaborative Services user credentials for eTrust SiteMinder" on page 964
If you protect the portal and any of the Domino and Extended Products Portlets or Common Mail portlet with Computer Associates eTrust SiteMinder, you must set the Lotus Collaborative Services to use the eTrust SiteMinder token instead of the default LTPA token.

"Supporting automatic mail detection with an LDAP directory other than Domino" on page 965
If the LDAP directory configured for the portal and for Lotus Collaborative Services is not IBM Domino, and you want the automatic mail detection feature in Domino messaging portlets, you can modify the `CSenvironment.properties` file to support the feature.

"Tuning performance of the Domino Directory" on page 967
If you are using Domino Directory as the primary (and only) LDAP server for WebSphere Portal Express, you can set the following property in the `CSEnvironment.properties` file to false to improve the performance of Domino Directory.

"Using LtpaToken2 for user login" on page 967
By default, the credential settings in the `CSEnvironment.properties` file are set to use an LTPA token for user login. If your environment is configured with LtpaToken2 only, you must modify the `CSEnvironment.properties` file to use LtpaToken2 instead of LtpaToken.

Editing the CSEnvironment.properties file

To modify any Lotus Collaborative Services environment properties, you must stop the portal server, locate the file in a location specific to your platform, back up the file before editing it, use a text editor to open and modify it, and then restart the server.

Before you begin

About this task

Procedure

1. Stop the WebSphere_Portal server.
2. Open the `CSEnvironment.properties` file in a text editor. The file is located in the following directory:

   Windows: *wp_profile_root*`\PortalServer\config\config`

   Linux: *wp_profile_root*`/PortalServer/config/config`

   IBM i: *wp_profile_root*`/PortalServer/config/config`
3. Before making any changes, make a back up copy of the `CSEnvironment.properties` file.
4. Edit the `CSEnvironment.properties` file to include the appropriate values.
5. Remove the comment tag (#) from the beginning of each edited line.
6. Save the changes.
7. Start the WebSphere_Portal server.

Auto-detecting user mail information from a secondary LDAP server

You can set the Lotus Collaborative Services in the portal to detect users' mail file information from an additional (secondary) non-Domino LDAP user directory. For example, you may need to configure two directories if your organization has one for customers and one for employees.

Before you begin

The secondary LDAP directory server is specified for the property `CS_SERVER_DOMINO_DIRECTORY_1.custom_ldap_host`.

About this task

Modify the `CSEnvironment.properties` file.

The following example shows the syntax with comments.

```
# Optional advanced settings
# The following fields are disabled, by default.
# If it is enabled (determined by custom_ldap_host)
and a different server is specified,
# The following user information will be retrieved
from this secondary server.
# Mail Server,  Mail file and Email address
#
#CS_SERVER_DOMINO_DIRECTORY_1.custom_ldap_host=my.server.com
#CS_SERVER_DOMINO_DIRECTORY_1.custom_ldap_port=389
#CS_SERVER_DOMINO_DIRECTORY_1.custom_ldap_ssl=true
#CS_SERVER_DOMINO_DIRECTORY_1.custom_ldap_searchBase=base
```

Customizing Collaborative Services user credentials for eTrust SiteMinder

If you protect the portal and any of the Domino and Extended Products Portlets or Common Mail portlet with Computer Associates eTrust SiteMinder, you must set the Lotus Collaborative Services to use the eTrust SiteMinder token instead of the default LTPA token.

Before you begin

The following are custom credential settings with the possible values shown as variables:

```
CS_SERVER_CUSTOM_CRED.enabled=true/false
CS_SERVER_CUSTOM_CRED.useridAttribSource=header/cookie
CS_SERVER_CUSTOM_CRED.useridAttrib=useridAttribName
CS_SERVER_CUSTOM_CRED.ssoTokenAttribSource=header/cookie
CS_SERVER_CUSTOM_CRED.ssoTokenAttrib=tokenAttribName
```

About this task

The custom settings that you use for this task accomplish two goals:

- They override the logged in user's credentials through a custom user name, allowing mapping of principal user identities (fully qualified user names or DNs) between two LDAP directories. In this case, the *useridAttrib* setting is retrieved from the header.
- They override the logged in user's credentials with a custom SSO token that is generated from eTrust SiteMinder. In this case, the *tokenAttribName* setting is retrieved from the cookie.

Procedure

1. Make sure that WebSphere Portal Express, Domino, and Sametime are all configured properly so that eTrust SiteMinder can authenticate.
2. Modify the `CSEnvironment.properties` file.
3. In the **Collaborative services Credential Overrides** section, modify settings to match the following example, where *SMSESSION* is the name of the token that is generated by eTrust SiteMinder, and *SM_USERDN* is the same as the attribute passed by eTrust SiteMinder to Domino and Sametime.

   Tip: The attribute is usually *SM_USERDN*. Other common variations are *SM_NOTESDN*, *SM_USER*, or *SM_USERUID*. If the Domino servers in your site are already protected by eTrust SiteMinder, examine the eTrust SiteMinder WebAgent Configuration file (`WebAgent.conf`) on the Domino server and use the attribute that is specified in the field *dominoheaderforlogin*.

   ```
   CS_SERVER_CUSTOM_CRED.enabled=true
   # Valid values are header/cookie
   CS_SERVER_CUSTOM_CRED.useridAttribSource=header
   CS_SERVER_CUSTOM_CRED.useridAttrib=SM_USERDN
   # Valid values are header/cookie
   CS_SERVER_CUSTOM_CRED.ssoTokenAttribSource=cookie
   CS_SERVER_CUSTOM_CRED.ssoTokenAttrib=SMSESSION
   ```
4. Create new parameters for each instance of the Common Mail and Lotus Notes View portlets in your site. For more information, see the section on the **`AuthTokenName`** parameter for Lotus Notes View, and the section on the **`CPP_PassHttpCookies`** parameter for the Common Mail portlet.

Related tasks:

"Configuring eTrust SiteMinder to perform authentication" on page 1661
IBM WebSphere Portal Express includes a configuration task called `enable-sm-tai`. This task interacts with IBM WebSphere Application Server security configuration to enable the eTrust SiteMinder TAI and to create it as one of the interceptors. You can configure eTrust SiteMinder to provide authentication independently from configuring it to provide authorization. Using it to perform authorization only is not supported at this time.

Supporting automatic mail detection with an LDAP directory other than Domino

If the LDAP directory configured for the portal and for Lotus Collaborative Services is not IBM Domino, and you want the automatic mail detection feature in Domino messaging portlets, you can modify the `CSenvironment.properties` file to support the feature.

Procedure

1. Modify the `CSEnvironment.properties` file.
2. Enable the lines under the Mail server and Mail File Queries section. The following example shows the syntax with comments.

   ```
   # Mail server and Mail File Queries:
   #CS_SERVER_DOMINO_DIRECTORY_1.
   mailfileserver_objectclass=person
   #CS_SERVER_DOMINO_DIRECTORY_1.
   mailserver_attrib=mailserver
   #CS_SERVER_DOMINO_DIRECTORY_1.
   mailfile_attrib=mailfile
   # Email Address query
   CS_SERVER_DOMINO_DIRECTORY_1.
   email_objectclass=person
   CS_SERVER_DOMINO_DIRECTORY_1.
   email_attrib=internetaddress
   ```

Note: If adding the mailserver attribute to a non-Domino LDAP server and Lotus Collaborative Services is configured with this server, make sure that this attribute is in the fully qualified distinguished name `my.server.com` format.

3. To enable the `CS_SERVER_DOMINO_DIRECTORY_1.mailserver_attrib` and `CS_SERVER_DOMINO_DIRECTORY_1.mailfile_attrib` attributes for the primary non-Domino LDAP directory that is configured for your portal site, add the following entry:

   ```
   CS_SERVER_DOMINO_DIRECTORY_1.primary_ldap_custom_attribute_enabled=true
   ```

   Note: By default, these attributes are used only if your portal configuration uses a custom LDAP directory, but adding the previous command line allows them to be used for a non-Domino LDAP.

4. Enable the following section in the file, specify your Domino LDAP for `CS_SERVER_DOMINO_DIRECTORY_1.hostname`, and make sure that there is no return character after the equal sign on that line:

   ```
   ############################################################
   #
   # DOMINO DIRECTORY properties
   # (LDAP server)
   # Important:
   # Should always point to a Domino Server.
   # Leave enabled flag as true.
   # Use the custom_ldap_* settings to point
    to any LDAP Server to
   # get user information.
   ############################################################

   CS_SERVER_DOMINO_DIRECTORY.enabled=true
   CS_SERVER_DOMINO_DIRECTORY_1.hostname=
   yourserver.yourdomain.com
   CS_SERVER_DOMINO_DIRECTORY_1.port=389
   CS_SERVER_DOMINO_DIRECTORY_1.ssl=false
   CS_SERVER_DOMINO_DIRECTORY_1.anonymous=true
   ```

5. Copy and paste the following section into the file directly following the section in the previous step, and enable it. If there are return characters that follow the equal signs or anywhere else in the middle of each statement, remove them.

   ```
   ########################################################
   # dual directory settings
   ########################################################

   CS_SERVER_DOMINO_DIRECTORY_1.searchBase=O=DominoPortal

   # In the following queries
   #   %c = common name
   #   %d = ldap dn
   #  %n = fq notes name
   #  %v = the first item in the id like "cn=jane doe" or "uid=jdoe"

   #CS_SERVER_DOMINO_DIRECTORY_1.query_base_search=(objectclass=*)

   #CS_SERVER_DOMINO_DIRECTORY_1.query_distinguished_name=
   (& (objectclass=person) (|(uid=%c)(cn=%c)))
   #CS_SERVER_DOMINO_DIRECTORY_1.query_distinguished_name_attrib=cn

   #CS_SERVER_DOMINO_DIRECTORY_1.query_domino_servers=
   (&(objectclass=dominoServer)(http-hostname=*))
   #CS_SERVER_DOMINO_DIRECTORY_1.query_domino_servers_attrib=
   http-hostname

   #CS_SERVER_DOMINO_DIRECTORY_1.query_http_host_name=
   (| (& (objectclass=server)(%v) ) (& (objectclass=dominoServer)(%v) ) )
   #CS_SERVER_DOMINO_DIRECTORY_1.query_http_host_name_attrib=
   http-hostname

   #CS_SERVER_DOMINO_DIRECTORY_1.query_last_resort=
   (| (& (objectclass=person)(%v) ) (& (objectclass=groupOfNames)(%v) )
   ```

```
(& (objectclass=server)(%v) )(& (objectclass=dominoServer)(%v) ))

CS_SERVER_DOMINO_DIRECTORY_1.query_user_emailaddr=
(&(objectclass=person)(cn=%c))
#CS_SERVER_DOMINO_DIRECTORY_1.query_user_emailaddr_attrib=
internetaddress

CS_SERVER_DOMINO_DIRECTORY_1.query_user_mailserverfile=
(&(objectclass=person)(cn=%c))
#CS_SERVER_DOMINO_DIRECTORY_1.query_user_mailserverfile_attrib=
mailserver,mailfile

#  %l = login name
#  %a = wmm attribute use with CS_SERVER_DOMINO_DIRECTORY_1.other_lookup_attribute

CS_SERVER_DOMINO_DIRECTORY_1.allow_dn_search=false
CS_SERVER_DOMINO_DIRECTORY_1.other_lookup_attribute=ibm-primaryEmail
CS_SERVER_DOMINO_DIRECTORY_1.query_user=(&(objectclass=person)(mail=%a))
```

6. Save the properties file.

Tuning performance of the Domino Directory

If you are using Domino Directory as the primary (and only) LDAP server for WebSphere Portal Express, you can set the following property in the `CSEnvironment.properties` file to false to improve the performance of Domino Directory.

About this task

Modify the `CSEnvironment.properties` file to change the following setting:

```
CS_PERF_PROP_USEWMM.enabled=true
```

Using LtpaToken2 for user login

By default, the credential settings in the `CSEnvironment.properties` file are set to use an LTPA token for user login. If your environment is configured with LtpaToken2 only, you must modify the `CSEnvironment.properties` file to use LtpaToken2 instead of LtpaToken.

About this task

Note: If your environment uses both tokens (LtpaToken and LtpaToken2), you do not need to change this setting in `CSEnvironment.properties`

Procedure

1. Modify the `CSEnvironment.properties` file.
2. To override the default LtpaToken setting and use LtpaToken2 instead, change the following setting to the value shown:

   ```
   CS_SERVER_WEBSPHERE_PORTAL_EXTEND.credential_type=LtpaToken2
   ```

Integrating business processes

Access relevant tasks and activities for multiple business process management solutions from a single user interface. The Unified Task List portlet provides a single point of integration between multiple systems and displays tasks that WebSphere Portal Express users must complete to advance workflows. The topics in this section provide information and instructions for developing and customizing the Unified Task List.

Unified Task List Portlet features a service provider layer that accesses, retrieves, and formats workflow events from a back-end system. The service provider layer

is coupled with a presentation layer that renders the data in a visually appealing user interface. The separation of the service layer and the presentation layer lets you develop, test, and deploy your own customized Unified Task List in a way that conforms with service-oriented architecture.

The Unified Task List portlet can be found on the Collaborations page of WebSphere Portal Express.

Note: If Unified Task List is deployed in clustered environment, there is no need to repeat any configuration steps on the secondary nodes.

"Overview of the Unified Task List portlet"
Review concepts about the Unified Task List portlet to understand the different elements.

"Configuring the Unified Task List portlet" on page 969
You can configure the Unified Task List portlet to retrieve tasks from IBM Forms Experience Builder, WebSphere Lombardi Edition, andIBM Business Process Manager.

"Configuring Unified Task List portlet with process servers" on page 976
You can configure specific process servers to run tasks in the Unified Task List portlet.

"Configuring the Unified Task List portlet with IBM Forms Experience Builder" on page 977
This task provider retrieves tasks from IBM Forms Experience Builder and surfaces them in the Unified Task List portlet.

Overview of the Unified Task List portlet

Review concepts about the Unified Task List portlet to understand the different elements.

"Task providers" on page 969
Task providers are services that access back-end systems to retrieve tasks. The task providers also use a Web Experience Factory Transform builder to provide a uniform data set that displays in the Unified Task List user interface.

"Task provider instance" on page 969
Task provider instances are services that access back-end systems to retrieve tasks. Task provider instances reside in the Task Provider Instance Registry (TPIR) and contain the parameters that you specify in task providers.

"Task Provider Instance Registry" on page 969
The Task Provider Instance Registry (TPIR) contains task provider instance configurations. A task provider instance configuration contains a set of parameters that are required to connect to a back-end system. It also contains a unique ID to map the parameters to the appropriate task provider. The Task Provider Instance Registry service is in IBM WebSphere Application Server and stores the task provider configurations in an XML variable. The Task Provider Instance Registry service also provides a service to get and modify task provider instances.

"Task dispatcher" on page 969
The task dispatcher acts as a link between the Unified Task List portlet and the task providers. When an action occurs in the portlet, the task dispatcher retrieves task provider instance configurations from the task provider instance registry service and calls a `getTaskList` service operation on each task provider instance configuration.

"Task handler" on page 969
Task handlers define what the Unified Task List portlet does when users select

a task to advance a workflow. The task handlers determine how the Unified Task List portlet connects to the tasks that the users must complete.

Task providers

Task providers are services that access back-end systems to retrieve tasks. The task providers also use a Web Experience Factory Transform builder to provide a uniform data set that displays in the Unified Task List user interface.

You can create multiple task providers to aggregate tasks from several back-end systems. For example, you can create a task provider to access, retrieve, and format tasks from a particular back-end system. You can then create another task provider to access, retrieve, and format tasks from a different back-end system. The result of these two task providers is that a single set of tasks displays in the user interface, but originate from two different back-end systems.

Task provider instance

Task provider instances are services that access back-end systems to retrieve tasks. Task provider instances reside in the Task Provider Instance Registry (TPIR) and contain the parameters that you specify in task providers.

Task Provider Instance Registry

The Task Provider Instance Registry (TPIR) contains task provider instance configurations. A task provider instance configuration contains a set of parameters that are required to connect to a back-end system. It also contains a unique ID to map the parameters to the appropriate task provider. The Task Provider Instance Registry service is in IBM WebSphere Application Server and stores the task provider configurations in an XML variable. The Task Provider Instance Registry service also provides a service to get and modify task provider instances.

Task dispatcher

The task dispatcher acts as a link between the Unified Task List portlet and the task providers. When an action occurs in the portlet, the task dispatcher retrieves task provider instance configurations from the task provider instance registry service and calls a `getTaskList` service operation on each task provider instance configuration.

You can configure the Unified Task List portlet to use a cached task dispatcher if you do not want to access the back-end systems each time an action occurs in the user interface. The cached task dispatcher uses dynamic caching in WebSphere Application Server to store and retrieve tasks.

Task handler

Task handlers define what the Unified Task List portlet does when users select a task to advance a workflow. The task handlers determine how the Unified Task List portlet connects to the tasks that the users must complete.

Front-end developers can customize the Unified Task List portlet in Web Experience Factory if you want to provide functions for completing tasks. For example, if there is a task to approve or deny a request in a Human Resources workflow, you can use Web Experience Factory builders to provide a simple menu with approve and deny actions.

Configuring the Unified Task List portlet

You can configure the Unified Task List portlet to retrieve tasks from IBM Forms Experience Builder, WebSphere Lombardi Edition, andIBM Business Process Manager.

"Configuring Unified Task List portlet for single sign-on"
Set up a single sign-on between IBM Process Server, IBM Forms Experience Builder, WebSphere Lombardi Edition, IBM Business Process Manager and WebSphere Portal Express. Single sign-on (SSO) provides a secure method of authenticating a user within an environment and applies that authentication to access other applications, systems, and networks for the duration of the session.

"Configuring Unified Task List portlet at run time"
You can customize the Unified Task List portlet at run time as a WebSphere Portal Express administrator. Customizing at run time involves accessing the deployed Unified Task List to configure the portlet settings.

Configuring Unified Task List portlet for single sign-on

Set up a single sign-on between IBM Process Server, IBM Forms Experience Builder, WebSphere Lombardi Edition, IBM Business Process Manager and WebSphere Portal Express. Single sign-on (SSO) provides a secure method of authenticating a user within an environment and applies that authentication to access other applications, systems, and networks for the duration of the session.

About this task

In the context of the Unified Task List portlet single sign-on enables communication between WebSphere Portal Express, Process Server, IBM Forms Experience Builder, WebSphere Lombardi Edition Server and IBM Business Process Manager . Because the single sign-on and LTPA configuration settings are WebSphere Application Server security settings, it can be applied to any server that runs on WebSphere Application Server .

Visit the WebSphere Application Server documentation for details on how to configure single sign-on and LTPA between two WebSphere Application Server.

Configuring Unified Task List portlet at run time

You can customize the Unified Task List portlet at run time as a WebSphere Portal Express administrator. Customizing at run time involves accessing the deployed Unified Task List to configure the portlet settings.

"Accessing the configuration view" on page 971
Customize the Unified Task List portlet by accessing the configuration view. The configuration view contains several windows that let you perform tasks such as editing settings, configuring task provider instances and task handlers, or adding and removing filters.

"Editing common settings" on page 971
Edit common settings from the Unified Task List portlet.

"Adding and removing filters" on page 972
Filters provide specialized views for tasks. Filters let you separate tasks according to the back-end systems where the tasks originate or by a particular context. When you add filters, the Unified Task List portlet displays tasks separated according to the task provider instance. For example, you can add a filter for tasks that come from IBM Process Server portlet so that these tasks are not displayed with all other tasks in the Unified Task List portlet.

"Localizing the task list filters" on page 973
When creating new filters, it is recommended that you know before hand which filters you need to create and then create filters all at once. Creating all filters at once minimizes the times you must restart the server. The restart is required to update the filter resource bundles on the file system with the new

filters and include them on the class loader. A WebSphere Application Server variable must also be created to define the name of the resource bundle to use for the localized filters.

"Exposing the Business Processes to enable Task Handling Configuration" on page 974
To enable task handling configuration data to be retrieved for the Unified Task List BPM task provider you must first expose your Business Process and Human Service data to all users. This section is only applicable if you are working with the IBM Business Process Manager task provider in the Unified Task List.

"Showing and hiding table columns" on page 974
The Unified Task List portlet shows tasks in a table. The task information presented in this table includes priority, description, owner, among others. You can show or hide these table columns.

"Editing shared settings" on page 974
The Unified Task List portlet has shared settings that are common to all users, such as the filters that display in the user interface. You can edit these settings by accessing the portlet menu.

"Changing the number of rows in the task list table" on page 975
The task list table contains rows of tasks that users can select. You can edit the task list table to display a specific number of rows per page in the Unified Task List portlet. You can specify any number of rows per page, however, if there are fewer tasks than the number of rows available, the Unified Task List portlet displays only the rows that contain tasks. If there are more tasks than the number of rows available, the Unified Task List portlet displays a link to another page for those tasks.

"Configuring dynamic user interface" on page 975
Create a dynamic page and register it using the ConfigEngine.

Accessing the configuration view:

Customize the Unified Task List portlet by accessing the configuration view. The configuration view contains several windows that let you perform tasks such as editing settings, configuring task provider instances and task handlers, or adding and removing filters.

About this task

Accessing the configuration view involves clicking the portlet menu icon from the Unified Task List portlet.

Procedure

1. Log in to WebSphere Portal Express with either as an administrator or a user assigned to the Manager role for the Unified Task List portlet.
2. Navigate to the Unified Task List portlet.
3. Hover the cursor over the Unified Task List portlet title bar to access the portlet menu and select Configure.

Editing common settings:

Edit common settings from the Unified Task List portlet.

Procedure

1. Log in to WebSphere Portal Express with administrative access to the Unified Task List portlet.
2. Access the Unified Task List configuration view and select Common Settings, which is in the navigation pane. The Common Settings window opens.
3. Provide values for the following fields to configure the common settings:

   Show Menu
   Select this check box to display the filter menu in the Unified Task List. The filter menu displays a list of filters that segregate tasks from multiple back-end systems.

   Enable Claim-Release of Tasks
   Select this check box to let users perform claim and release operations on tasks. Users perform claim and release operations in the Unified Task List through the use of the Claim and Release buttons in the user interface. When users select a task and click **Claim**, they take ownership of the task and remove the task from the list of shared tasks. When users select a task and click **Release**, they no longer claim ownership of the task. The task then returns to the list of shared tasks. Users can also perform claim and release operations through the drag-and-drop menu that is displayed in the user interface.

   Enable Business Card Integration
   Select this check box to show user details when hovering over the owner or originator in a task list. The Unified Task List uses the people-awareness features in WebSphere Portal Express to retrieve the user details from the corresponding profile pages. To show user awareness of the owner or originator in a task list, you must configure a IBM Sametime server to interact with the WebSphere Portal Express Server.

   Use Dynamic Cache
   Select this check box to store and retrieve tasks in the WebSphere Application Server dynamic cache service. When using the WebSphere Application Server dynamic cache service, the task dispatcher stores and retrieves tasks in the cache for faster task loading.

   Dynamic Page Extension Node UID
   This field is required if you have task handlers that use dynamic pages for task completion. Specify the ID for the dynamic page extension node in WebSphere Portal Express. Leave the default value in this field if you do not have task handlers that use dynamic pages.
4. Click **Save** and **Back** to return to the main portlet page.

Adding and removing filters:

Filters provide specialized views for tasks. Filters let you separate tasks according to the back-end systems where the tasks originate or by a particular context. When you add filters, the Unified Task List portlet displays tasks separated according to the task provider instance. For example, you can add a filter for tasks that come from IBM Process Server portlet so that these tasks are not displayed with all other tasks in the Unified Task List portlet.

Adding filters:

Procedure

1. Log in to WebSphere Portal Express with administrative access to the Unified Task List portlet.
2. Access the Unified Task List portlet configuration view and select **Filters** from the navigation menu. The **Filters** window opens.
3. Select **Add**.
4. Enter a name for the filter in the **Name** field.
5. In the **Resource Key** field enter a resource key.
6. From the list select the task provider instances to include in the filter.

   Note: You can apply the filter to multiple task provider instances. Select the add icon to add additional task provider instances.
7. Select **Submit**.
8. Select **Save**.

Removing filters:
Procedure

1. Log in to WebSphere Portal Express with administrative access to the Unified Task List portlet.
2. Access the Unified Task List portlet configuration view and select **Filters** from the navigation menu. The **Filters** window opens.
3. Select the filter that you want to remove then select **Remove**.
4. Select **Save**.

Localizing the task list filters:

When creating new filters, it is recommended that you know before hand which filters you need to create and then create filters all at once. Creating all filters at once minimizes the times you must restart the server. The restart is required to update the filter resource bundles on the file system with the new filters and include them on the class loader. A WebSphere Application Server variable must also be created to define the name of the resource bundle to use for the localized filters.

Procedure

1. Access the WebSphere Integrated Solutions Console.
2. Navigate through **Resources** > **Resource Environment** > **Resource environment providers** > **WP ConfigService** > **Custom properties**.
3. Click **New**.
4. Enter `processintegration.filtersetresourcebundle` in the **Name** field.
5. Enter the name of your resource bundle in the **Value** field including the complete package name. For example: If your file is stored under `shared/app/`*`your_company`*`/filterlocales_en.properties` you would enter *`your_company`*`.filterlocales`.
6. Click **Apply**.
7. Click **OK** and then **Save to the master configuration**.
8. Place your resource bundles in the `shared/app/`*`your_company`* directory.
9. Restart the portal server.
10. Open the filter configuration page of the Unified Task List portlet and add the new filters as they were entered in the English version of the resource bundle.

The **Name** field must match the value of the resource key and the **Resource key** field must match the key used for the property in the resource bundle.

Exposing the Business Processes to enable Task Handling Configuration:

To enable task handling configuration data to be retrieved for the Unified Task List BPM task provider you must first expose your Business Process and Human Service data to all users. This section is only applicable if you are working with the IBM Business Process Manager task provider in the Unified Task List.

Procedure

1. Access the Business Process Designer for BPM and open a Business Process.
2. Select the **Overview** tab.
3. Under the section title **Exposing** click the **Select** button for the **Expose to start** field.
4. From the list of **Participant Groups** select the entry which is most appropriate for this Business Process. Either select **All Users** or the group where the Unified Task List administrative user is a member.
5. Repeat steps 3 and 4 for the **Expose business data** and **Expose performance metrics** fields.
6. Click **Save**.
7. Open the first **Human Service in the Business Process**.
8. Open the **Overview** tab.
9. Under the **Exposing** section click the **Select** button for the **Expose to start** field.
10. Select the appropriate group from the list of **Participant Groups**.
11. Click **Save**.
12. Repeat steps 8-11 for each Human Service in the Business Process.
13. Repeat steps 2-12 for each Business Process in the Designer.

Showing and hiding table columns:

The Unified Task List portlet shows tasks in a table. The task information presented in this table includes priority, description, owner, among others. You can show or hide these table columns.

Procedure

1. Log in to WebSphere Portal Express with administrative access to the Unified Task List portlet.
2. Access the Unified Task List portlet configuration view and select **Table Customizer**from the navigation pane. The **Table Customizer** window opens and a list of all the available columns.
3. Select either **Show** or **Hide** for the appropriate columns.
4. Select **Save**.

Editing shared settings:

The Unified Task List portlet has shared settings that are common to all users, such as the filters that display in the user interface. You can edit these settings by accessing the portlet menu.

Procedure

1. Log in to WebSphere Portal Express with administrative access to the Unified Task List portlet.
2. From the main page of the Unified Task List portlet, hover the cursor over the title bar to access the portlet menu and select **Edit Shared Settings**. The **Configure List of Visible Filters** window opens.
3. Select a filter from the list to add to the shared settings and click **Save**. It is possible to add multiple filters to the shared settings by selecting the **add icon**, which is parallel to the list.
4. Select the **remove icon**, which is parallel to the filter name in the **Available Filters** list to remove filters from the shared settings and click **Save**.

Changing the number of rows in the task list table:

The task list table contains rows of tasks that users can select. You can edit the task list table to display a specific number of rows per page in the Unified Task List portlet. You can specify any number of rows per page, however, if there are fewer tasks than the number of rows available, the Unified Task List portlet displays only the rows that contain tasks. If there are more tasks than the number of rows available, the Unified Task List portlet displays a link to another page for those tasks.

Procedure

1. Log in to WebSphere Portal Express with administrative access to the Unified Task List portlet.
2. From the main page of the Unified Task List portlet, hover the cursor over the title bar to access the portlet menu and select **Personalize**. The **User Interface Settings** window opens.
3. In the **Paging Size** field, enter the number of rows that you want to display per page in the task list table.
4. Select **Save**.

Configuring dynamic user interface:

Create a dynamic page and register it using the ConfigEngine.

Procedure

1. Create a page, give it a unique ID, such as `wps.utl`, and place the Unified Task List portlet on the new page.
2. Create the dynamic page template and give it a unique ID. Make sure that the page is set to inherit parent theme in the page options.
3. Open a command prompt.
4. Change to the `IBM\WebSphere\wp_profile\ConfigEngine` directory.
5. Enable the Unified Task List portlet page to start dynamic pages by running the following command:
   - Linux: `./ConfigEngine.sh action-enable-page-as-extension-node-wp.dynamicui.config -DPageUniqueName=wps.utl`
   - Windows: `ConfigEngine.bat action-enable-page-as-extension-node-wp.dynamicui.config -DPageUniqueName=wps.utl`
   - IBM i: `ConfigEngine.sh action-enable-page-as-extension-node-wp.dynamicui.config -DPageUniqueName=wps.utl`

Where **-DPageUniqueName** is the value of the unique ID you previously specified.

Configuring Unified Task List portlet with process servers

You can configure specific process servers to run tasks in the Unified Task List portlet.

The following process servers can be configured with the Unified Task List portlet:

- WebSphere Lombardi Edition.
- IBM Business Process Manager.

"Configuring a WebSphere Lombardi Edition process server"
This specific provider surfaces tasks for WebSphere Lombardi Edition in the Unified Task List portlet.

"Configuring an IBM Business Process Manager process server"
This task provider retrieves tasks from IBM Business Process Manager and surfaces them in the Unified Task List portlet.

Configuring a WebSphere Lombardi Edition process server

This specific provider surfaces tasks for WebSphere Lombardi Edition in the Unified Task List portlet.

Procedure

1. Open the applications menu and go to **Collaboration** > **Unified Task List**.
2. Open the page with the Unified Task List portlet on it.
3. Configure the Unified Task List portlet to add task provider instances for WebSphere Lombardi Edition. For more information, see *Configuring a WebSphere Lombardi Edition task provider instance* in the related links.
4. Create a task handler for the process instances of IBM Business Process Manager. For more information, see *Adding and removing task handlers* in the related links.
5. Open the applications menu and go to **Collaboration** > **Unified Task List**.
6. Add the Coach portlet to the Unified Task List page.
7. Configure the Coach portlet with host name and port number of the Lombardi server. For more information, see *Configuring the Coach portlets* in the related links.
8. Add wires to link the Coach portlet and theUnified Task List portlet. For more information on adding wires, see *Adding a wire* in the related links.
9. Claim a Lombardi task and then select it to start the process instance in the Coach portlet.

Configuring an IBM Business Process Manager process server

This task provider retrieves tasks from IBM Business Process Manager and surfaces them in the Unified Task List portlet.

Procedure

1. Open the applications menu and go to **Collaboration** > **Unified Task List**.
2. Open the page with the Unified Task List portlet on it.
3. Configure the Unified Task List portlet to add task provider instances for Business Process Manager. For more information, see *Configuring an IBM Business Process Manager task provider instance* in the related links.

4. Create a task handler for the process instances of Business Process Manager. For more information, see *Adding and removing task handlers* in the related links.
5. Open the applications menu and go to **Collaboration** > **Unified Task List**.
6. Add the Coach portlet to the Unified Task List page.
7. Configure the Coach portlet with host name and port number of the Business Process Manager server. For more information, see *Configuring the Coach portlets* in the related links.
8. Add wires to link the Coach portlet and theUnified Task List portlet. For more information on adding wires, see *Adding a wire* in the related links.
9. Claim a Business Process Manager task and then select it to start the process instance in the Coach portlet.

Configuring the Unified Task List portlet with IBM Forms Experience Builder

This task provider retrieves tasks from IBM Forms Experience Builder and surfaces them in the Unified Task List portlet.

Procedure

1. Open the page with the Unified Task List portlet on it.
2. Configure a Forms Experience Builder task provider instance. For more information, see *Configuring an IBM Forms Experience Builder task provider instance* in the related links.
3. Configure a Forms Experience Builder task handler. For more information, see *Adding and removing task handlers* in the related links.
4. Add theForms Experience Builder portlet to the Unified Task List page.
5. Configure theForms Experience Builder portlet.
6. Add a wire to link the Forms Experience Builder portlet and the Unified Task List portlet. For more information on adding wires, see *Adding a wire* in the related links.

Integrating with web applications

The web application bridge uses reverse proxy technology to integrate web-based content providers, such as the Microsoft SharePoint server, with IBM WebSphere Portal Express. Administrators must first define the virtual web applications or content providers. A lightweight iFrame portlet renders the content from the backend applications. Users can then access the iFrame on a page without requiring direct network access to the backend application. A special engine maps Uniform Resource Identifier (URIs) on the iFrame portlet to real URIs from the content providers.

The intention of the Web Applications Bridge is to support the complex and heterogeneous web applications. Ensure that the web applications are constructed with common practices and function according to industry accepted standards. Since there are no rigorous industry standards, it is impossible to verify that the Web Application Bridge can work with any arbitrary web application. You might need to customize the Web Application Bridge or change the web application to ensure success. The Web Applications Bridge was designed to integrate Microsoft Sharepoint into your website. However, you can integrate with other products. The Web Application Bridge is routinely enhanced to support integration with other products and systems per demand.

For those web applications that do not comply or are unable to work correctly, modify the application to support the integration through the Web Application Bridge. In cases where the application cannot be modified to comply with the Web Application Bridge, add custom filters to the Web Application Bridge to process the markup. Take care when you implement these filters for performance and reliability. The customer makes the customization themselves. When you use the Web Application Bridge, access WebSphere Portal Express with the fully qualified host name.

Integrating the web applications with WebSphere Portal Express is a multistep process. To learn more about the web application bridge, go to Web application bridge.

CF03 Mobile support is provided for web applications that were developed and tested for rendering inside mobile device browsers. If the web application was originally built and tested for desktops, they will not work properly on a mobile device. Instead, they display the same way that they would on the desktop. If the web application provides navigation, content, and features for a mobile device, it is available when rendered through the Web Dock portlet.

CF03 Configure the Web Dock portlet to always allow dynamic size. This setting allows the screen to respond to the different sizes of the applications. If you do not have Dynamic Size set to always, the Web Dock portlet shows the desktop version of the content.

Note: CF03 Mobile devices might not show scroll bars for any overflow content. Instead, the swipe feature of the mobile is enabled.

"Configuring the web application bridge for anonymous login"
You can map your web application bridge to allow anonymous users to log on and access information.

"Configuring multiple web dock applications on a page" on page 980
You can have multiple web dock applications that point to different hosts on the same portal page. You can also allow users to open two different portal pages that contain web dock applications in a new browser window or tab.

"Troubleshooting the web application bridge" on page 981
The troubleshooting information is useful for planning and implementing your web application bridge integration (WAB).

Related tasks:

"Providing short vanity URLs" on page 2101
You might want to make your vanity URLs as short and simple as possible for your customers. You can create vanity URLs that contain only the vanity segment by omitting the string `/wps/vanityurl`. In this case, you must use a web server and define a rewrite rule. If you also use IBM Web Application Bridge, or if you have static files in the root of the HTTP server document directory, adapt the rewrite rule.

Related information:

OpenAjax.hub.publish(name, publisherData)

Configuring the web application bridge for anonymous login

You can map your web application bridge to allow anonymous users to log on and access information.

Procedure

1. Complete the following steps to set the IBM WebSphere Portal Express security:
   a. Log in to the WebSphere Integrated Solutions Console.
   b. Go to **Applications** > **Application Types** > **WebSphere enterprise applications**.
   c. Select **wp.vwat.servlet.ear**.
   d. Select **Security role to user/group mapping**.
   e. Select **All Role** check box.
   f. Select **Everyone** from the **Map Special Subjects** menu.
   g. Click **OK**.
   h. Save your changes.
   i. Go to **Security** > **Global Security** > **Web and SIP Security**.
   j. Click **General Settings**.
   k. Select the **Use available authentication data when an unprotected URI is accessed** check box.
   l. Click **OK**.
   m. Save your changes.
   n. Log out of the WebSphere Integrated Solutions Console.
   o. Restart WebSphere Portal Express.
2. Log on to WebSphere Portal Express.
3. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
4. Click **Content Root** > **Home**.
5. Click **New page** and create a page for your web application. For this example, the new page is called TestPage.
6. Complete the following steps to give the page anonymous user access:
   a. Click the **Administration menu** icon. Then, click **Access** > **Resource Permissions**.
   b. Select **Pages** as the resource type.
   c. Locate the page that you created. For example, go to **Resource Type** > **Pages** > **Content Root** > **Home** > **TestPage**.
   d. Select **Assign access**.
   e. Click **Edit Role** corresponding to the user.
   f. Click **Add**.
   g. Select **Anonymous Portal User**.
   h. Click **Apply**.
7. Complete the following steps to give the web dock application anonymous user access:

   Note: Configure anonymous user access for each web dock application that you create.
   a. Click the **Administration menu** icon. Then, click **Access** > **Resource Permissions**.
   b. Select **Portlets** as the resource type.
   c. Locate the web dock application.
   d. Select **Assign access**.
   e. Click **Edit Role** corresponding to the user.

f. Click **Add**.
g. Select **Anonymous Portal User**.
h. Click **Apply**.

8. Add the web dock application to the page.

   Tip: To get the web dock application to render on a page, the page must either have the **Web Dock** profile or a profile that includes the wp_webdock module. Edit the page properties and change the profile or add the wp_webdock module to the profile applied to the page:

   CF03 Starting with CF03, the Web Dock profile no longer exists. If you are using the Resource Aggregator for Portlets, no additional steps are necessary. If you are not using the Resource Aggregator for Portlets, add the **wp_webdock** module to an existing profile on your page.

   a. Connect to the theme repository with the fs-type1 connection.
   b. Go to your theme.
   c. Open the profile file in the `/profiles` directory.
   d. Make a copy of the profile file and give it a unique name.
   e. Edit the `.json` file and add the **wp_webdock** module ID.
   f. Copy the profile that you created to the `/profiles` directory.
   g. Invalidate the resource aggregator cache to integrate your changes. Click the **Administration menu** icon in the toolbar. Then click **Theme Analyzer** > **Utilities** > **Control Center** > **Invalidate cache**. CF06 Auto invalidation recognizes your changes automatically for WebDAV based themes. No further action is required. For more information, see Utilities.

Configuring multiple web dock applications on a page

You can have multiple web dock applications that point to different hosts on the same portal page. You can also allow users to open two different portal pages that contain web dock applications in a new browser window or tab.

About this task

Restriction: If you complete these steps, the inter-portlet communication feature is disabled.

Important: This alias is prefixed to the server name used to access the portal. It is then used to access the portal internally. If the server name is `servername.domain.ibm.com`, then make sure that the server is accessible through `myhost1.servername.domain.ibm.com` and `myhost2.servername.domain.ibm.com`.

Procedure

1. Log in to the WebSphere Integrated Solutions Console.
2. Go to **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Click **WP Virtual Web Application Manager Config**.
4. Click **Custom properties** in the **Additional Properties** heading.
5. Click **host_alias_mapping**.
6. Enter the aliases for the hosts in **Values**. For example, enter `myhost1 = http://remotesite1.example.ibm.com, myhost2 = http://remotesite2.example.ibm.com`.

Important: The aliases for the hosts must all be lowercase.

7. Click **OK**.
8. Complete the following steps to enable single sign-on:
   a. Go to **Security** > **Global Security** > **Web and SIP security** > **Single sign-on (SSO)**.
   b. Set the **Domain name** field. For example, enter `domain.ibm.com`.
9. Restart the WebSphere_Portal server.
10. Go to your domain name server (DNS) and enter the host aliases for each of the host aliases that you plan to use. For example, enter the following information:

    portal_serverIP `myhost1.servername.domain.ibm.com`

    portal_serverIP `myhost2.servername.domain.ibm.com`

    Where *portal_serverIP* is something like *9.27.27.155*.

What to do next

Complete the following steps to remove this enhancement:

1. Log in to the WebSphere Integrated Solutions Console.
2. Go to **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Click **WP Virtual Web Application Manager Config**.
4. Click **Custom properties** in the **Additional Properties** heading.
5. Click **host_alias_mapping**.
6. Remove the aliases for the hosts in the **Values** field.
7. Click **OK**.
8. Restart the WebSphere_Portal server.

Troubleshooting the web application bridge

The troubleshooting information is useful for planning and implementing your web application bridge integration (WAB).

About this task

The following items are known limitations:

Known issue with Microsoft Silver light plug-in
Disable the Microsoft Silver light plug-in in browsers when you use the Sharepoint features with the Web Application Bridge.

Known issue when you edit the host or port information for a content provider profile
If the system administrator changes the host or port information in the content provider profile, you must edit the web dock application and reselect the profile. Otherwise, the web dock application does not pick up the changes.

Known security issue
Do not enter < or > into any of the text boxes.

Known issue with turning on information mode
If you turn on information mode while you create a content provider, the dialog closes. You return to the summary table with no additional information shown. Turn on information mode and then create your content provider.

Web applications that contain navigation to multiple hosts
A web page that is served by an application that contains URLs that point to a host that is not the registered host for the web application does not work.

Web applications with URL redirection
Any web application that uses URL redirection for serving content is not supported. For example, `<meta http-equiv="refresh" content="2;url=http://someotherserver.com/">` does not work.

Web applications that serve content with incorrect mime-types
Any web application that serves web resources with an incorrect mime-type does not work. For example, a jpg image that is served as text/html instead of the correct image/jpeg mime-type does not work.

Web applications that server an iFrame that overlays the complete page
Any web application that serves content inside an iFrame that overlays the complete page is not supported.

Web applications that prevent their content to be loaded inside an iFrame
Any web application that contains a script to check whether the page document window is the main window or not before it displays the contents does not work.

Web applications with non-standard browser features
Any web application that depends specifically on a non-standard feature that is provided by a specific vendor's web browser does not work.

Web applications with vendor-specific client-side capabilities
Any web application that uses vendor-specific offline caching APIs, for example: Google and Gears, does not work.

Web dock applications and Mobile devices

CF03 Starting with CF03, mobile support is provided for web applications that were developed and tested for rendering inside mobile device browsers. If the web application was originally built and tested for desktops, it does not work properly on mobile devices.

Note: Mobile devices might not show scroll bars for any overflow content. Instead, the swipe feature of the mobile is enabled.

If you have CF02 or earlier, web dock applications are not supported on mobile devices. You might see an error message; for example: "Error 500: Internal Server Error: Some unexpected error". The administrator can configure their web server to redirect these errors to a friendlier error message. Refer to your web server documentation for information on how to redirect Error 500 messages.

Tip: You can also write a visibility rule to hide the portlet from mobile devices:

- Log in to WebSphere Portal Express.
- Go to **Personalization**.
- Create the following rule to hide your portlet:

```
Hide page or portlet when
    current Device.Device Class  includes  smartphone  or
    current Device.Device Class  includes  tablet
  Otherwise show
```

- To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
- Find your page and click **Edit layout**.
- Go to the web dock application that you want to apply the rule to.

  Note: If necessary, you need to apply this rule to every web dock application that you created.
- Select **Show Portlet Rule Mapping**.
- Select the rule that you created.
- Save your changes.

Edit Shared Settings does not work on mobile devices
The web application bridge does not support the **Edit Shared Settings** mode when accessed through a mobile device.

Reverse Proxy servlet
- No Persistent Cookie handling (all cookies are treated as sessions)
- No special HTTP 1.0 support
- Caching that is based on HTTP headers not yet implemented

The Portal search does not work for WAB - iFrame content
If the search crawler indexes a page, the page might not be available for search in a later session. Therefore, make sure that the Search crawler user ID does not have access to the web application bridge.

Sharepoint integration
The **View RSS feed** option is not supported.

Review the following questions to help you integrate and troubleshoot your web application bridge integration:

Can I use the localhost or IP address of an application to integrating it with the web application bridge?
Always specify fully qualified host name of your application. Do not use the localhost or IP address. You must also use the fully qualified host name when you access WebSphere Portal Express.

What request headers are always propagated?
- Date
- Pragma
- Via
- Accept
- Accept-Charset
- Accept-Encoding
- Accept-Language
- From
- Referrer
- Allow
- Content-Encoding
- Content-Language
- Content-Length
- Content-Location
- Content-MD5

- Content-Type
- User-Agent

What request headers are always blocked?

- Host
- Authorization
- Range
- Connection
- Keep-Alive
- Proxy-Authorization
- TE
- Trailers
- Transfer-Encoding
- Upgrade

What response headers are always propagated?

- Date
- Pragma
- Via
- Server
- Allow
- Content-Encoding
- Content-Language
- Content-Length
- Content-Location
- Content-MD5
- Content-Type

What response headers are always blocked?

- Location
- WWW-Authenticate
- Proxy-Authenticate
- Accept-Ranges
- Content-Range
- Connection
- Trailers
- Transfer-Encoding
- Upgrade

What do I do if I consistently see an error page in the view mode?

Log out of WebSphere Portal Express. Clear all cookies and log back in to WebSphere Portal Express. Use the fully qualified host name.

How do I remove the Sharepoint navigation that shows in the Web Dock portlet?

Review your Sharepoint documentation about hiding the navigation pane without changing the master page. You can also add filters to your application to parse the content that is not required.

What sites create integration problems?

The following types of sites can cause integration problems:

- Sites that have absolute URLs in the content or JavaScript
- Sites that use JavaScript, META refresh, absolute URLs for redirection
- Sites that use JavaScript to ensure that a web application is within the top frame and not an embedded frame, for example:

  ```
  if(top.location.href!=self.location.href){top.location.href=self.location.href
  ```

What is the unit for specifying height and width in the Web Dock portlet?
Height and width are specified in pixels. The page theme might restrict the width to maximum limit allowed by the theme.

How can I enable tracing for the web application bridge?
Add the following string to the Enable Tracing portlet in WebSphere Portal Express:

```
com.ibm.wps.vwat.*=all:com.ibm.wps.wab.*=all
```

Why can I not access my internet mail application?
If you are using an internet application that has complex security features, the web application bridge might not be able to access the application. One example of an internet application with complex security features is Yahoo. If you are unable to access your internet application and you verified that your settings in the application component are correct, contact Support.

Browse page never loads and keeps cycling
If your web browser never loads and keeps cycling, your server times might not match. If you are using Firefox, you see a message that your portal session timed out and to log in. To fix this issue, match your server time to the application you are bridging and restart portal.

Public URLs with multiple host do not display properly
If the public site fetches content from multiple hosts, then such sites do not render properly on the Web Dock portlet. For example, Wikipedia loads content from the following different hosts:

- `en.wikipedia.org`
- `bits.wikimedia.org`
- `upload.wikimedia.org`
- `meta.wikimedia.org`

Cookies that are exchanged between the web server and the application server
Setting the cookies that are exchanged between the web server and the application server to **HTTP only** disables the web application bridge.

Does WAB support SAML (Security Assertion Markup Language)?
If you have CF05 or earlier, WAB does not support SAML, unless the SAML token is inside of a cookie and the Portal server and the target server are in the same domain. The client-side cookie forwarding feature can be used in such a scenario. For example, the name of the cookie needs to be specified in the **Policy** > **Request** > **HTTP Cookies** section of the VWA Manager portlet.

Note: CF06 Starting with CF06, WAB does support SAML for single sign-on authentication.

What if the context root for the application that needs to be integrated matches the context root of another application that is already installed on the portal server?
You want to integrate an application whose URI starts with `/abc`. A second application exists on the Portal server with the same context root: `/abc`. If

the first application needs to be integrated into portal through the web application bridge, then the second application needs to be stopped or removed from the portal server.

I see too many pollings requests when I integrate with Netweaver
This issue might be because some headers are not being forwarded to the Netweaver server. These headers are **is_icm_polling_user**, **userId**, and **Content-Type**. Open the Virtual Web Application Manager portlet and add these headers to the list of allowed headers in the policy for Netweaver.

Using Simple and Protected GSS-API Negotiation (SPNEGO) on Windows
With SPNEGO-based single-sign on, the credentials are always automatically picked. Therefore, the user does not need to provide the credentials in the **Personalize** mode. They are always the user who is logged in to Windows.

Single sign-on
It is expected that the login buttons or similar things like "Switch user" buttons of the backend application are not available. They are either removed with the filters or the backend applications are directly configured. In other words, the content author can provide the credentials only through the **Personalize** mode of the web dock application portlet.

Single sign-on settings
The Web Application Bridge stores the authentication cookies that are received from the backend application in session. Perform authentication with the backend server only once per session. You do not need to authenticate for every request. After the user logs in to the backend server with single sign-on through WAB, the authentication remains active for the entire session. Even if the single sign-on settings in the policy change for the backend application, the user is still authenticated until the user logs out of portal. Therefore, if you change the policy, make sure that you log out of portal and then log back on to completely change the policy settings.

Integrating with SAP NetWeaver Portal

You can use IBM WebSphere Portal Express Integrator for SAP to integrate content from an SAP NetWeaver Portal into your IBM WebSphere Portal Express. You can integrate navigational structures and single content pieces.

About this task

Integrator for SAP integrates the SAP NetWeaver Portal navigation into WebSphere Portal Express. The navigation is retrieved new from the SAP NetWeaver Portal for each WebSphere Portal Express session and used for the duration of that session.

By default, the installation names the SAP NetWeaver Portal integration page **SAP**. WebSphere Portal Express users can navigate to this page by selecting **Applications** > **IBM WebSphere Portal Express Integrator for SAP** > **SAP**.

The following topics describe the prerequisites and procedure for setting up Integrator for SAP.

"Prerequisites and support for Integrator for SAP" on page 987
Integrator for SAP works with the prerequisites and under the conditions and limitations listed here.

Prerequisites and support for Integrator for SAP

Integrator for SAP works with the prerequisites and under the conditions and limitations listed here.

1. For the integration, use Integrator for SAP only with SAP NetWeaver Portal Version 7.3 SP2 with note 1638641 *Interoperability fix for SAML1.1 assertions in web services*. For more information about this note see the web link section. Note that you need a user ID and password from SAP to access this information.
2. Make sure that both WebSphere Portal Express and SAP NetWeaver Portal are installed in the same domain, for example *example*.com and have direct access to each other.
3. For virtual portals, WebSphere Portal Express supports only one integration point per virtual portal.
4. Use only the browsers and browser versions that are supported by both WebSphere Portal Express and SAP NetWeaver Portal. For details about browser support, read the appropriate documentation:
   - For WebSphere Portal Express, read *System requirements*.
   - For SAP NetWeaver Portal, read the product documentation for that product.
5. To configure single sign-on (SSO) between WebSphere Portal Express and SAP NetWeaver Portal, you can use either of the following options:
   - HTTP Basic Authentication for single sign-on (SSO) to SAP NetWeaver Portal. To configure HTTP Basic Authentication for single sign-on, configure Basic Authentication for SSO for the SAP navigation integration. For details, read *Configuring Integrator for SAP*.
   - A Security Assertion Markup Language (SAML) environment using IBM Tivoli Federated Identity Manager. For the installation and setup instructions, read the product documentation for IBM Tivoli Federated Identity Manager.

The navigation integration is temporary for each session and not imported or persisted in WebSphere Portal Express. Therefore the following limitations apply:

- Users cannot bookmark integrated pages, or tag or rate integrated content.
- If a search crawler indexes a page, the page might not be available for search by users in a later WebSphere Portal Express session. Therefore make sure that the WebSphere Portal Express Search crawler user ID does not have access to the SAP integration content.

Related tasks:

"Preparing your system environment and the prerequisites for Integrator for SAP"
To prepare your WebSphere Portal Express and the prerequisites for installing Integrator for SAP, make sure that you have all the required files and configure your outbound HTTP connections.

"Configuring Integrator for SAP" on page 991
Before you can use the SAP navigation, you need to perform the configuration.

Related information:

"System requirements" on page 103
Before you install IBM WebSphere Portal Express, review the hardware and software requirements to ensure that you have the supported versions of prerequisite and corequisite software and the required hardware.

Preparing your system environment and the prerequisites for Integrator for SAP

To prepare your WebSphere Portal Express and the prerequisites for installing Integrator for SAP, make sure that you have all the required files and configure your outbound HTTP connections.

Procedure

1. Configure your outbound HTTP connections:
   a. Copy the following sample code and save it in a file named `proxy-config.xml` in a temporary directory on your WebSphere Portal Express server:

```
<?xml version="1.0" encoding="UTF-8"?>
<proxy:proxy-rules
     xmlns:proxy="http://www.ibm.com/xmlns/prod/sw/ajax/proxy-config/1.0"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <proxy:mapping url="*" contextpath="/proxy"/>
    <proxy:mapping url="*" contextpath="/myproxy"/>
    <proxy:mapping url="*" contextpath="/common_proxy"/>
    <proxy:policy  url="xample_sap_portal.company.com:50000/*"
                   basic-authsupport="true" acf="none">
        <proxy:actions>
            <proxy:method>GET</proxy:method>
            <proxy:method>HEAD</proxy:method>
        </proxy:actions>
        <proxy:cookies>
            <proxy:cookie>MYSAPSSO2</proxy:cookie>
        </proxy:cookies>
        <proxy:headers>
            <proxy:header>User-Agent</proxy:header>
            <proxy:header>Accept*</proxy:header>
            <proxy:header>Content*</proxy:header>
            <proxy:header>Authorization*</proxy:header>
            <proxy:header>set-cookie</proxy:header>
        </proxy:headers>
    </proxy:policy>
    <proxy:meta-data>
        <proxy:name>socket-timeout</proxy:name>
        <proxy:value>10000</proxy:value>
```

```
    </proxy:meta-data>
    <proxy:meta-data>
        <proxy:name>retries</proxy:name>
        <proxy:value>2</proxy:value>
    </proxy:meta-data>
    <proxy:meta-data>
        <proxy:name>max-connections-per-host</proxy:name>
        <proxy:value>5</proxy:value>
    </proxy:meta-data>
    <proxy:meta-data>
        <proxy:name>max-total-connections</proxy:name>
        <proxy:value>100</proxy:value>
    </proxy:meta-data>
    <proxy:meta-data>
        <proxy:name>forward-credentials-from-vault</proxy:name>
        <proxy:value>true</proxy:value>
    </proxy:meta-data>
</proxy:proxy-rules>
```

For more information about configuring your outbound HTTP connection services, read *Configuring outbound HTTP connections*.

b. If you do not use Basic Authentication for single sign-on, remove the references to Basic Authentication from the file `proxy-config.xml` .

c. In the file `proxy-config.xml`, set the values for the parameters **`socket-timeout`** and **`retries`** according to your environment. If your SAP NetWeaver Portal is not available for some reason, these parameters determine the amount of time that the task spends on the inaccessible connection. WebSphere Portal Express tries a connection once for each user who logs in to the WebSphere Portal Express and who has access rights to the SAP navigation.

   Note: If the connection fails for many users, the failures can affect the performance of WebSphere Portal Express.

d. In the file `proxy-config.xml`, replace the proxy URL with your SAP NetWeaver Portal host and port. Example: `http://example_sap_portal.company.com:50000` .

e. In the file `proxy-config.xml`, make sure to add your SSO token name to the cookie section of the SAP NetWeaver Portal host. For example, the token name can be `MYSAPSSO2` .

f. If you want to run the configuration task in the following step without specifying passwords, add the user IDs and passwords for WebSphere Application Server and WebSphere Portal Express to the file `wp_profile/ConfigEngine/properties/wkplc.properties` .

g. Run the WebSphere Portal Express configuration task `checkin-wp-proxy-config` as follows:

   - If you added user IDs and passwords to the file `wp_profile/ConfigEngine/properties/wkplc.properties` , enter the task as follows:

     ```
     ConfigEngine.bat|sh create-outbound-http-connection-config
                         -DConfigFileName=/proxy-config.xml
                         -DOutboundProfileType=global
     ```

   - If you want to specify the user IDs and passwords when you run the configuration task, enter the task as follows:

     ```
     ConfigEngine.bat|sh create-outbound-http-connection-config
                         -DConfigFileName=/proxy-config.xml
                         -DOutboundProfileType=global
                         -DWasPassword=password
                         -DPortalAdminPwd=password
     ```

For more information, read *Configuring outbound HTTP connections by using configuration tasks*.

2. Optional: If you do not have a page with the unique name `ibm.portal.page.Applications` in your WebSphere Portal Express, create it before you install Integrator for SAP. The installation process expects to find a page with the unique name `ibm.portal.page.Applications` in WebSphere Portal Express. It adds integration artifacts to this page as child pages. If you do not have this page in your WebSphere Portal Express and run the Solution Installer install task, an XMLAccess exception occurs.

Results

You have completed the preparation for Integrator for SAP.

Related tasks:

"Installing Integrator for SAP"
Integrator for SAP is delivered as a portal application archive (PAA) file. You find it under the following location: *PortalServer_root*/base/wp.integration/sap.package/installableApps/sap_integration.paa . You install and deploy the application by using WebSphere Portal Express Solution Installer.

"Configuring outbound HTTP connections" on page 2992
In WebSphere Portal Express Version 8.0 and earlier versions, outbound HTTP connections were accessible through the Ajax Proxy service. The Ajax Proxy service was configured by a configuration document named `proxy-config.xml`. You find this document in the `/WEB-INF` directory of the web module that uses the Ajax Proxy service. Starting with WebSphere Portal Express Version 8.5 and the new outbound connection service, the configuration of outbound HTTP connections is now part of the standard datastore-based portal configuration.

"Configuring outbound HTTP connections by using configuration tasks" on page 3013
Programmers can create, read, update, or delete settings of the outbound HTTP connection by using the appropriate portal configuration engine tasks.

Installing Integrator for SAP

Integrator for SAP is delivered as a portal application archive (PAA) file. You find it under the following location: *PortalServer_root*/base/wp.integration/sap.package/installableApps/sap_integration.paa . You install and deploy the application by using WebSphere Portal Express Solution Installer.

About this task

For details about the Solution Installer, read the section about *Installing add-ons*.

Note: The installation and deployment process expects to find a page with the unique name `ibm.portal.page.Applications` in WebSphere Portal Express. It adds integration artifacts to that applications page as child pages. If you do not have this applications page in your WebSphere Portal Express and run the Solution Installer install task, an XMLAccess exception occurs. In this case create a page with the unique name `ibm.portal.page.Applications` and rerun the task that failed.

Procedure

1. Install the Integrator for SAP PAA file by using the portal Solution Installer. For this installation step, run the installation configuration task as follows:

Linux `./ConfigEngine.sh install-paa -DPAALocation=/sap_integration.paa`

IBM i `ConfigEngine.sh install-paa -DPAALocation=/sap_integration.paa`

Windows
`ConfigEngine.bat install-paa -DPAALocation=/sap_integration.paa`

2. Deploy the Integrator for SAP PAA file by using the portal Solution Installer. For this deploy step, run the deployment configuration task, and specify the value `sap_integration` for the parameter `appName` as follows:

   Linux `./ConfigEngine.sh deploy-paa -DappName=sap_integration`

   IBM i `ConfigEngine.sh deploy-paa -DappName=sap_integration`

   Windows
   `ConfigEngine.bat deploy-paa -DappName=sap_integration`

3. Restart your WebSphere Portal Express for the installation to become active. You must restart your portal before you configure Integrator for SAP.

Related tasks:

"Configuring Integrator for SAP"
Before you can use the SAP navigation, you need to perform the configuration.

"Installing add-ons" on page 210
You can use the Solution Installer through the Configuration Wizard to install and uninstall add-ons to an IBM WebSphere Portal Express server instance. The Solution Installer uses the Portal Application Archive (PAA) format as the standard format for application distribution. Portal Application Archive (PAA) updates are not supported in the configuration wizard currently. For more information about updating add-ons using a command prompt, see the Managing your existing PAA file section.

Configuring Integrator for SAP

Before you can use the SAP navigation, you need to perform the configuration.

About this task

The SAP navigation integration uses a set of configuration parameters, but you must configure these parameters separately:

Procedure

Configuring the SAP navigation page: By default, the WebSphere Portal Express installation names the SAP integration page **SAP** and places it as a subpage of **Applications** > **IBM WebSphere Portal Express Integrator for SAP**.
If you want to change the title of the SAP navigation page or move the page to another location in your WebSphere Portal Express, use the standard tools of WebSphere Portal Express. For example, you can use **Manage Pages** in the WebSphere Portal Express administration.
All required page properties for configuring the SAP navigation page are stored in the page properties of the SAP navigation page. For a list of configuration parameters see the topic about *Configuration parameters for the SAP navigation integration*. You must check and verify these settings and change them as appropriate for your environment, if required. To perform this task, proceed as follows:

1. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.

2. Click **Content root** > **Applications** > **IBM WebSphere Portal Integrator for SAP**.
3. Click **Edit page properties** for the page that is named **SAP**, or however you might have renamed it.
4. Select **Advanced options** > **I want to set parameters**.
5. Change the page properties as required. For a list of configuration parameters, read the topic about *Configuration properties for the SAP navigation integration*. If your values for the page properties remain within a length limit of 255 characters, you can set them directly in the page properties as described here. If the values for any of the properties exceed 255 characters, you must set that property in the WP Configuration Service, reference it in the page properties, and prefix its value in the page properties with `ConfigService:` . For example, the most likely parameter to have a value that exceeds 255 characters is `sap.SSOTokenUrl` . In this case you configure the following settings:
   - In the WP Configuration Service: *`actual.SSO.tokenUrl = "your_URL"`*
   - In the page properties: `sap.SSOTokenUrl = "ConfigService:`*`actual.SSO.tokenUrl`*`"`

   For details about how portal service configuration properties and how to set them, read *Configuration Service* and *Setting service configuration properties*. For the parameter `sap.SSOTokenUrl`, you can specify a page URL of the SAP portal of your choice, except the `sap.BaseUri`.
6. Restrict the access to the SAP navigation page to the correct audience, for example to all or selected SAP users. To perform this task, use the WebSphere Portal Express Access Control.
7. Optional: Like portal pages, you can configure the navigational integration label to use a theme template. Within the navigational integration, integrated SAP NetWeaver portal pages inherit that configuration from the label. To perform this configuration, add the page parameter `com.ibm.portal.theme.template.file.name.htm` to the label by using the XMLAccess configuration interface. After you install Integrator for SAP, all integration pages are already configured to use a side navigation theme template. You can remove the side navigation by removing the parameter for the side navigation theme template from the label.

What to do next

After you have completed the configuration, restart your WebSphere Portal Express server for your changes to take effect.

For more information about how to configure Integrator for SAP see these topics:

"Configuring Basic Authentication for SSO for the SAP navigation integration" on page 993
For single sign-on between WebSphere Portal Express and SAP NetWeaver Portal, you can configure HTTP Basic Authentication using the Credential Vault.

"Configuring Tivoli Federated Identity Manager with SAML for single sign-on to SAP NetWeaver Portal" on page 994
You can also use Tivoli Federated Identity Manager with Security Assertion Markup Language (SAML) for single sign-on to SAP NetWeaver Portal.

"Configuring logout handling" on page 995
When a user logs out of WebSphere Portal Express, a log out from SAP NetWeaver Portal needs to be performed as well. Otherwise, the user session on the SAP NetWeaver Portal remains open until it times out.

"Completing the configuration" on page 996
After you have completed configuring Integrator for SAP, restart your WebSphere Portal Express server for your changes to take effect.

"Page properties for configuring the SAP navigation integration" on page 997
To configure the SAP navigation integration of Integrator for SAP, you can set the following page configuration properties.

Related concepts:

"Configuring authentication filters" on page 265
The portal authentication filters are a set of plug-in points. You can use them to intercept or extend the portal login, logout, session timeout, and request processing by custom code, for example to redirect users to a specific URL.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Related reference:

"Performance tuning for Integrator for SAP" on page 998
See the following hints and tips that might help improve performance of your Integrator for SAP.

"Page properties for configuring the SAP navigation integration" on page 997
To configure the SAP navigation integration of Integrator for SAP, you can set the following page configuration properties.

"Configuration Service" on page 297
The portal Configuration Service is responsible for collecting the most essential configuration data of the IBM WebSphere Portal Express engine.

Configuring Basic Authentication for SSO for the SAP navigation integration

For single sign-on between WebSphere Portal Express and SAP NetWeaver Portal, you can configure HTTP Basic Authentication using the Credential Vault.

About this task

For you to be able to perform this configuration, the SAP Navigation WS must be running and accessible. This should be given by default in a SAP NetWeaver portal installation.

To configure HTTP Basic Authentication for SSO, proceed as follows:

Procedure

1. Access the portal with an administrative user ID.
2. Create a Credential Vault slot that can later store a user's credentials by using the WebSphere Portal Express administration. For more information, read *Credential Vault*.
3. Configure HTTP Basic Authentication for SSO for Integrator for SAP:
   a. Set the page parameter for the SAP integration page `sap.CredentialSlotId` to the name of the Credential Slot that you created in the previous step.
   b. Set the parameter `sap.SSOTokenUrl` to a URL in your SAP NetWeaver Portal.

   For details see the topics *Configuring Integrator for SAP* and *Configuration parameters for the SAP navigation integration*.

4. Users must add their credentials to the slot in the Credential Vault Dialog. They can access the Credential Vault Dialog by typing the web address of the Credential Vault Dialog into a web browser. For example, `http://<host>:<port>/wps/mypoc?uri=cvfiller:<credentialVaultSlotName>`.
5. Optional: If you do not want users to be able to edit the user ID and password credentials that the integrator portlet uses with Basic Authentication, then you can revoke the Privileged User role at the portlet for these users. You do this by using the WebSphere Portal Express Access Control. This can be useful if you use a shared Credential Vault slot and a group of users share the same user ID and password for accessing the SAP NetWeaver Portal.
6. Optional: Configure single sign-on with the SAP navigation integration for browsers. If you configure HTTP Basic Authentication for single sign-on, Integrator for SAP provides single sign-on between WebSphere Portal Express and the SAP NetWeaver Portal navigation only. This means that users can see the integrated navigation, but when they access an integrated page, SAP NetWeaver Portal prompts them for authentication, if SSO is not implemented by other means. You can include browsers in the configuration of this single sign-on. If you want WebSphere Portal Express to pass the SAP NetWeaver Portal authentication token to the user's browser, you must perform both of the following tasks:
   a. Set the page parameter `sap.SSOTOkenDomain` to the domain for which you want to set the token. For details, read the topic about *Configuration properties for the SAP navigation integration*.
   b. Configure the following login and logout filters in the Resource Environment Provider WP Authentication Service:

```
login.explicit.filterchain    com.ibm.wps.integration.sap.login.LoginFilter
login.implicit.filterchain    com.ibm.wps.integration.sap.login.LoginFilter
logout.explicit.filterchain   com.ibm.wps.integration.sap.logout.LogoutFilter
logout.implicit.filterchain   com.ibm.wps.integration.sap.logout.LogoutFilter
```

      For details, read the topic *Configuring authentication filters*.

What to do next

Note that configuring single sign-on with the SAP navigation integration for browsers is supported only for HTTP Basic Authentication.

Related concepts:

"Credential Vault" on page 1523
The Credential Vault is a service that stores credentials that allow portlets to log in to applications outside the realm on behalf of the user. It manages multiple identities for portlets and users.

"Configuring authentication filters" on page 265
The portal authentication filters are a set of plug-in points. You can use them to intercept or extend the portal login, logout, session timeout, and request processing by custom code, for example to redirect users to a specific URL.

Related reference:

"Page properties for configuring the SAP navigation integration" on page 997
To configure the SAP navigation integration of Integrator for SAP, you can set the following page configuration properties.

Configuring Tivoli Federated Identity Manager with SAML for single sign-on to SAP NetWeaver Portal

You can also use Tivoli Federated Identity Manager with Security Assertion Markup Language (SAML) for single sign-on to SAP NetWeaver Portal.

About this task

In such a scenario, Tivoli Federated Identity Manager with SAML is responsible for handling the authentication flow by using Security Assertion Markup Language. For the SAP integration into WebSphere Portal Express, the supported SAML scenario is named *Service Provider initiated single sign-on*. To use such a scenario, you need technical expertise for all three participating systems: IBM WebSphere Portal, IBM Tivoli Federated Identity Manager, and SAP NetWeaver Portal.

To use Tivoli Federated Identity Manager (Tivoli Federated Identity Manager) for single sign-on to SAP NetWeaver Portal with Integrator for SAP, follow these instructions:

Procedure

- Make sure that your Tivoli Federated Identity Manager is configured correctly for authentication of the participating service providers and the users in a service-provider initiated single sign-on scenario. The service providers are the SAP NetWeaver Portal instance and the WebSphere Portal Express instance.
  - For the navigation integration, you must set up a Web Service Single Sign On for the Web Service Client **NavigationWS**. This Web Service Client is hosted in the enterprise application **IntegrationSAP** in the WebSphere Integrated Solutions Console.
  - For the SAP navigation integration, you must set up Web Single Sign On to the SAP NetWeaver Portal.
- To make the Integrator for SAP, use Tivoli Federated Identity Manager do not set any other authentication configuration:
  - For the SAP navigation integration, do not set the parameters `sap.CredentialSlotId` and `sap.SSOTokenUrl`. Also, do not configure single sign-on for browsers as described under the topic about *Configuring basic authentication for single sign-on to SAP NetWeaver Portal*.
  - Do not add the login or logout filter of the SAP integration to the filter chains.
- To test and verify your environment use the SAP navigation integration. This test requires that the web service single sign-on is configured.

Related tasks:

"Configuring logout handling"
When a user logs out of WebSphere Portal Express, a log out from SAP NetWeaver Portal needs to be performed as well. Otherwise, the user session on the SAP NetWeaver Portal remains open until it times out.

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Related reference:

"Configuration Service" on page 297
The portal Configuration Service is responsible for collecting the most essential configuration data of the IBM WebSphere Portal Express engine.

Configuring logout handling

When a user logs out of WebSphere Portal Express, a log out from SAP NetWeaver Portal needs to be performed as well. Otherwise, the user session on the SAP NetWeaver Portal remains open until it times out.

About this task

You can configure automatic log out from SAP NetWeaver Portal when a user logs out of WebSphere Portal Express. To perform this configuration, insert the following JavaScript function into your theme and call it on click of the logout link. Replace *your_sap_portal_host* and *port* with the values for your SAP NetWeaver Portal host and port.

```
function logoffFinalCall() {

   if (document.cookie.length>0) {
      isCookieExisting=document.cookie.indexOf("MYSAPSSO2")
      if (isCookieExisting != -1) {
         var lnDotPos = document.domain.indexOf( "." );
         if(lnDotPos>=0)document.domain = document.domain.substr(lnDotPos+1);

         var logoffForm = document.createElement("form");

         var logoffParam = document.createElement("input");
         logoffParam.name = "Command";
         logoffParam.value = "LOGOFF";
         logoffForm.appendChild(logoffParam);

         var logoffParam2 = document.createElement("input");
         logoffParam2.name = "Autoclose";
         logoffParam2.value = "1000";
         logoffForm.appendChild(logoffParam2);

   //This component logs the user out of SAP NetWeaver portal.
      logoffForm.action =
      "http://your_sap_portal_host:port/irj/servlet/prt/portal/prtroot/com.sap.portal.dsm.Terminator";

         logoffForm.method = "post";
         logoffForm.target = "_blank";
         document.body.appendChild(logoffForm);

         logoffForm.submit();
      }
   }
}
```

Replace the variables as follows:

MYSAPSSO2
: Replace this variable with the name of the SSO cookie for your SAP NetWeaver Portal.

your_sap_portal_host
: Replace this variable with the fully qualified URL to your SAP NetWeaver Portal.

port
: Replace this variable with the port of your SAP NetWeaver Portal.

Related tasks:

"Completing the configuration"
After you have completed configuring Integrator for SAP, restart your WebSphere Portal Express server for your changes to take effect.

Completing the configuration

After you have completed configuring Integrator for SAP, restart your WebSphere Portal Express server for your changes to take effect.

About this task

Users can now navigate to SAP NetWeaver Portal content as configured determined by their access rights. To navigate to SAP content, users select **Applications** > **IBM WebSphere Portal Express Integrator for SAP**, and then **SAP**.

Related reference:

"Page properties for configuring the SAP navigation integration" on page 997
To configure the SAP navigation integration of Integrator for SAP, you can set the following page configuration properties.

Page properties for configuring the SAP navigation integration

To configure the SAP navigation integration of Integrator for SAP, you can set the following page configuration properties.

For details about where and how to set these properties, read the topic about *Configuring your Integrator for SAP*.

sap.BaseUri

This property is mandatory. Use this property to specify the base URI to the SAP NetWeaver Portal. Example: `http://sapportal.company.com:50000` . This property has no default.

sap.CredentialSlotId

This property is mandatory only if you use Basic Authentication for SSO. Use this property to specify the name of the Credential Vault slot that you want to use for authentication to the SAP NetWeaver Portal.

sap.SSOTokenUrl

This property is mandatory only if you use Basic Authentication for SSO and if you have created a Credential Vault slot for this authentication method. In this case use this property to specify the absolute URL to a protected resource of your choice in your SAP NetWeaver Portal, for example a specific page or iView. This URL is used for login to retrieve the SSO token. During retrieval of the SSO token, WebSphere Portal Express follows all HTTP redirects. If you use a SAML scenario, you do not need to set this property.

sap.SSOTokenDomain

This property is mandatory only if you use Basic Authentication for SSO. Add this property if you want to pass also the SSO token defined by the property `sap.SSOTokenName` from WebSphere Portal Express to the client browser. If you do so, the integration also authenticates the clients that use the configured SSO scenario between WebSphere Portal Express and SAP NetWeaver Portal. To enable this authentication by token, specify the domain for which you want to set the token, starting with a dot, for example `.ibm.com` . This property has no default. If you set this property, you also must do the following:

- Add the login filter implementation `com.ibm.wps.integration.sap.login.LoginFilter` to both the explicit **and** implicit login filter chains.
- Add the logout filter implementation `com.ibm.wps.integration.sap.logout.LogoutFilter` to both the explicit **and** implicit logout filter chains.

For details see the topic about *Configuring authentication filters*.

sap.SSOTokenName = (MYSAPSSO2)

This property is optional. Use it only if you use Basic Authentication for SSO. Use this property to specify the SSO token name of your SAP NetWeaver Portal. If you use the Credential Vault, use this property for authentication of the web service call. The default value is `MYSAPSSO2` .

sap.NavUri = (/NavigationWS/NavigationWSConfig?style=document)

This property is optional. Use this property to specify the relative URI of the SAP NetWeaver Portal navigation web service. The default value is `/NavigationWS/NavigationWSConfig?style=document` .

sap.InteropUri = (/irj/portal/interop)
This property is optional. Use this property to specify the relative URI of the SAP Interop service. The default value is `/irj/portal/interop` .

sap.NavUriTimeout
This property is optional. Use this property to specify a timeout in seconds for the web service call. The default value is `5` .

sap.ClientSideLogging = (false)
This property is optional. Use this property to determine whether WebSphere Portal Express gives out client-side JavaScript debugging messages to the JavaScript console. This is helpful if problems with the automatic resizing of the iframe occur. If you want the portal to give out client-side JavaScript debugging messages to the JavaScript console, set this property to `true` . Restart the WebSphere Portal Express server for the change to become active.

Related concepts:

"Configuring authentication filters" on page 265
The portal authentication filters are a set of plug-in points. You can use them to intercept or extend the portal login, logout, session timeout, and request processing by custom code, for example to redirect users to a specific URL.

Related tasks:

"Configuring Integrator for SAP" on page 991
Before you can use the SAP navigation, you need to perform the configuration.

Performance tuning for Integrator for SAP

See the following hints and tips that might help improve performance of your Integrator for SAP.

General considerations

- Cache sizes have a direct impact on the memory requirements of WebSphere Portal Express, specifically the demands on the Java heap. To determine if your portal has enough memory resources available to handle an additional increase, monitor the usage of caches and the portal memory usage under a heavy workload, before you increase the cache sizes.
- If you do not use the portlet menu for IBM API portlets, disable it. To do this, proceed as follows:
  1. Access the WebSphere Integrated Solutions Console.
  2. Navigate to **Resources** > **Resource Environment** > **Resource Environment Providers**.
  3. Select **WP ConfigService**.
  4. Add the following custom property:
     - Property name: `navigation.portletmenu.mode`
     - Value: `0`

Limiting availability of the SAP navigation page for users

Each time a user accesses the SAP navigation page for the first time during a WebSphere Portal Express session, WebSphere Portal Express sends a request to the SAP NetWeaver Portal. Therefore limit the access to the SAP navigation page to the correct audience, for example, to all or selected SAP users. To perform this task, use the WebSphere Portal Express Access Control. This measure limits request volume to the appropriate requests only. For more information about Portal Access

Control refer to the topic *Controlling access*.

Caches for performance and memory consumption

For tuning purposes, Integrator for SAP provides two portal caches with entries for each integrated SAP portal page and for each logged in user. Set their size and lifetime according to your environment.

com.ibm.wps.integration.sap.NodeCache
Content: This cache holds one entry for each integrated page per locale, independent of the user. For example, if your SAP NetWeaver Portal contains a maximum of 500 pages and all your users use one and the same locale, this cache can never exceed 500 entries.

com.ibm.wps.integration.sap.NodeCache.size = (1000)
Default size: 1000 entries

com.ibm.wps.integration.sap.NodeCache.lifetime = (-1)
Default lifetime: This cache never expires.

com.ibm.wps.integration.sap.ModelCache
Content: This cache holds one entry for each logged in user who has access to the SAP navigation. This cache entry is removed on logout by the user.

If you have only limited memory available, you can use this cache to limit the memory consumption of the SAP navigation integration. However, limiting memory consumption this way might in turn affect performance. Note that if you add pages to SAP NetWeaver Portal, the memory consumption grows accordingly, even if you do not increase this cache size. If you set this cache size too small, performance might decrease.

com.ibm.wps.integration.sap.ModelCache.size = (1000)
Default size: 1000 entries

com.ibm.wps.integration.sap.ModelCache.lifetime = (-1)
Default lifetime: This cache never expires.

For more information about how to configure WebSphere Portal Express caches, read *Cache Manager Service* and *Setting service configuration properties*.

Ajax Proxy configuration

Set appropriate values in the file `proxy-config.xml` according to your environment for the maximum number of connections and the number of connections per host. Otherwise, a limitation of available connections can occur.

Set appropriate values for the parameters `socket-timeout` and `retries` . If SAP NetWeaver Portal is not available for some reason, these two parameters limit the length of time that the task spends on the inaccessible connection. For details about the two parameters read*Preparing your system environment and the prerequisites*. For each user who logs in to WebSphere Portal Express and who has access rights to the SAP NetWeaver Portal navigation page, WebSphere Portal Express tries a connection at least once, depending on the cache settings described in the previous section. If these attempts fail for many users, it can affect the performance of WebSphere Portal Express.

For more information about configuring your Ajax Proxy, read *Configuring Outbound HTTP Connections*.

Related tasks:

"Preparing your system environment and the prerequisites for Integrator for SAP" on page 988
To prepare your WebSphere Portal Express and the prerequisites for installing Integrator for SAP, make sure that you have all the required files and configure your outbound HTTP connections.

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

"Configuring outbound HTTP connections" on page 2992
In WebSphere Portal Express Version 8.0 and earlier versions, outbound HTTP connections were accessible through the Ajax Proxy service. The Ajax Proxy service was configured by a configuration document named `proxy-config.xml`. You find this document in the `/WEB-INF` directory of the web module that uses the Ajax Proxy service. Starting with WebSphere Portal Express Version 8.5 and the new outbound connection service, the configuration of outbound HTTP connections is now part of the standard datastore-based portal configuration.

Related reference:

"Hints and tips for Integrator for SAP"
Observe the following hints and tips in case of problems with Integrator for SAP.

"Configuration Service" on page 297
The portal Configuration Service is responsible for collecting the most essential configuration data of the IBM WebSphere Portal Express engine.

"Cache Manager Service" on page 292
The portal Cache Manager Service is responsible for managing the different caches used in WebSphere Portal Express.

Hints and tips for Integrator for SAP

Observe the following hints and tips in case of problems with Integrator for SAP.

Autosizing the SAP navigation page does not work for all themes in client-side mode

The automatic resize feature for the SAP navigation iframe does not work for static pages in client-side rendering mode. Under these conditions a height of 600 pixels applies to the SAP navigation page. Autosizing the SAP navigation page works for pages with all themes rendered in server-side mode.

This limitation does not apply to the integrator portlet IBM WebSphere Portal Express Integrator for SAP.

Replacement page in case of errors

If the SAP navigation retrieves an exception, it shows a replacement page as a child page to the navigation page. It shows a message such as the ones given in the following list. In this case check the system logs for the root cause of the exception. The following examples give some typical issues and their resolution.

EJQIA0017E: The Ajax Proxy returned HTTP: [403 Forbidden] when trying to get the SSO Token from URL

Reason: The Ajax Proxy blocked the request.

Action: Configure the Ajax Proxy. Verify the Ajax Proxy configuration by calling the Ajax Proxy directly: `http://`

your_IBM_WebSphere_Portal.domain.com:*port*/wps/*your_proxy*/http/
sap.ssoTokenURL-Value?hpaa.slotid=*your_Credential_Vault_Slot_ID*

EJQIA0017E: The Ajax Proxy returned HTTP: [502 Bad Gateway] when trying to get the SSO Token from URL

Reason: SAP NetWeaver Portal is not available.

Action: Make sure that SAP NetWeaver Portal is available.

EJQIA0017E: The Ajax Proxy returned HTTP: [504 Gateway Timeout] when trying to get the SSO Token from URL

Reason: SAP NetWeaver Portal is not available.

Action: Make sure that SAP NetWeaver Portal is available.

Related reference:

"Using Web Application Bridge"
If you plan to use IBM Web Application Bridge, you need to configure all remote context roots.

Using Web Application Bridge

If you plan to use IBM Web Application Bridge, you need to configure all remote context roots.

For the detailed steps see the documentation for Web Application Bridge.

Note: Web Application Bridge does not support SAML (Security Assertion Markup Language) unless the SAML token is inside a cookie, and the Portal server and the target server (SAP Netweaver portal in this case) are in the same domain. The client-side cookie forwarding feature can be used in such a scenario. For example, the name of the cookie must be specified in the **Policy** > **HTTP Cookies** section of the VWA Manager portlet.
To identify the context root on SAP NetWeaver Portal, consult your SAP NetWeaver Portal administrator. Note that the context roots depend on your environment and usage scenario. You can use the following list as a starting point in a review meeting with your SAP administrator.

```
/irj
/com.sap.portal.navigation.afp.tln
/com.sap.portal.navigation.afp.resources
/com.sap.portal.navigation.afp.pagetoolbar
/com.sap.portal.navigation.afp.masthead
/com.sap.portal.navigation.afp.layout
/com.sap.portal.navigation.afp.dynamicnavigation
/com.sap.portal.navigation.afp.dtn
/com.sap.portal.epcf.loader
/com.sap.portal.dsm
/com.sap.portal.design.urdesigndata
/com.sap.portal.design.portaldesigndata
/classes
/AFPServlet
/wsnavigator
/webdynpro
/utl
/useradmin
/ur
/sr_central
/rtmfCommunicator
/resources
/nwa
/logon_ui_resources
/js
/irj
```

```
/htmlb
/ejbexplorer
/common
/com.sap.ui.lightspeed
/com.sap.portal.runtime.gwtintegration
/com.sap.portal.pagebuilder
/com.sap.portal.navigation.objbased
/com.sap.portal.navigation.helperservice
/com.sap.portal.navigation.contentarea
/com.sap.portal.navigation.afp.widgets
```

Integrating with IBM MobileFirst

You can integrate WebSphere Portal Express with MobileFirst to provide multi-channel support to your web communities. You can create a hybrid application that adds native device functions and a unified web experience on mobile device browsers and in mobile device native applications. You can use MobileFirst to create a hybrid application that adds native device functions to your portal.

There are three types of applications you can create for your cross-platform environments with WebSphere Portal Express and MobileFirst.

Native applications
Native applications for enterprise mobile apps have the highest UI fidelity and range of function of the types of web applications included here, but have the highest cost to develop and maintain. Cross-platform issues can increase costs, and mobile operating system updates can require frequent updates to be submitted in application stores. MobileFirst provides tools to reduce costs.

Pure web applications
Pure web applications that are based on HTML5, CSS3, and JavaScript, or that use client-side frameworks like Dojo and jQuery, provide a simple way to make mobile-friendly websites. Development and maintenance costs are less expensive. You can work more easily across multiple devices and are not typically impacted by mobile operating system updates. WebSphere Portal Express provides a platform for pure web applications and websites. With a pure web application, you only have access to the native device features that the browser provides to you, and you are more limited in the user interface fidelity.

Hybrid applications
Hybrid applications combine the characteristics of pure and native applications. You can build an application with the simplicity of developing with HTML/CSS/JavaScript. But you can augment that with a wide range of native services and produce an application for application stores. With technology like Apache Cordova, which MobileFirst includes, you can call native features with JavaScript from your web markup. For example, you can call the camera with a simple JavaScript line, such as: `navigator.camera.getPicture`. MobileFirst provides tools for creating these hybrid applications. Hybrid applications are native applications that wrap the WebSphere Portal Express web application. The two are tightly integrated to use each other's capabilities to make the development experience as quick and easy as possible.

As a WebSphere Portal Express customer, you can use the MobileFirst tools for free and create two free applications. But the applications must be hybrid applications that use WebSphere Portal Express.

Planning to install IBM MobileFirst

Determine the services and function of your hybrid application before you install MobileFirst. You must install a MobileFirst server in some instances.

Running a hybrid application in a portal page

When your hybrid application runs with your WebSphere Portal Express pages rendered in a native application, WebSphere Portal Express loads the appropriate native resources for the device. These resources are loaded automatically through modules that are provided in WebSphere Portal Express. It starts with the `wp_worklight_ext` module, which is listed in some of the default profiles, including `profile_deferred.json`, `profile_dojo_deferred.json`, and `profile_basic_content.json`.

If you want access to the appropriate native resources for the device on a particular page of your WebSphere Portal Express, use a profile that includes the `wp_worklight_ext` module. The default profile, `profile_deferred.json`, includes `wp_worklight_ext`, so the appropriate native resources are available to your WebSphere Portal Express pages by default.

`wp_worklight` is a version-independent meta-module that is defined by the `mobilefirst70.json` file in your theme's contributions folder. This module is a prerequisite of the default MobileFirst resources that enable access to native features. It also includes the overrides that enhance performance and allow the API libraries to work within the module framework.

The version-dependent platform modules that are included by the module framework are `mf_ios_70` and `mf_android_70`. These platform modules are defined by the `plugin.xml` file in your theme's *PortalServer_root*`\theme\wp.theme.worklight.ext\installedApps\wp.theme.worklight.ext.ear\`

wp.theme.worklight.ext.war\WEB-INF folder. These platform modules load the appropriate native resources for the device, giving access to the full MobileFirst and Cordova APIs. For example, it gives access to the following resources on the device:

- Camera
- Geolocation
- Contacts
- Local storage
- Media
- Push notifications
- User information

These platform modules are loaded or not based on device class conditions, as you can see in the following plugin.xml code snippet for the mf_android_70 module:

```
<module id="mf_android_70">
         <runtimeActivation>
    <condition deviceClass="worklight+android"/>
   </runtimeActivation>
```

The mf_android_70 module loads if the device class is both MobileFirst and Android. The device class is determined by the WebSphere Portal Express server that is based on the user agent string of the client device. For example, a user agent string for an Android phone looks like this example:

```
Mozilla/5.0 (Linux; U; Android 4.0.4; en-gb; GT-I9300 Build/IMM76D) AppleWebKit/534.30 (KHTML, like
```

A MobileFirst hybrid application automatically appends "/Worklight/*version*" to the end of the user agent string, such as:

```
Mozilla/5.0 (Linux; U; Android 4.0.4; en-gb; GT-I9300 Build/IMM76D) AppleWebKit/534.30 (KHTML, like
```

Windows Phone MobileFirst applications cannot modify the user agent. Instead, WebSphere Portal Express sets a session cookie that is called wp.agent.ext to "/Worklight" whenever it detects a uri=wl:id request parameter, and appends that cookie's value to the agent before it evaluates device classes. This parameter must be present on the initial request from the hybrid application or MobileFirst is not available on Windows Phone devices.

Appropriate matching is used to determine the device classes from the user agent string. The device classes in turn determine the appropriate platform modules to load. For example, the mf_android_70 native resources are loaded for a portal that runs in a MobileFirst hybrid application on an Android device. But it does not load in many other cases. For example, if it is on an iOS device, or if it is a portal that is not wrapped in a MobileFirst hybrid application, these resources do not load.

The same portal pages adapt their capability that is automatically based on the context in which they are running. For example, a page can provide access to the device's camera if it runs in the context of a MobileFirst hybrid application. The same page cannot get access while it is running outside the context of a MobileFirst hybrid application.

The Cordova and MobileFirst API have overrides to improve performance and allow integration with WebSphere Portal Express. The overrides allow the MobileFirst Client API to find the resources in the deployed web application. The overrides also allow the Cordova plug-ins to be packaged into a module and allow

the multiple JavaScript resources to be fetched in one request by the resource aggregation framework.

Shell application

If your application is a shell that uses a web view to render all markup from the WebSphere Portal Express site, then you do not need a MobileFirst server.

If your application is using a mixed model approach where some of the application markup is coming from your WebSphere Portal Express and other markup of the application is coming from native resources that might be fetching Web Content Manager resources, then you must install a MobileFirst server to provide these resources.

Direct update service

If you plan to use the direct update service feature to update the embedded markup for changes, you require a MobileFirst server.

Native notifications

If your application uses native notifications, MobileFirst is required to generate the iOS and Android notification service.

Authentication services

If you use the MobileFirst authentication or access control service for single sign-on (SSO) between MobileFirst and WebSphere Portal Express, you must install a MobileFirst server. If you plan to use anonymous access or access all resources of the application through WebSphere Portal Express or IBM Web Content Manager, then you do not need a MobileFirst server.

Tracking usage

If your application uses MobileFirst to track usage, you must install a MobileFirst server. The server that you install must also support the load from clients who send usage data.

Device provisioning

If you are providing device provisioning, you must install a MobileFirst server to provide the certificate for the device and data.

Application Center EAR

The Application Center Ear is an optional application that provides an application store environment. If you are using it to manage applications on devices as an MDM solution, you need a MobileFirst server to run the MobileFirst EAR and the Application Center EAR.

Default component overview

When you integrate MobileFirst and WebSphere Portal Express, you can create a MobileFirst hybrid application that includes a WebSphere Portal Express web application. This hybrid application can run in a mobile browser and as a native mobile application.

Modules

- wp_worklight_ext
- `wp_worklight`
- wp_worklight_css

The `wp_worklight_ext` module is included in the default deferred profile and is active by default. This module automatically loads the MobileFirst Client and Cordova APIs for you so you can add native device capabilities in your hybrid applications. The APIs are JavaScript resources and are optimized for each device. For example, the iOS resources are loaded in a MobileFirst hybrid application on an iOS device. But they are not loaded in other circumstances such as in a web application, on an android device, or on a desktop.

Device classes

- `worklight`
- `ios`
- `android`
- `iemobile`
- `blackberry`
- `smartphone`
- `tablet`

With these device classes, you can target the appropriate environment to expose your native capabilities within WebSphere Portal Express. The default MobileFirst modules use these devices classes to determine which resources to load. If you are adding to the default capabilities, you can also use these device classes to optimally load your own device-specific resources.

Device class equations

- `android+smartphone`
- `worklight+(ios/android)`
- `(android/ios)+worklight+!tablet`

With device class equations, you can create a specific device class that includes or excludes some of the device classes. You can use parenthesis to group the device classes. To use the operation AND, use the plus sign, `+`. To use the operation OR use the forward slash, `/`. To use the operation NOT, use the exclamation point, `!`.

APIs & Samples

- Worklight Client
  - `WL.Client.getUserInfo`
  - `WL.Client.Push.subscribe`
  - And others
- Cordova
  - `navigator.camera.getPicture`
  - `navigator.geolocation.getCurrentPosition`
  - And others
- Authentication
  - `WL.Client.createChallengeHandler`
  - And others

The new MobileFirst resources allow access to native device capabilities by using the high-level JavaScript without having to know or use the native device programming language. The JavaScript APIs call the native device APIs for you. Sample apps are provided on the catalog for each of the APIs to show example usage syntax. You can copy, paste, and modify these samples to fit your needs.

To get started, create a MobileFirst hybrid application that points to and renders your WebSphere Portal Express URL. Modify your WebSphere Portal Express code to call the JavaScript APIs to access the native device capabilities. For example, your application could have a feature for taking and uploading a picture that is available on devices that have cameras and unavailable otherwise. Or a feature that is tailored to the user's geolocation on devices that have GPS and not otherwise. Or a feature that sends and receives push notifications on devices that support push notifications and not otherwise.

Related information:

Apache Cordova

MobileFirst API

Creating a MobileFirst hybrid application for your portal

You can create a hybrid application to add native device capabilities to your portal with IBM MobileFirst.

Before you begin

To test an iPhone and iPad hybrid application, you must use a Mac with Xcode installed. To test an Android hybrid application, you must have an Android Virtual Device created. To test a Windows Phone hybrid application, you must have Windows 8 and the Windows Phone SDK installed.

Procedure

1. Create your MobileFirst Hybrid Application in the MobileFirst Eclipse development environment by selecting **New** > **MobileFirst Project**.
2. In the **Name** field, name your project. In this example, name your project `MFPortal`. In the **Project Templates** field, select **Hybrid Application**, which is the default, and click **Next**.
3. In the **Application Name** field, name your application. In this example, name your application `MFPortalApp`. Check the JavaScript libraries that you want your application to use if any and click **Finish**. The project and application artifacts are created. You can see your *project_name*\apps*app_name* folder in the Project Explorer. Your application descriptor, `application-descriptor.xml`, is in the Application Descriptor Editor.
4. In `application-descriptor.xml`, you can change the basic settings of your application, such as your `application id`, `displayName`, `description`, and `author` details.
5. In `application-descriptor.xml`, change `mainFile` to your WebSphere Portal Express URL with the **`uri=wl:id`** parameter appended to the end. For example, enter `http://localhost:port/wps/portal?uri=wl:id:MFPortalApp`. Replace **localhost** with your host name and `MFPortalApp` with your application name.
6. In the `common/images` folder, replace the `icon.png` and `thumbnail.png` files with the custom images that you want for your application. The `thumbnailImage` shows that the icons used for your application are in the `common/images` folder.

7. Save your changes.
8. Create a MobileFirst environment to build the native part of the hybrid application. Right-click your *project_name*\apps*app_name* folder and then select **New** > **MobileFirst Environment**.
9. In the New MobileFirst Environment dialog, select any native environments that you want your application to support, such as iPhone, iPad, Windows Phone, and Android phones and tablets. Click **Finish**. Your Project Explorer window is updated with a native application in a folder named *project_name**app_name**platform*. In this example, the folder for Android is MFPortal\MFPortalApp\Android. For iOS, it is MFPortal\MFPortalApp\iphone or MFPortal\MFPortalApp\ipad for iOS. For Windows Phone, it is MFPortal\MFPortalApp\windowsphone8. For Windows Phone applications, set the URI entered for the **mainFile** value of the application-descriptor.xml as the **StartPageUri** value in MainPage.xaml. MobileFirst manages the lifecycle of these folders. When the web application you initially created in the project is built and deployed, the native applications are overwritten with any application changes.
10. To test your application, right-click the *project_name*\apps*app_name* folder and select **Run As** > **Build All Environments**.
    a. To test an Android hybrid application, you must have an Android Virtual Device created. Then, right-click on your native Android application folder, and select **Run As** > **Android Application**. The native Android application is a peer to your project. In this example, a project that is named MFPortalMFPortalAppAndroid is in your MobileFirst project.
    b. To test an iPhone and iPad hybrid application, you must use a Mac with Xcode installed. Right-click on your native iPhone or iPad application folder, which is *project_name*\apps*app_name*\iphoneor *project_name*\apps*app_name*\ipad, and select **Run As** > **Xcode project**. Then, in Xcode, select your emulator and run the application.
    c. To test a Windows Phone hybrid application, you must have Windows 8 with Windows Phone 8 SDK installed. Right-click your native Windows Phone application folder, which is project_name\apps\app_name\ windowsphone8, and select **Run As** > **Visual Studio Project**. Then, in Visual Studio, select your emulator and run the application.

Module framework for IBM MobileFirst

The module framework allows extensions to contribute to different areas of a page to provide flexibility, enhance the user experience, and maximize performance.

IBM WebSphere Portal Express provides the MobileFirst 7.0 resources by default through modules that are defined in the default profiles. These modules use prerequisites to ensure that all necessary resources are aggregated and device classes to ensure that they are only provided for a particular environment. For more information, see *The module framework*.

"Configuring the MobileFirst properties" on page 1009
Configure the MobileFirst properties to add your preferences to the portal page.

CF07 "Setting up single sign-on with MobileFirst 7.0" on page 1010
You can set up single sign-on with MobileFirst so users can share a session between a WebSphere Portal Express and MobileFirst server.

"Meta-Modules for IBM MobileFirst integration" on page 1022
MobileFirst provides a set of ready-to-use modules.

Related concepts:

"The module framework" on page 2526
The module framework allows extensions to contribute to different areas of a page to provide flexibility, enhance the user experience, and maximize performance.

Configuring the MobileFirst properties

Configure the MobileFirst properties to add your preferences to the portal page.

Before you begin

The wp_worklight module requires a wl_config module that is defined in *PortalServer_root*\theme\wp.theme.worklight.ext\installedApps\wp.theme.worklight.ext.ear\wp.theme.worklight.ext.war\WEB-INF\plugin.xml file.

Procedure

1. Use the wl_config module to load the MobileFirst configuration properties.

```
<extension point="com.ibm.portal.resourceaggregator.module" id="wl_portal_config" >
    <module id="wl_config">
     <runtimeActivation>
   <condition deviceClass="worklight"/>
  </runtimeActivation>

          <contribution type="config">
              <sub-contribution type="config_dynamic">
                  <uri value="wl:id" />
              </sub-contribution>
          </contribution>
      </module>
  </extension>
```

2. To activate the MobileFirst configuration properties, add the **"?uri=wl:id:appid"** parameter to the URL from the hybrid application, where the **appid** is replaced with the actual application ID in MobileFirst. After it is activated, the MobileFirst configuration properties are dynamically injected from the application HTML file to the portal page. For example:

```
<script>
  // Define WL namespace.
  var WL = WL ? WL : {};
  /**  * WLClient configuration variables.  * Values are injected by the deployer that packs t
  WL.StaticAppProps = {    "APP_DISPLAY_NAME": "DemoApp",
         "APP_SERVICES_URL": "\/apps\/services\/",
         "ENVIRONMENT": "iphone",
         "LOGIN_DISPLAY_TYPE": "embedded",
         "POSTFIX_APP_SERVICES_URL": "\/apps\/services\/",
         "POSTFIX_WORKLIGHT_ROOT_URL": "\/apps\/services\/api\/DemoApp\/iphone\/",
         "WORKLIGHT_ROOT_URL": "\/apps\/services\/api\/DemoApp\/iphone\/"
       };
</script>
```

3. Configure the MobileFirst configuration properties through JavaScript. For example, use this code:

```
WL.StaticAppProps.ENVIRONMENT
```

4. If your application needs configuration properties in addition to the base ones provided, you can easily mix them into this WL.StaticAppProps object with JavaScript. For example:

```
<script type="text/javascript">
i$.bindDomEvt(window, "onload", function(evt) {
 if (window.WL && window.WL.StaticAppProps) {
  window.WL.StaticAppProps.APP_VERSION = "1.0";
```

```
 window.WL.StaticAppProps.WORKLIGHT_PLATFORM_VERSION = "7.0.0";
 }
});
</script>
```

5. Inject the JavaScript code that you created onto your page from your theme. You can add the code that you just created to one of your existing dynamic content spot `.jsps` or create a new dynamic content spot. Or, you can add the JavaScript to an HTML file in an existing or new module.

Setting up single sign-on with MobileFirst 7.0

You can set up single sign-on with MobileFirst so users can share a session between a WebSphere Portal Express and MobileFirst server.

Before you begin

Both WebSphere Portal Express and MobileFirst servers must be configured to use the same user registry, LTPA keys, and be set with a specified domain for SSO. If you are using the WebSphere Application Server, see the WebSphere Application Server documentation.

For more information about system support requirements, see System requirements.

Procedure

1. Run the following **configEngine** tasks. These **configEngine** tasks are available only on the WebSphere Portal Express server. If the MobileFirst server is on WebSphere Application Server, run the following commands from the WebSphere Integrated Solutions Console. You can use steps B and C if MobileFirst is on the same instance as WebSphere Portal Express.
   a. **ConfigEngine.bat configure-single-signon -Ddomain=<domain name> -DWasRemoteHostName=<hostname> -DWasSoapPort=<port> -DWasPassword=<password> -Dinteroperable=true -DattributePropagation=true -DrequiresSSL=false**
   b. Optional: **ConfigEngine.bat export-ltpakeys-single-signon -DkeyFile=c:\ltpa.txt -DkeyPass=<testpass> -DdmgrFlag=false -DWasRemoteHostName=<hostname> -DWasSoapPort=<port> -DWasPassword=<password>**
   c. Optional: **ConfigEngine.bat import-ltpakeys-single-signon -DkeyFile=c:\ltpa_demo.txt -DkeyPass=<somepassword> -DdmgrFlag=false -DWasRemoteHostName=<hostname> -DWasSoapPort=<port> -DWasPassword=<password>**
2. To prepare the MobileFirst server, you must update the MobileFirst WAR to enable applications to authenticate with the user registry. Update `authenticationConfig.xml` in your MobileFirst project. IT is in *MobileFirst Project*`/server/conf/authenticationConfig.xml`. Find the `<securityTests>` element and add the mobile and web security tests.
   a. Find the `<securityTests>` element and add the mobile and web security tests from the following example.

```
<mobileSecurityTest name="mobileTests">
    <testDeviceId provisioningType="none" />
    <testUser realm="WASLTPARealm" />
</mobileSecurityTest>
<webSecurityTest name="WASLTPARealmTests">
    <testUser realm="WASLTPARealm"/>
</webSecurityTest>
```

b. Uncomment the security realm.

```
<!-- For websphere -->
<realm name="WASLTPARealm" loginModule="WASLTPAModule">
   <className>com.worklight.core.auth.ext.WebSphereFormBasedAuthenticator
       </className>
   <parameter name="login-page" value="/login.html"/>
   <parameter name="error-page" value="/loginError.html"/>
</realm>
```

c. Uncomment the login module for WebSphere.

```
<!-- For websphere -->
<loginModule name="WASLTPAModule">
    <className>com.worklight.core.auth.ext.WebSphereLoginModule</className>
</loginModule>
```

3. Modify the MobileFirst project WAR by adding two new HTML files to the WAR. The project WAR is in */MobileFirst Project*/bin. Copy this WAR to another location for editing.

a. Create an login.html file with the following contents.

```
<html>
    <head>
        <title>Login</title>
    </head>
    <body>
        <form method="post" action="j_security_check">
            <label for="j_username">User name:</label>
            <input type="text" id="j_username" name="j_username" />
            <br />
            <label for="j_password">Password:</label>
            <input type="password" id="j_password" name="j_password" />
            <br />
            <input type="submit" id="login" name="login" value="Log In" />
        </form>
    </body>
</html>
```

b. Create a loginError.html file with the following contents.

```
<html>
    <head></head>
    <body>
        Login Error
    </body>
</html>
```

c. Add the login.html and loginError.html files to the highest level directory of the WAR.

4. Modify the MobileFirst project WAR that was updated in the previous step by editing the web.xml file in the WEB-INF directory.

```
<login-config>
    <auth-method>FORM</auth-method>
    <form-login-config>
        <form-login-page>/login.html</form-login-page>
        <form-error-page>/loginError.html</form-error-page>
    </form-login-config>
</login-config>
```

5. Optional: Add a security constraint to protect the web resource by modifying web.xml in the updated MobileFirst project WAR

```
<security-constraint id="SecurityConstraint_1">
 <web-resource-collection id="WebResourceCollection_1">
  <web-resource-name>mobilefirst</web-resource-name>
  <description>Protecting mobilefirst application</description>
  <url-pattern>/*</url-pattern>
  <http-method>GET</http-method>
  <http-method>POST</http-method>
```

```
 </web-resource-collection>
 <auth-constraint id="AuthConstraint_1">
  <description>MobileFirst applications</description>
  <role-name>Administrator</role-name>
 </auth-constraint>
 <user-data-constraint id="UserDataConstraint_1">
  <transport-guarantee>NONE</transport-guarantee>
 </user-data-constraint>
</security-constraint>

<security-role id="SecurityRole_1">
 <description>Only specific users</description>
 <role-name>Administrator</role-name>
</security-role>
```

6. When the MobileFirst project WAR is updated, deploy it to the MobileFirst server.
7. Restart the server where MobileFirst is installed. If you added the security constraint, map the group or user to the EAR file.
8. Update the `application-descriptor.xml` file to add the security tests you configured. Open *MobileFirst Project*`/apps/`*mobilefirst app*`/application-descriptor.xml` in the design view and update it to have the correct realms and security tests.
9. In the main application, add a security test that is called **WASLTPARealmTests** in the Common (optional) section.
10. In **Android phones and tablets** > **Details**, add the Security test called **mobileTests**.
11. After you create the server-side WAR, change the client side to allow authentication between the two servers. In a demonstration application, modify the HTML of *MobileFirst Project*`/apps/mobilefirst app/common/index.html` to include a login form and JavaScript to handle the response. In a demonstration, `index.html` includes the code in the following example.

```
<!DOCTYPE HTML>
<html>
     <head>
      <meta charset="UTF-8">
      <title>index</title>
      <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, m
      <link rel="shortcut icon" href="images/favicon.png">
      <link rel="apple-touch-icon" href="images/apple-touch-icon.png">
      <link rel="stylesheet" href="css/main.css">
      <script>window.$ = window.jQuery = WLJQ;</script>
     </head>
     <body id="content" style="display: none">
      <div id="AppBody">
          <div class="wrapper">
          </div>
      </div>
      <div id="AuthBody" style="display: none">
          <div id="loginForm">
              Username:<br/>
              <input type="text" id="usernameInputField" autocorrect="off" autocapitalize="off"
              Password:<br/>
              <input type="password" id="passwordInputField" autocorrect="off" autocapitalize="o
              <input type="button" id="loginButton" value="Login" />
              <input type="button" id="cancelButton" value="Cancel" />
          </div>
      </div>
      <!--application UI goes here-->
      Hello Worklight
      <script src="js/initOptions.js"></script>
      <script src="js/main.js"></script>
```

```
      <script src="js/messages.js"></script>
      <script src="js/challengeResponse.js"></script>
     </body>
</html>
```

12. Update the initialization options for MobileFirst to force the application to connect to the MobileFirst server on start by adding `WL.Client.connect();` in *MobileFirst Project*/apps/*mobilefirst app*/common/js/main.js.
13. Add JavaScript to handle the response from the MobileFirst and WebSphere Portal Express servers. In this SSO demonstration application, create a file that is called challengeResponse.js, in *MobileFirst Project*/apps/*mobilefirst app*/common/js. Add the following example to the contents of the file. Update the line `goToPortalServer("http://server:port/wps/myportal");` to point to your WebSphere Portal Express server.

```
/*
*  Licensed Materials - Property of IBM
*  5725-G92 (C) Copyright IBM Corp. 2006, 2012. All Rights Reserved.
*  US Government Users Restricted Rights - Use, duplication or
*  disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
*/

var sampleAppRealmChallengeHandler = WL.Client.createChallengeHandler("WASLTPARealm");
var busyIndicator = new WL.BusyIndicator('content', {text: "Loading..."});

sampleAppRealmChallengeHandler.isCustomResponse = function(response) {
    if (!response || response.responseText === null) {
        return false;
    }
    var indicatorIdx = response.responseText.search('j_security_check');

    if (indicatorIdx >= 0){
  return true;
 }
 return false;
};

sampleAppRealmChallengeHandler.handleChallenge = function(response) {
 busyIndicator.show();
 $('#AppBody').hide();
 WL.EncryptedCache.open("wpsadmin", true, onReadOpen, onOpenError);
};

sampleAppRealmChallengeHandler.submitLoginFormCallback = function(response) {
    var isLoginFormResponse = sampleAppRealmChallengeHandler.isCustomResponse(response);
    if (isLoginFormResponse){
     sampleAppRealmChallengeHandler.handleChallenge(response);
    } else {
     $('#AppBody').show();
     $('#AuthBody').hide();
     sampleAppRealmChallengeHandler.submitSuccess();
     WL.Client.getCookies().then(function (cookies) {
      busyIndicator.hide();
      goToPortalServer("http://server:port/wps/myportal");
     });
    }
};


$('#loginButton').bind('click', function () {
 busyIndicator.show();
 WL.EncryptedCache.write("username", $('#usernameInputField').val(), onWriteSuccess, onWriteF
 function onWriteSuccess(status){
  WL.EncryptedCache.write("password", $('#passwordInputField').val(), onWriteSuccess2, onWrit
  function onWriteSuccess2(status2){
   WL.EncryptedCache.close(onCloseCompleteHandler, onCloseFailureHandler);
```

```
  }
 }
 function onWriteFailure(status){
  alert("Encrypted cache closed, writing failed");
 }
});


function onCloseCompleteHandler(status){
 var reqURL = '/j_security_check';
    var options = {};
    options.parameters = {
        j_username : $('#usernameInputField').val(),
        j_password : $('#passwordInputField').val()
    };
    options.headers = {};
    sampleAppRealmChallengeHandler.submitLoginForm(reqURL, options, sampleAppRealmChallengeHandl
}


function onCloseFailureHandler(status){
 alert("close faiure");
}


$('#cancelButton').bind('click', function () {
 sampleAppRealmChallengeHandler.submitFailure();
 $('#AppBody').show();
 $('#AuthBody').hide();
});

function onReadOpen(status){
 WL.EncryptedCache.read("username", onDecryptReadSuccess, onDecryptReadFailure);
 function onDecryptReadSuccess(value){
  WL.EncryptedCache.read("password", onDecryptReadSuccess2, onDecryptReadFailure);
  function onDecryptReadSuccess2(value2){
   if (value && value2){
    // submit 1 & 2
    var reqURL = '/j_security_check';
       var options = {};
       options.parameters = {
           j_username : value,
           j_password : value2
       };
       options.headers = {};
       sampleAppRealmChallengeHandler.submitLoginForm(reqURL, options, sampleAppRealmChallengeHa
   } else {
    // Didn't find any cached info, ask for login.
    busyIndicator.hide();
    $('#AuthBody').show();
    $('#passwordInputField').val('');
   }
  }
 }
 function onDecryptReadFailure(status){
  alert("Encrypted cache closed, reading failed");
 }
}

 function onOpenError(status){
 switch(status){
  case WL.EncryptedCache.ERROR_KEY_CREATION_IN_PROGRESS:
   alert("ERROR: KEY CREATION IN PROGRESS");
   break;
  case WL.EncryptedCache.ERROR_LOCAL_STORAGE_NOT_SUPPORTED:
   alert("ERROR: LOCAL STORAGE NOT SUPPORTED");
```

```
   break;
  case WL.EncryptedCache.ERROR_NO_EOC:
   alert("ERROR: NO EOC");
   break;
  case WL.EncryptedCache.ERROR_COULD_NOT_GENERATE_KEY:
   alert("ERROR: COULD NOT GENERATE KEY");
   break;
  case WL.EncryptedCache.ERROR_CREDENTIALS_MISMATCH:
   alert("ERROR: CREDENTIALS MISMATCH");
   break;
  default:
   alert("AN ERROR HAS OCCURED. STATUS :: " + status);
 }
}
```

yes

14. Create a Cordova Plugin that adds the token the the WebView's cookie store so WebSphere Portal Express can recieve the LtpaToken from an Android application.
    a. Create a Java file called `CookieInjector.java` in *`MobileFirst Project`*`/apps/`*`mobilefirst app`*`/android/native/src/com/`*`mobilefirst app`*. Add the following code to it. Replace *mobilefirst app* with your app name.

```
package com.mobilefirst_app;

import java.util.List;

import org.apache.cordova.CallbackContext;
import org.apache.cordova.CordovaArgs;
import org.apache.cordova.CordovaPlugin;
import org.apache.http.client.CookieStore;
import org.apache.http.cookie.Cookie;
import org.json.JSONException;

import android.webkit.CookieManager;

import com.worklight.common.Logger;
import com.worklight.wlclient.HttpClientManager;

public class CookieInjector extends CordovaPlugin {
 Logger l = Logger.getInstance(CookieInjector.class.getName());

 @Override
 public boolean execute(String action, CordovaArgs args, CallbackContext callbackContext)
  if ("INJECT-COOKIES-TO-WEBVIEW".equals(action)){
   l.debug("Started injecting cookies");

   CookieStore cookieStore = HttpClientManager.getInstance().getHttpClient().getCookieSto
   List<Cookie> list = cookieStore.getCookies();
   for (Cookie cookie : list){
    String cookieName = cookie.getName();
    l.debug("Found cookie :: " + cookieName);
    if ("LtpaToken".equals(cookieName)){
     l.debug("Found LtpaToken cookie");
     CookieManager cookieManager = CookieManager.getInstance();
     String cookieValue = cookie.getName() + "=" + cookie.getValue();
     cookieManager.setCookie(cookie.getDomain(), cookieValue);
     break;
    }
    l.debug("LtpaToken cookie not found");
   }
   l.debug("Done injecting cookies");
   callbackContext.success();
   return true;
```

```
        }
        return false;
      }
    }
```

b. Add the following code to *MobileFirst Project*/apps/*mobilefirst app*/android/native/res/xml/config.xml before the closing </widget> tag. Replace *mobilefirst app* with your app name.

```
<feature name="CookieInjector">
    <param name="android-package" value="com.mobilefirst_app.CookieInjector" />
</feature>
```

15. Add the following function to *MobileFirst Project*/apps/*mobilefirst app*/android/apps/mobilefirst app/android/js/main.js.

```
function goToPortalServer(url) {
 cordova.exec(function () {
  location.href = url;
 }, function () {
  alert('failure injecting cookies');
 }, "CookieInjector", "INJECT-COOKIES-TO-WEBVIEW",[]);
}
```

16. Build your MobileFirst application for your MobileFirst server by right-clicking the MobileFirst application and selecting **Run As** > **Build Settings and Deploy Target**.
17. Select **Build the application to work with a different MobileFirst server**.
18. Add the information for your MobileFirst server to the **Server** and **Context path** fields.
19. Build your application by right-clicking the MobileFirst application and selecting **Run As** > **Build All Environments**.
20. Install the MobileFirst application to your MobileFirst server. Open the MobileFirst console at http://server:port/worklightconsole and upload the MobileFirst application by adding it to the **Deploy application or adapter** field. Your MobileFirst application file can be found in your Eclipse workspace in the bin folder. The MobileFirst application file has the .wlapp extension.

CF07 "Setting up single sign-on with MobileFirst"
You can set up single sign-on with MobileFirst so users can share a session between a WebSphere Portal Express and MobileFirst server.

Setting up single sign-on with MobileFirst:

You can set up single sign-on with MobileFirst so users can share a session between a WebSphere Portal Express and MobileFirst server.

Before you begin

SSO is no longer supported on Android with Worklight 6.2.

Both WebSphere Portal Express and MobileFirst servers must be configured to use the same user registry, LTPA keys, and be set with a specified domain for SSO. For more information, see *Configuring Portal to use a user registry* and *Managing your user registry*. Or, if you are using the WebSphere Application Server, see the WebSphere Application Server documentation.

For more information about system support requirements, see System requirements.

Procedure

1. Run the following **configEngine** tasks. These **configEngine** tasks are available only on the WebSphere Portal Express server. If the MobileFirst server is on WebSphere Application Server, run the following commands from the WebSphere Integrated Solutions Console. You can use steps B and C if MobileFirst is on the same instance as WebSphere Portal Express.
   a. **ConfigEngine.bat configure-single-signon -Ddomain=<domain name> -DWasRemoteHostName=<hostname> -DWasSoapPort=<port> -DWasPassword=<password> -Dinteroperable=true -DattributePropagation=true -DrequiresSSL=false**
   b. Optional: **ConfigEngine.bat export-ltpakeys-single-signon -DkeyFile=c:\ltpa.txt -DkeyPass=<testpass> -DdmgrFlag=false -DWasRemoteHostName=<hostname> -DWasSoapPort=<port> -DWasPassword=<password>**
   c. Optional: **ConfigEngine.bat import-ltpakeys-single-signon -DkeyFile=c:\ltpa_demo.txt -DkeyPass=<somepassword> -DdmgrFlag=false -DWasRemoteHostName=<hostname> -DWasSoapPort=<port> -DWasPassword=<password>**
2. To prepare the MobileFirst server, you must update the MobileFirst WAR to enable applications to authenticate with the user registry. Update `authenticationConfig.xml` in your MobileFirst project. IT is in *MobileFirst Project*`/server/conf/authenticationConfig.xml`. Find the `<securityTests>` element and add the mobile and web security tests.
   a. Find the `<securityTests>` element and add the mobile and web security tests from the following example.

```
<mobileSecurityTest name="mobileTests">
    <testDeviceId provisioningType="none" />
    <testUser realm="WASLTPARealm" />
</mobileSecurityTest>
<webSecurityTest name="WASLTPARealmTests">
    <testUser realm="WASLTPARealm"/>
</webSecurityTest>
```

   b. Uncomment the security realm.

```
<!-- For websphere -->
<realm name="WASLTPARealm" loginModule="WASLTPAModule">
   <className>com.worklight.core.auth.ext.WebSphereFormBasedAuthenticator
       </className>
   <parameter name="login-page" value="/login.html"/>
   <parameter name="error-page" value="/loginError.html"/>
</realm>
```

   c. Uncomment the login module for WebSphere.

```
<!-- For websphere -->
<loginModule name="WASLTPAModule">
    <className>com.worklight.core.auth.ext.WebSphereLoginModule</className>
</loginModule>
```

3. Modify the MobileFirst project WAR by adding two new HTML files to the WAR. The project WAR is in */MobileFirst Project*`/bin`. Copy this WAR to another location for editing.
   a. Create an `login.html` file with the following contents.

```
<html>
    <head>
        <title>Login</title>
    </head>
    <body>
        <form method="post" action="j_security_check">
            <label for="j_username">User name:</label>
```

```
                <input type="text" id="j_username" name="j_username" />
                <br />
                <label for="j_password">Password:</label>
                <input type="password" id="j_password" name="j_password" />
                <br />
                <input type="submit" id="login" name="login" value="Log In" />
            </form>
        </body>
    </html>
```

b. Create a `loginError.html` file with the following contents.

```
<html>
    <head></head>
    <body>
        Login Error
    </body>
</html>
```

c. Add the `login.html` and `loginError.html` files to the highest level directory of the WAR.

4. Modify the MobileFirst project WAR that was updated in the previous step by editing the `web.xml` file in the `WEB-INF` directory.

```
<login-config>
    <auth-method>FORM</auth-method>
    <form-login-config>
        <form-login-page>/login.html</form-login-page>
        <form-error-page>/loginError.html</form-error-page>
    </form-login-config>
</login-config>
```

5. Optional: Add a security constraint to protect the web resource by modifying `web.xml` in the updated MobileFirst project WAR

```
<security-constraint id="SecurityConstraint_1">
 <web-resource-collection id="WebResourceCollection_1">
  <web-resource-name>mobilefirst</web-resource-name>
  <description>Protecting mobilefirst application</description>
  <url-pattern>/*</url-pattern>
  <http-method>GET</http-method>
  <http-method>POST</http-method>
 </web-resource-collection>
 <auth-constraint id="AuthConstraint_1">
  <description>MobileFirst applications</description>
  <role-name>Administrator</role-name>
 </auth-constraint>
 <user-data-constraint id="UserDataConstraint_1">
  <transport-guarantee>NONE</transport-guarantee>
 </user-data-constraint>
</security-constraint>

<security-role id="SecurityRole_1">
 <description>Only specific users</description>
 <role-name>Administrator</role-name>
</security-role>
```

6. When the MobileFirst project WAR is updated, deploy it to the MobileFirst server.
7. Restart the server where MobileFirst is installed. If you added the security constraint, map the group or user to the EAR file.
8. Update the `application-descriptor.xml` file to add the security tests you configured. Open *MobileFirst Project*`/apps/`*mobilefirst app*`/application-descriptor.xml` in the design view and update it to have the correct realms and security tests.
9. In the main application, add a security test that is called **WASLTPARealmTests** in the Common (optional) section.

10. In **Android phones and tablets** > **Details**, add the Security test called **mobileTests**.
11. Open *MobileFirst Project*/apps/*mobilefirst app*/application-descriptor.xml and add the <securityTests> element.

```
<securityTests>
    <mobileSecurityTest name="mobileTests">
        <testDeviceId provisioningType="none" />
        <testUser realm="WASLTPARealm" />
    </mobileSecurityTest>
    <customSecurityTest name="WASLTPARealmTests">
        <test realm="WASLTPARealm" isInternalUserID="true"/>
    </customSecurityTest>
</securityTests>
```

12. Add the example <realm> for WASLTPARealm.

```
<realm loginModule="WASLTPAModule" name="WASLTPARealm">
      <className>com.worklight.core.auth.ext.FormBasedAuthenticator</className>
  </realm>
 </realms>
```

13. Add the example <loginModule> for WASLTPAModule.

```
<loginModule name="WASLTPAModule">
 <className>com.worklight.core.auth.ext.NonValidatingLoginModule</className>
</loginModule>
```

14. After you create the server-side WAR, change the client side to allow authentication between the two servers. In an SSO demonstration application, modify the HTML of *MobileFirst Project*/apps/mobilefirst app/common/index.html to include a login form and JavaScript to handle the response. In a demonstration, index.html includes the code in the following example.

```
<!DOCTYPE HTML>
<html>
     <head>
      <meta charset="UTF-8">
      <title>index</title>
      <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0
      <link rel="shortcut icon" href="images/favicon.png">
      <link rel="apple-touch-icon" href="images/apple-touch-icon.png">
      <link rel="stylesheet" href="css/main.css">
      <script>window.$ = window.jQuery = WLJQ;</script>
     </head>
     <body id="content" style="display: none">
      <div id="AppBody">
          <div class="wrapper">
          </div>
      </div>
      <div id="AuthBody" style="display: none">
          <div id="loginForm">
              Username:<br/>
              <input type="text" id="usernameInputField" autocorrect="off" autocapitalize="of
              Password:<br/>
              <input type="password" id="passwordInputField" autocorrect="off" autocapitalize
              <input type="button" id="loginButton" value="Login" />
              <input type="button" id="cancelButton" value="Cancel" />
          </div>
      </div>
      <!--application UI goes here-->
      Hello Worklight
      <script src="js/initOptions.js"></script>
      <script src="js/main.js"></script>
      <script src="js/messages.js"></script>
      <script src="js/challengeResponse.js"></script>
     </body>
</html>
```

15. Update the initialization options for MobileFirst to force the application to connect to the MobileFirst server on start by changing `connectOnStartup` from **`false`** to **`true`** in *`MobileFirst Project`*`/apps/`*`mobilefirst app`*`/common/js/initOptions.js`.
16. Add JavaScript to handle the response from the MobileFirst and WebSphere Portal Express servers. In this SSO demonstration application, create a file that is called `challengeResponse.js`, in *`MobileFirst Project`*`/apps/`*`mobilefirst app`*`/common/js`. Add the following example to the contents of the file. Update the line <code>location.href= "http://server:port/wps/myportal"</code> to point to your WebSphere Portal Express server.

```
/*
*  Licensed Materials - Property of IBM
*  5725-G92 (C) Copyright IBM Corp. 2006, 2012. All Rights Reserved.
*  US Government Users Restricted Rights - Use, duplication or
*  disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
*/

var sampleAppRealmChallengeHandler = WL.Client.createChallengeHandler("WASLTPARealm");
var busyIndicator = new WL.BusyIndicator('content', {text: "Loading..."});

sampleAppRealmChallengeHandler.isCustomResponse = function(response) {
    if (!response || response.responseText === null) {
        return false;
    }
    var indicatorIdx = response.responseText.search('j_security_check');

    if (indicatorIdx >= 0){
  return true;
 }
 return false;
};

sampleAppRealmChallengeHandler.handleChallenge = function(response) {
 busyIndicator.show();
 $('#AppBody').hide();
 WL.EncryptedCache.open("wpsadmin", true, onReadOpen, onOpenError);
};

sampleAppRealmChallengeHandler.submitLoginFormCallback = function(response) {
    var isLoginFormResponse = sampleAppRealmChallengeHandler.isCustomResponse(response);
    if (isLoginFormResponse){
     sampleAppRealmChallengeHandler.handleChallenge(response);
    } else {
  $('#AppBody').show();
  $('#AuthBody').hide();
  sampleAppRealmChallengeHandler.submitSuccess();
  setTimeout(function(){
      busyIndicator.hide();
   location.href= "http://server:port/wps/myportal";
  }, 1000);
    }
};

$('#loginButton').bind('click', function () {
 busyIndicator.show();
 WL.EncryptedCache.write("username", $('#usernameInputField').val(), onWriteSuccess, onWriteFail
 function onWriteSuccess(status){
  WL.EncryptedCache.write("password", $('#passwordInputField').val(), onWriteSuccess2, onWriteFa
  function onWriteSuccess2(status2){
   WL.EncryptedCache.close(onCloseCompleteHandler, onCloseFailureHandler);

  }
 }
 function onWriteFailure(status){
  alert("Encrypted cache closed, writing failed");
```

```
 }
});


function onCloseCompleteHandler(status){
 var reqURL = '/j_security_check';
    var options = {};
    options.parameters = {
        j_username : $('#usernameInputField').val(),
        j_password : $('#passwordInputField').val()
    };
    options.headers = {};
    sampleAppRealmChallengeHandler.submitLoginForm(reqURL, options, sampleAppRealmChallengeHa
}


function onCloseFailureHandler(status){
 alert("close faiure");
}


$('#cancelButton').bind('click', function () {
 sampleAppRealmChallengeHandler.submitFailure();
 $('#AppBody').show();
 $('#AuthBody').hide();
});

function onReadOpen(status){
 WL.EncryptedCache.read("username", onDecryptReadSuccess, onDecryptReadFailure);
 function onDecryptReadSuccess(value){
  WL.EncryptedCache.read("password", onDecryptReadSuccess2, onDecryptReadFailure);
  function onDecryptReadSuccess2(value2){
   if (value && value2){
    // submit 1 & 2
    var reqURL = '/j_security_check';
       var options = {};
       options.parameters = {
           j_username : value,
           j_password : value2
       };
       options.headers = {};
       sampleAppRealmChallengeHandler.submitLoginForm(reqURL, options, sampleAppRealmChalleng
   } else {
    // Didn't find any cached info, ask for login.
    busyIndicator.hide();
    $('#AuthBody').show();
    $('#passwordInputField').val('');
   }
  }
 }
 function onDecryptReadFailure(status){
  alert("Encrypted cache closed, reading failed");
 }
}

 function onOpenError(status){
 switch(status){
  case WL.EncryptedCache.ERROR_KEY_CREATION_IN_PROGRESS:
   alert("ERROR: KEY CREATION IN PROGRESS");
   break;
  case WL.EncryptedCache.ERROR_LOCAL_STORAGE_NOT_SUPPORTED:
   alert("ERROR: LOCAL STORAGE NOT SUPPORTED");
   break;
  case WL.EncryptedCache.ERROR_NO_EOC:
   alert("ERROR: NO EOC");
   break;
  case WL.EncryptedCache.ERROR_COULD_NOT_GENERATE_KEY:
```

```
    alert("ERROR: COULD NOT GENERATE KEY");
    break;
   case WL.EncryptedCache.ERROR_CREDENTIALS_MISMATCH:
    alert("ERROR: CREDENTIALS MISMATCH");
    break;
   default:
    alert("AN ERROR HAS OCCURED. STATUS :: " + status);
  }
 }
```

yes

17. Build your MobileFirst application for your MobileFirst server by right-clicking the MobileFirst application and selecting **Run As** > **Build Settings and Deploy Target**.
18. Select **Build the application to work with a different MobileFirst server**.
19. Add the information for your MobileFirst server to the **Server** and **Context path** fields.
20. Build your application by right-clicking the MobileFirst application and selecting **Run As** > **Build All Environments**.
21. Install the MobileFirst application to your MobileFirst server. Open the MobileFirst console at `http://server:port/worklight/console` and upload the MobileFirst application by adding it to the **Deploy application or adapter** field. Your MobileFirst application file can be found in your Eclipse workspace in the `bin` folder. The MobileFirst application file has the `.wlapp` extension.

Meta-Modules for IBM MobileFirst integration

MobileFirst provides a set of ready-to-use modules.

These MobileFirst integration modules are not associated with a specific release. You can define which version of MobileFirst or Worklight to use.

The Meta-Module definitions are stored in the following files: CF07

- `mobilefirst70.json`
- `dav:fs-type1/themes/Portal8.5/contributions/mobilefirst70.json`
- If you want to use a previous version of MobileFirst, other versions are available at `PortalServer/theme/wp.theme.worklight.ext/installedApps/wp.theme.worklight.ext.ear/wp.theme.worklight.ext.war/`*`worklight version`*`/`

You can have only one file in the folder at one time because the contribution files are not supported at the same time. Specific MobileFirst Integration modules are listed in the MobileFirst 6.1 Integration sections.

MobileFirst Meta-Modules

- wp_worklight_ext
- wp_worklight
- wp_worklight_css
- wp_worklight_css_android
- wp_worklight_css_ios
- wp_worklight_jsonstore

CF07

MobileFirst 7.0 Integration

These modules initialize the MobileFirst Client and Cordova API to enable support for native device capabilities and other functions of the MobileFirst server. The modules are only aggregated when accessed through the MobileFirst hybrid shell.

The `plugin.xml` file location is *PortalServer_root*`/theme/wp.theme.worklight.ext/installableApps/wp.theme.worklight.ext.ear/wp.theme.worklight.ext.war/WEB-INF/plugin.xml`

For more information about these modules, their prerequisites or runtime activation, use the Theme Analyzer Portlet.

Table 133. List of MobileFirst 7.0 integration modules

Module	Description
mf_overrides_70	Provides overrides to the MobileFirst Client API to allow integration with WebSphere Portal Express
mf_android_70	Provides MobileFirst Client and Cordova JavaScript resources for Android devices
mf_ios_70	Provides MobileFirst Client and Cordova JavaScript resources for iOS devices
mf_winphone_70	Provides MobileFirst Client and Cordova JavaScript resources for Windows Phone devices.
mf_plugins_reg_android_70	Provides the Cordova plug-in definition list
mf_plugins_android_70	Provides the Cordova plug-ins JavaScript resources that enable native feature access for Android devices
mf_plugins_ios_70	Provides the Cordova plug-ins JavaScript resources that enable native feature access for iOS devices
mf_plugins_windphone_70	Provides the Cordova plug-ins JavaScript resources that enable native feature access for Windows Phone devices.
mf_client_css_android_70	Provides MobileFirst client CSS for Android devices, specifically for the diagnostic window, and modal dialog
mf_client_css_ios_70	Provides MobileFirst client CSS for iOS devices, specifically for the diagnostic window, and modal dialog
mf_client_css_winphone_70	Provides MobileFirst client CSS for Windows Phone devices, specifically for the diagnostic window, and modal dialog.
mf_client_jsonstore_android_70	Provides the JSON store feature for Android devices
mf_client_jsonstore_ios_70	Provides the JSON store feature for iOS devices
mf_client_jsonstore_winphone_70	Provides the JSON store feature for Windows Phone devices.

CF03

MobileFirst 6.2 Integration

These modules initialize the MobileFirst Client and Cordova API to enable support for native device capabilities and other functions of the MobileFirst server. The modules are only aggregated when accessed through the MobileFirst hybrid shell.

The `plugin.xml` file location is *PortalServer_root*`/theme/wp.theme.worklight.ext/installableApps/wp.theme.worklight.ext.ear/wp.theme.worklight.ext.war/WEB-INF/plugin.xml`

For more information about these modules, their prerequisites or runtime activation, use the Theme Analyzer Portlet.

Table 134. List of MobileFirst 6.2 integration modules

Module	Description
wl_overrides_62	Provides overrides to the MobileFirst Client API to allow integration with WebSphere Portal Express
wl_android_62	Provides MobileFirst Client and Cordova JavaScript resources for Android devices
wl_ios_62	Provides MobileFirst Client and Cordova JavaScript resources for iOS devices
wl_winphone_62	Provides MobileFirst Client and Cordova JavaScript resources for Windows Phone devices.
wl_plugins_reg_android_62	Provides the Cordova plug-in definition list
wl_plugins_android_62	Provides the Cordova plug-ins JavaScript resources that enable native feature access for Android devices
wl_plugins_ios_62	Provides the Cordova plug-ins JavaScript resources that enable native feature access for iOS devices
wl_plugins_windphone_62	Provides the Cordova plug-ins JavaScript resources that enable native feature access for Windows Phone devices.
wl_client_css_android_62	Provides MobileFirst client CSS for Android devices, specifically for the diagnostic window, and modal dialog
wl_client_css_ios_62	Provides MobileFirst client CSS for iOS devices, specifically for the diagnostic window, and modal dialog
wl_client_css_winphone_62	Provides MobileFirst client CSS for Windows Phone devices, specifically for the diagnostic window, and modal dialog.
wl_client_jsonstore_android_62	Provides the JSON store feature for Android devices
wl_client_jsonstore_ios_62	Provides the JSON store feature for iOS devices
wl_client_jsonstore_winphone_62	Provides the JSON store feature for Windows Phone devices.

MobileFirst 6.1 Integration

These modules initialize the MobileFirst Client and Cordova API to enable support for native device capabilities and other functions of the MobileFirst server. The modules are only aggregated when accessed through the MobileFirst hybrid shell.

The `plugin.xml` file location is *PortalServer_root*`/theme/wp.theme.worklight.ext/installableApps/wp.theme.worklight.ext.ear/wp.theme.worklight.ext.war/WEB-INF/plugin.xml`

For more information about these modules, their prerequisites or runtime activation, use the Theme Analyzer Portlet.

Table 135. List of MobileFirst 6.1 integration modules

Module	Description
wl_overrides_61	Provides overrides to the MobileFirst Client API to allow integration with WebSphere Portal Express
wl_android_61	Provides MobileFirst Client and Cordova JavaScript resources for Android devices
wl_ios_61	Provides MobileFirst Client and Cordova JavaScript resources for iOS devices
wl_plugins_reg_android_61	Provides the Cordova plug-in definition list
wl_plugins_android_61	Provides the Cordova plug-ins JavaScript resources that enable native feature access for Android devices
wl_plugins_ios_61	Provides the Cordova plug-ins JavaScript resources that enable native feature access for iOS devices
wl_client_css_android_61	Provides MobileFirst client CSS for Android devices, specifically for the diagnostic window, and modal dialog
wl_client_css_ios_61	Provides MobileFirst client CSS for iOS devices, specifically for the diagnostic window, and modal dialog
wl_cordova_css_61	Provides Cordova client CSS, specifically for the tab bar component
wl_client_jsonstore_android_61	Provides the JSON store feature for Android devices
wl_client_jsonstore_ios_61	Provides the JSON store feature for iOS devices

"Enabling JSON store module"
JSONStore features add the ability to store JSON documents in IBM MobileFirst applications.

Enabling JSON store module:

JSONStore features add the ability to store JSON documents in IBM MobileFirst applications.

About this task

JSONStore is a lightweight, document-oriented storage system that is included as a feature of MobileFirst, and enables persistent storage of JSON documents by using JavaScript API. Documents in an application are available in JSONStore even when the device that runs the application is offline. This persistent, always-available storage gives customers, employees, or users access to documents when for example there is no network connection to the device.

Procedure

1. Create page.
2. Place your custom portlet with the JSON store feature on the page that you created.
3. Edit the page profile and add the module `wp_mobilefirst_jsonstore` to the profile and save it.

Target MobileFirst resources

You can change the way that a web page looks for any device with responsive web design. Device classes are generic groupings of form factors so client devices can view web pages for every form factor without designing the page for each device.

When a client communicates with the server, it can be profiled into one or more of the generic device classes. Then, the client can be sent only the resources it needs for rendering its form factor. If you are using device classes, often you can also use device class logic equations, which are Boolean equation of device classes. Device class logic equations help narrow down the exact situation you want resources to be loaded.

With the default theme, responsive design and device classes and logic minimize resources that are downloaded by devices. For example, when you use navigation across devices, the theme modules provide a different navigation JSP for rendering and separate style sheets for different device types. The desktop gets the normal default experience. When a tablet or smartphone is detected, it switches over to the mobile navigation JSP page. With a `tablet` device class, the mobile navigation JSP turns into a side navigation. If any smartphone is detected and given the `smartphone` device class, the mobile navigation is rendered in a list. The device classes also change the containers to allow the content to fit on the smaller screen. Some features, such as edit mode, can be disabled.

The main use of device classes is by modules, which can use them to specify when a subcontribution resource is used. Dynamic content spots can also target a device class with a multiview choice (MVC) URL. For more information, see *mvc:URI scheme*. The device class is available as an attribute on the Composite Capabilities/Preference Profiles and can be retrieved within a JSP. You can also use device equations, which are device classes with Boolean logic for further control of what devices receive which set of resources. For more information, see *Device class equations*. There is a global variable available on the configuration object that can help targeting code in the JavaScript editor. For more information, see *Additional information about device classes for developers*.

There are some cases where you cannot use device classes with the device class logic equations. They cannot be used to determine layout templates, page filtering with the `supported-deviceclass` tag, and personalization. Personalization cannot

use device equation logic directly as defined by the Boolean logic. But it can create its own targeting rules of Boolean logic with the visibility rules editor. For more information, see *Device classes*.

A global JavaScript variable is provided for further customizing the user experience. The global variable `com_ibm_device_class` provides an array list of all currently set devices classes. If none are set, it is an empty array. For example, if you have JavaScript specific for processing smartphones, you can include the following code.

```
if (com_ibm_device_class.indexOf("smartphone") !== -1) {
//process smartphone
}
```

If you wanted to be more specific and target an iOS device that runs in a MobileFirst container, you can include the following code.

```
if (com_ibm_device_class.indexOf("ios") !== -1 && com_ibm_device_class.indexOf("worklight") !== -1
//process ios worklight
}
```

Related concepts:

"mvc:URI scheme" on page 2823
The `mvc:URI` scheme is a special URI format that accesses different resources, depending on the device class. This scheme is used by the Portal 8001 theme in the definition of several dynamic content spots.

"Device class equations" on page 2825
Device class equations are expressions that involve a mixture of device class operands and boolean logic operators.

"Device classes" on page 2821
Device classes are used in IBM WebSphere Portal Express as an abstraction for common properties for the device of a client. For instance, tablet computers can be grouped into a device class `tablets`, since they share a form factor and possibly other traits such as touch interface, or additional hardware sensors.

"Additional information about device classes for developers" on page 2822
The `DeviceClass` profile attribute contains only the highest priority device class on the client. Highest priority is determined as the first device class listed for the client. `DeviceClassList` provides access to all device classes on a client, as a string of comma-separated values.

Upgrading MobileFirst

You can create an EAR file, and copy MobileFirst resources into that EAR to keep up to date with the newest MobileFirst release.

About this task

The existing MobileFirst extension EAR is installed under the `\PortalServer\theme` location. You cannot modify this location directly since any changes can be overridden by a fix pack. You must create your own custom MobileFirst extension EAR and install it in `\wp_profile`. When you create your own EAR, start with a copy of the existing EAR and modify it. During this process, you replace occurrences of `wp` and `wps` with `custom` and occurrences of the current MobileFirst version number with your MobileFirst version number.

Procedure

1. Export the `worklight_extension.ear` from the WebSphere Integrated Solutions Console.

a. Click **Applications** > **Application Types** > **WebSphere enterprise applications**.
b. Click **Next** until you find the **worklight_extension** application.
c. Select **worklight_extension** and click **Export** on the toolbar.
d. Click the `worklight_extension.ear` link to download and save the EAR file to your file system.

2. Import the `worklight_extension.ear` into Eclipse or Rational Application Developer with Java EE Developer tools plug-in.
   a. Click **File** > **Import**.
   b. Select **Java EE** > **EAR file**. Then, click **Next**.
   c. Click **Browse**. Find and select `worklight_extension.ear` you exported to your file system.
   d. Rename the EAR project from **worklight_extension** to `custom_mobilefirst_extension` and then click **Next** twice.
   e. In the **Project Name** field, change the name from **wp.theme.worklight.ext** to `custom.theme.mobilefirst.ext` and then click **Finish**.
3. Customize the ear in Eclipse or Rational Application Developer for the new version of MobileFirst.
   a. In the custom_mobilefirst_extension EAR project, delete the `EarContent\META-INF\ibmconfig` folder and its contents.
   b. Right-click on the custom_mobilefirst_extension EAR project and select **Properties**. Select **Deployment Assembly**.
   c. Select the **wp.theme.worklight.ext.war Deploy Path** and rename it to `custom.theme.mobilefirst.ext.war`.
   d. Click **Apply** and **OK**.
   e. In the custom_mobilefirst_extension EAR project, delete the `EarContent\wp.theme.worklight.ext.war` file.
4. In the `custom_mobilefirst_extension` EAR project, modify the `EarContent\META-INF\application.xml` file. Change the display-name from MobileFirst Extension to `Custom Mobilefirst Extension`.
   a. Change the module ID from `wp.theme.worklight.ext` to `custom.theme.mobilefirst.ext`.
   b. Change the `web-uri` from `wp.theme.worklight.ext.war` to `custom.theme.mobilefirst.ext.war`.
   c. Change the `context-root` from `/wps/worklightExt` to `/custom/mobilefirstExt`.
5. In the `custom.theme.mobilefirst.ext` war project, modify the `WebContent\WEB-INF\web.xml` file.
   a. Change the web-app ID from wp_theme_worklight_ext_webapp_1 to `custom_theme_mobilefirst_ext_webapp_1`.
   b. Change the display-name from `Worklight_Extensions` to `Custom MobileFirst Extensions`.
6. In the `custom.theme.mobilefirst.ext` war project, in WebContent, create your new version folder and its contents. First, examine the existing version folder and its contents because that shows the folder structure that you create.
   a. Right-click on **WebContent** and select **New** > **Folder**.
   b. Enter the MobileFirst version number that you are working with as the folder name. For example, use v*your_new_mobilefirst_version*, where *your_new_mobilefirst_version* is your current version number.

c. Right-click on the folder you created and create child folders that are called `android` and `ios`.

d. Locate the Android-specific folders in your MobileFirst project. In previous versions, they were in `\YourMFProject\apps\YourApp\android\native\assets\www\default\js` and `\YourMFProject\apps\YourApp\android\native\assets\www\default\worklight` folders in your Eclipse MobileFirst project. Copy and paste them into the `v*your_new_mobilefirst_version*\android` folder.

e. Locate the iOS specific folders in your MobileFirst project. In previous versions, they were in `\YourMFProject\apps\YourApp\iphone\native\assets\www\default\js` and `\YourMFProject\apps\YourApp\iphone\native\assets\www\default\worklight` folders in your Eclipse MobileFirst project. Copy and paste them into the `v*your_new_mobilefirst_version*\ios` folder.

f. Locate the Windows Phone specific folders in your Worklight project. In previous versions, they were in `\YourMFProject\apps\YourApp\windowsphone8\native\assets\www\default\js` and `\YourMFProject\apps\YourApp\windowsphone8\native\assets\www\default\worklight` folders in your Eclipse MobileFirst project. Copy and paste them into the `v*your_new_mobilefirst_version*\winphone` folder.

g. Locate the `init.js` file in `WebContent\v*previousversion*\android\js` folder and copy it to the `WebContent\v*new_version*\android\js` folder.

h. Locate the `init.js` file in `WebContent\v*previousversion*\ios\js` folder and copy it to the `WebContent\v*new_version*\ios\js` folder.

i. Locate the `init.js` file in `WebContent\v*previousversion*\winphone\js` folder and copy it to the `WebContent\v*new_version*\winphone\js` folder.

j. Locate the `mobilefirst*previous_version*.json` file in the current version folder, where *previous_version* is the current MobileFirst version number. Copy it and paste it in your new version folder. Rename the file `mobilefirst*new_version*.json`, where *new_version* is your new MobileFirst version number.

k. Edit the newly copied file, find, and replace all occurrences of the current MobileFirst version number with your MobileFirst version number and save the file. This file defines the version-independent meta modules that `prereq` the version-dependent modules that are defined in the `plugin.xml` file in the next step.

7. Determine whether overrides for Cordova and MobileFirst APIs are required. WebSphere Portal Express defines overrides for Cordova and MobileFirst APIs to improve performance and integrate with WebSphere Portal Express.

   a. To prevent an error when MobileFirst resources load, the `WL.Utils.loadWLClientMessages` and `WL.Utils.setLocalization` functions are overridden. If these are still needed, locate and copy the portal directory from each of the Android and iOS directories in `WebContent\v*previous_version*\` and copy them to their respective directories in `WebContent\v*new_version*\`. Verify that the content of these functions accurately reflects the content from your new version of MobileFirst.

   b. If the Cordova APIs did not change, either copy the changes in the `WebContent\v*previousversion*\android\worklight\cordova.js` file and `WebContent\v*previousversion*\ios\worklight\cordova.js` to the new version of the MobileFirst folder. This method reduces the number of requests to WebSphere Portal Express. Or, Leave the Cordova API, and add a `<script>` element to all the pages that require MobileFirst resources. The `<script>` element must reference the `cordova.js` file directory. To

improve performance for MobileFirst applications in WebSphere Portal Express, two Cordova APIs were changed. Originally, the Cordova APIs attempt to load all of the Cordova plug-ins individually by dynamically determining their location. However, it is not possible with WebSphere Portal Express resource aggregation. The `injectScript` and `findCordovaPath` functions are rewritten to use the `ibmCfg.portalConfig.worklightResourcesPath[version_number]` JavaScript variable to determine the location of the Cordova resources. This value of this variable comes from the Resource Environment Provider property that you create in a following step. The `version_number` used in the variable must be updated to reflect the new version number, which is reflected in the following changes to the resource environment provider property.

c. Copy the changes that are in `WebContent\vprevious_version\android\worklight\cordova.js` and `WebContent\vprevious_version\ios\worklight\cordova.js` and copy them into the new version of MobileFirst that you are upgrading to. This offers a performance improvement, because it reduces the number of requests that are made to the WebSphere Portal Express server for the MobileFirst resources.

d. Leave the Cordova API as it is and add a `<script>` element to all pages that require MobileFirst resources. The `<script>` element must reference the `cordova.js` file directly.

8. In the `custom.theme.mobilefirst.ext` war project, edit the `WebContent\WEB-INF\plugin.xml` file.

   a. Change the plug-in ID from `wp.theme.worklight.ext` to `custom.theme.mobilefirst.ext`.

   b. Change the plug-in name from MobileFirst plug-ins to `Custom MobileFirst plugins`.

   c. Change the plug-in provider-name from IBM to your company's name.

   d. Find and replace all occurrences of the current MobileFirst version number with your MobileFirst version number. Find and replace version numbers in both the formats 000 and 0.0.0. These changes must correspond with the changes made previously to `mobilefirstprevious_version.json`. This `plugin.xml` file defines the version-dependent modules that are `prereqed` by the version-independent meta modules that are defined in `mobilefirstprevious_version.json` in the previous step.

   e. Locate the `mf_android_new_version` and `mf_ios_new_version` modules, where *new_version* is the new MobileFirst version number, and verify all of their subcontributions. Review the resources for the new version of MobileFirst, and if there are any differences, modify the subcontributions so that each resource file has a subcontribution. In these modules, each resource is explicitly listed when debug mode is enabled. When it is not enabled, WebSphere Portal Express provides a single JavaScript layer for the MobileFirst resources. Create a similar layer for the resources of the new version of MobileFirst. This layer must be defined as a subcontribution here. If you choose not to create a JavaScript layer for the MobileFirst resources, add and verify that required resources are listed as subcontributions.

   f. Locate the `mf_plugins_android_new_version` and `mf_plugins_ios_new_version` modules, where *new_version* is the new MobileFirst version number, and verify all of their subcontributions. Review the resources for the new version of MobileFirst, and if there are any differences, modify the subcontributions so that each resource file has a subcontribution. In these modules, each resource is explicitly listed when debug mode is enabled. When it is not enabled, WebSphere Portal Express

provides a single JavaScript layer for the MobileFirst resources. Create a similar layer for the resources of the new version of MobileFirst. This layer must be defined as a subcontribution here. If you choose not to create a JavaScript layer for the MobileFirst resources, add and verify that required resources are listed as subcontributions.

g. In the `custom.theme.mobilefirst.ext` war project, delete the `WebContent\previous_version` folder, where *previous_version* is the current MobileFirst version number, and all of its contents.

9. Export your customized ear as `custom_mobilefirst_extension.ear` from Eclipse or Rational Application Developer.
   a. Right-click **custom_mobilefirst_extension ear project** and select **Export** > **EAR** file.
   b. Click **Browse** and choose a destination folder and name of `custom_mobilefirst_extension.ear`.
   c. Click **Finish** to save the EAR file to your file system.
10. Add a `resources.mobilefirst.extensions.`*new_version* resource environment provider property in the WebSphere Integrated Solutions Console.
   a. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
   b. Click **Next** until you find the **WP ConfigService** resource environment provider and then select it.
   c. Click the **Custom properties** link.
   d. Click **New** from the toolbar.
   e. Enter `resources.mobilefirst.extensions.`*new_version* for the name, where *new_version* is replaced by your new MobileFirst version number.
   f. Enter `/custom/mobilefirstExt/`*new_version* for the value where *new_version* is replaced by your new MobileFirst version number.
   g. Enter the path to the MobileFirst v*new_version_number* extensions for the description where *new_version_number* is replaced by your new MobileFirst version number in the `v0.0.0` format.
   h. Click **OK**.
   i. Click **Save** to save directly to the master configuration.
11. Deploy the `custom_mobilefirst_extension.ear` in the WebSphere Integrated Solutions Console.
   a. Click **Applications** > **Application Types** > **WebSphere enterprise applications**.
   b. Click **Install** in the toolbar.
   c. Click **Browse** in the local file system, find, and select your `custom_mobilefirst_extension.ear` file and click **Next**.
   d. Take the defaults and click **Next**.
   e. Take the defaults and click **Next**.
   f. Verify that the Directory to install application field is blank so that it installs to the default location. Delete anything in the field if it is not.
   g. Take the defaults and click **Next**.
   h. Click **Finish**.
   i. When the EAR is done installing without error, click **Save** to save to the master configuration.

j. Find and check your new Custom MobileFirst Extension enterprise application and click **Start** from the toolbar. Your EAR is now installed at \wp_profile\installedApps*cell*\Custom Mobilefirst Extension.ear.

12. Modify your theme contributions to load your new version of the MobileFirst extensions. You continue to use the same meta module names in your theme profiles, such as wp_worklight_android and wp_worklight_ios. You must change the JSON file in the theme contributions folder to change which version-specific modules the meta modules load.
    a. Use WebDAV to connect to fs-type1:themes*yourtheme*\contributions.
    b. Delete the mobilefirst*previous_version*.json file, where *previous_version* is the previous MobileFirst version number. If you back up the file, move it to a different location. Do not rename the file .jsonbak, for example, because every file in the contributions folder is loaded by the system. You must remove the file completely.
    c. Copy your mobilefirst*new_version*.json file into the contributions folder. The file can be copied from \wp_profile\installedApps*cell*\Custom Mobilefirst Extension.ear\custom.theme.mobilefirst.ext.war\v*new_version*\mobilefirst*new_version*.json, where *new_version* is replaced by your MobileFirst version number. If you must revert to the previous version of MobileFirst, you can get the mobilefirst*previous_version*.json file back from \PortalServer\theme\wp.theme.worklight.ext\installedApps\wp.theme.worklight.ext.ear\wp.theme.worklight.ext.war\v*previous_version*\mobilefirst*previous_version*.json, where *previous_version* is the original MobileFirst version number.
13. Restart the WebSphere Portal Express server.

Results

The MobileFirst meta modules, such as wp_worklight_ext, now load and use the resources for your new version of MobileFirst.

Integrating with Brightcove

If you are using videos as part of your website, integrate your portal with the Brightcove video streaming server.

Brightcove is a video hosting and streaming service in the cloud that is designed to host large video files and serve simultaneous requests. Web Content Manager content repository and the Rich Media Edition Asset Management system are not designed for video. Brightcove also transcodes the video into different formats and bit rates for mobile users. Web Content Manager provides default integration with Brightcove. Users can seamlessly select video for the content author and a rendering of the video with the Brightcove cloud.

Security

Brightcove uses a model that is based on tokens and not user IDs and passwords or certificates. These tokens are stored as part of the portlet preferences and exported with xmlaccess exports of the portal configuration.

"Configuring WebSphere Portal Express to use Brightcove" on page 1033
Before you can integrate with Brightcove, you must configure WebSphere Portal Express.

Configuring WebSphere Portal Express to use Brightcove

Before you can integrate with Brightcove, you must configure WebSphere Portal Express.

Before you begin

You must have a Brightcove account to integrate it with WebSphere Portal Express.

Procedure

1. Log in to Brightcove.
2. Click **Account Settings** > **API Management**.
3. Select **Read Token** and **Upload Token**.
4. Run the ConfigEngine task to set up Brightcove. In the following command, replace *readToken* and *uploadToken* with the read and upload tokens you selected from Brightcove. Upload tokens are called Write tokens in Brightcove. Replace *wasPassword* and *wpsPassword* with the WebSphere Application Server and WebSphere Portal Express passwords.

   ```
   ConfigEngine.sh setup-brightcove-plugins -DBrightcove.ReadToken=readToken -DBrightcove.UploadT
   ```

   By default, this task is performed on the base portal. To run this task on a different virtual portal, identify the virtual portal by adding one of the following parameters to the command line. Each parameter requires the prefix **-D** on the command line.

   VirtualPortalHostName
   Specify the host name of the virtual portal. For example, vp.*your_host*.com.

VirtualPortalContext
Specify the virtual portal context that identifies the virtual portal. For example, `vp1`.

5. Restart WebSphere Portal Express.

   Note: For cluster installs, the ConfigEngine task must be performed the primary node only. The restart must be performed on all nodes in the cluster.

Results

The Brightcove setup task performs the following actions.

- Install the `dam.brightcove` WAR file that contains the integration plug-in and the Brightcove video management portlet with unique name `ibm.portal.Brightcove`.
- Sets the preferences `BC_READ_TOKEN` and `BC_UPLOAD_TOKEN` to the provided values.
- Creates a hidden page with unique name `ibm.portal.page.hidden.Brightcove` and puts the Brightcove video management portlet on that page.

Updating the Brightcove read or upload tokens

The read or upload Brightcove tokens can be updated with a ConfigEngine task.

Procedure

1. To update the Brightcove tokens in WebSphere Portal Express, run a ConfigEngine task to set your portlet preferences. Use the same *readToken* and *uploadToken* from Brightcove. If you do not set these values, the test tokens are installed. Your *readToken* must have URL access to enable the preview video on the video details page.

   `ConfigEngine.sh set-portlet-preferences.brightcove -DBrightcove.ReadToken=`*`readToken`* `-DBrightcove.`

   By default, this task is performed on the base portal. To run this task on a different virtual portal, identify the virtual portal by adding one of the following parameters to the command line. Each parameter requires the prefix **-D** on the command line.

   VirtualPortalHostName
   Specify the host name of the virtual portal. For example, `vp.`*`your_host`*`.com`.

   VirtualPortalContext
   Specify the virtual portal context that identifies the virtual portal. For example, `vp1`.

2. Restart WebSphere Portal Express.

   Note: For cluster installs, the ConfigEngine task must be performed the primary node only. The restart must be performed on all nodes in the cluster.

Brightcove video management portlet

The Brightcove video management portlet renders the selection user interface for selecting a video. It uses the public Brightcove REST API to connect to Brightcove and retrieve the data. It connects with the Outbound and Ajax proxy.

Portlet configuration settings

Access the Brightcove video management portlet. Click the **Administration menu** icon. Then, click **Portlet Management** > **Portlets**. Search for the Brightcove video management portlet. Click **Configure**.

BC_BASE_LIBRARY_URL
Default: `http://api.brightcove.com/services/library` Brightcove URL used for the REST read calls.

BC_BASE_URL
Default: `https://videocloud.brightcove.com/`. Brightcove URL started for the **Launch Video Manager** button.

BC_DEFAULT_SORT_TYPE
Default: `Publish Date`. Default sort criteria for the videos list. Valid values: `Publish date, Creation date, Last modified date, Total plays, Plays this week`.

BC_META_DATA
Default:
`id,name,shortDescription,linkURL,linkText,tags,thumbnailURL,referenceId,FLVURL,ec`
Metadata that is retrieved from Brightcove from the REST call.

BC_READ_TOKEN
Default: `none`. Brightcove Read REST token, specific to the Brightcove account, see Update tokens section.

BC_UPLOAD_TOKEN
Brightcove Write REST token, specific to the Brightcove account, For more information, see *Update the Brightcove read or upload tokens*.

BC_UPLOAD_URL
Default: `http://api.brightcove.com/services/post`. Brightcove URL for uploading videos.

BC_USE_AJAX_PROXY
Enables or disables the Ajax proxy. If you disable the Ajax proxy, upload no longer works because of browser security restrictions.

Configuring the Outbound/AJAX proxy

The default configuration is stored in the `proxy-config.xml` of the Brightcove WAR file. For more information, see *Set up Ajax proxy*.

Related tasks:

"Set up Ajax proxy" on page 748
The support for community pages uses the Ajax proxy to access the remote server. The Ajax proxy is updated for the base Connections URLS during the Connections Portlets installation. If FileNet is used, you must still configure the Ajax proxy manually to update for FileNet. Configure the Ajax proxy so that direct requests that the CCM portlet makes to the FileNet server are allowed to pass through the proxy Server.

"Updating the Brightcove read or upload tokens" on page 1034
The read or upload Brightcove tokens can be updated with a ConfigEngine task.

Overriding default configurations

The Brightcove portlet stores some default values, such as the Brightcove REST read and upload token or the default view to render. You can override the read REST token, the upload REST token, and the view configuration settings with the content template.

About this task

When you override the default configuration settings, you can make specific content templates for playlists. Or you can create content templates for a new account.

Procedure

1. In the Brightcove portlet preferences, configure the content template elements `BC_OVERRIDE_READ_TOKEN`, `BC_OVERRIDE_UPLOAD_TOKEN`, and `BC_OVERRIDE_VIEW`. For example, set `BC_OVERRIDE_READ_TOKEN` to read, `BC_OVERRIDE_UPLOAD_TOKEN` to upload and `BC_OVERRIDE_VIEW` to view.
2. Define the corresponding elements in then default content. The elements must be of type text or OptionSelection. The values defined in the content template and default content are active. The elements can be hidden from the content authors creating content items based on the these templates.

Brightcove selection user interface overview

Brightcove is integrated with the Web Content Manager user interface.

Selecting a video or playlist

After you install Brightcove, the authoring form of the File Resource Component or a File Resource element in a content items authoring template provides an option to choose from in the Brightcove Video Manager. If a user clicks **Select**, the Brightcove video management portlet starts in a new window that displays the videos from the Brightcove account that matches the read token. The user can select a video or playlist, or read the details of the selection. To preview a video, the browser must be able to display the uploaded file format.

Uploading a video

You can upload a new video. Click **Upload**. The video is not immediately available for playback on the website while Brightcove transcodes the video.

Starting the video management user interface

With **Launch Video Manager**, the full Brightcove user interface can be started in a new browser tab to work with videos. You must log in with the Brightcove user ID and password.

Search

You can search videos with the search box by the video name, description and tags. You can also click a tag on the detail page of a video to search for other videos with that tag.

To search for terms that start with a letter sequence, you can use the `*` as wildcard, such as `sam*` to find `sample`.

Related tasks:

Searching for Videos with the Media API

Integrate the Brightcove video player

The video that you select in your file element or file component does not render without a Brightcove video player. Brightcove provides a customizable player that you can copy and paste into a Web Content Manager presentation template or HTML component.

In this example, the video ID can be accessed with the format parameter **videoID** in component or element tags.

```
[Element context="current" type="content" key="Video" format="videoID"]
Accesses the video ID of a file element named "Video".
```

The following example replaces the hardcoded video player ID with the element tag. Replace the value *YOUR PLAYER TOKEN* and *YOUR PLAYER KEY* in the **playerID** and **playerKey** parameters.

```
<style type="text/css">
  .outer-container {
           position: relative;
           height: 0;
           padding-bottom: 56.25%;
         }
         .BrightcoveExperience {
           position: absolute;
           top: 0;
           left: 0;
           width: 100%;
           height: 100%;
         }
     </style>

<div id="container1" class="outer-container">
<!-- Start of Brightcove Player -->
<div style="display:none">

</div>

<!--
By use of this code snippet, I agree to the Brightcove Publisher T and C
found at https://accounts.brightcove.com/en/terms-and-conditions/.
-->

<script language="JavaScript" type="text/javascript" src="http://admin.brightcove.com/js/Brightcov

<object id="myExperience_[Property context="current" type="content" field="id"]" class="Brightcove
  <param name="wmode" value="transparent" />

  <param name="bgcolor" value="#FFFFFF" />
  <param name="playerID" value="YOUR PLAYER TOKEN" />
  <param name="playerKey" value="YOUR PLAYER KEY" />
  <param name="isVid" value="true" />
  <param name="isUI" value="true" />
  <param name="dynamicStreaming" value="true" />

  <param name="@videoPlayer" value="[Element context="current" type="content" key="Video" format="
</object>

<!--
This script tag will cause the Brightcove Players defined above it to be created as soon
as the line is read by the browser. If you wish to have the player instantiated only after
the rest of the HTML is processed and the page load is complete, remove the line.
-->
```

```
<script type="text/javascript">brightcove.createExperiences();</script>

<!-- End of Brightcove Player -->
</div>
```

If the following message displays for every video, `The video you are trying to watch is currently unavailable. Please check back soon.`, verify the player token and key are correct.

For a playlist player, the **playlistTabs** parameter must point to the ID of the currently selected playlist.

```
<param name="@playlistTabs" value="[Element context="current" type="content" key="Video" format="vide
```

You can store these markup templates as an HTML component and have different components for each player. Then, you can combine the file and player in one authoring template. For example, you can add a `File Element` called `Video` and a `Component Reference` called `Video Player`.

Then, display the player in the presentation template. In this example, you can change the selected video with inline editing.

```
[EditableElement context="current" type="content" key="Video" format="div"]
   [Element context="current" type="content" key="Video player"]
 [/EditableElement]
```

Note: The Brightcove video player is not supported in Opera.

Related tasks:

Creating an editable element tag

Uninstalling Brightcove

Uninstalling Brightcove removes the integration WAR file and the hidden portal page.

Procedure

1. Run the **remove-brightcove-plugins** ConfigEngine task. `ConfigEngine.sh remove-brightcove-plugins [-DWasPassword=`*wasPassword*`] [-DPortalAdminPwd=<`*wpsPassword*`]`. This task is performed on the base portal. But since it removes the WAR file, you must also remove the hidden pages from all virtual portals that have the Brightcove hidden page deployed.
2. To remove the hidden pages, run the **action-remove-pages-portlet.brightcove** ConfigEngine task. `ConfigEngine.sh action-remove-pages-portlet.brightcove [-DWasPassword=<`*wasPassword*`>] [-DPortalAdminPwd=<`*wpsPassword*`>`

   By default, this task is performed on the base portal. To run this task on a different virtual portal, identify the virtual portal by adding one of the following parameters to the command line. Each parameter requires the prefix **-D** on the command line.

   VirtualPortalHostName
   Specify the host name of the virtual portal. For example, `vp.`*`your_host`*`.com`.

   VirtualPortalContext
   Specify the virtual portal context that identifies the virtual portal. For example, `vp1`.
3. Restart WebSphere Portal Express.

Note: For cluster installs, the ConfigEngine task must be performed the primary node only. The restart must be performed on all nodes in the cluster.

Troubleshooting the Brightcove player

If you are having trouble with your videos, check this troubleshooting section to see if there are workarounds.

Brightcove video does not render on the screen

Check to see whether the video player markup has the correct player ID and key. Compare it to the markup from the Brightcove video cloud user interface. If that doesn't work, check to see whether the Brightcove cloud is experiencing any issues with the Brightcove Status page.

Brightcove video cannot be edited

Check to see whether the Brightcove plug-in was successfully deployed and if the read and upload tokens are correct. For more information, see *Brightcove video management portlet*.

Video on subscriber portals do not play

Without the plug-in and proper tokens, Web Content Manager cannot play the video on a subscriber portal. Syndicate a virtual portal instead. The Brightcove plug-in is global and available to base and virtual portals.

Limitations and known issues

- The Video details view contains a preview of the video. This preview renders the originally uploaded video, and creates an error if the current browser does not support the video format.
- The **Upload** and **Delete** buttons do not display in Internet Explorer. Internet Explorer does not support JSON return messages.
- Brightcove searches use word stemming by default. For more information, see *Searching for videos in the media module*.
- Playlists are not searchable.
- Tags in Brightcove do not synchronize with WebSphere Portal Express and do not show up as tags in your portal.
- Page numbers do not immediately reflect additional video uploads.
- Internet Explorer 10 or older does not support media fragments in an HTML5 video player. The preview and setting starting and ending points with the slider function are not functional.
- For any playback issues on different devices and browsers, see the Brightcove support site for more information.

Related concepts:

"Brightcove video management portlet" on page 1034
The Brightcove video management portlet renders the selection user interface for selecting a video. It uses the public Brightcove REST API to connect to Brightcove and retrieve the data. It connects with the Outbound and Ajax proxy.

Related tasks:

Brightcove status

Chapter 11. Administering

Use the administration tools that are provided with the portal to do various day-to-day administration tasks. There are two methods for editing portal setup: using the administration portlets or the XML configuration interface. The administration portlets are a convenient way to make real-time updates to the portal's configuration. While the XML configuration interface is suited to more advanced administration, including batch processing of updates.

"Managing your site" on page 1296
Earlier versions of WebSphere Portal Express provided the Resource Manager portlet for performing site management. With Version 8.0 of WebSphere Portal Express and Web Content Manager, this site management functionality has been replaced by the new functionality for managing pages.

"Tagging and rating" on page 1297
Get an overview of the administrative tasks related to tagging and rating.

"Using WebDAV with WebSphere Portal Express" on page 1351
WebSphere Portal Express provides a Web-based Distributed Authoring and Versioning (WebDAV) implementation that individual services can use by plugging into. WebDAV is a set of extensions to the HTTP protocol that allows you to collaborate on editing and managing files on remote Web servers. Caching Proxy supports WebDAV methods used by Microsoft Exchange Server, and user-defined (customized) methods. These methods are hard coded and managed by the Enable and Disable directives. Administrators can also use the corresponding method-mask defined in the PROTECT directive to authorize the use of these methods.

"Virtual portals" on page 1361
View information on how you can scope your WebSphere Portal Express to have multiple virtual portals.

"Language support" on page 1419
To reach as many users as possible, WebSphere Portal Express supports different languages for different locations. For instance, a large, international corporation might address users in different countries or regions through multilingual Web sites. In this context the portal can concurrently serve portal views to large numbers of users, each in the user's preferred language.

"WSRP services" on page 1433
IBM WebSphere Portal Express supports the Web Services for Remote Portlets (WSRP) standard. By using this standard, portals can provide portlets, applications, and content as WSRP services. Other portals can then integrate the WSRP services as remote portlets for their users.

"Browser behavior and scenarios" on page 1506
Browser behavior for the back button, bookmarks and history affect user interaction and cause unexpected results.

Portal administration tools

Learn about the different tools that you can use to administer your portal.

You can administer and configure portal resources by using one of the following tools:

- The portal administration portlets.
- The portal XML configuration interface.
- The Portal Scripting Interface.
- The portal ReleaseBuilder.
- The configuration wizard.

The portal provides several administration tools for limited purposes. These tools are documented in the context where they can be used. An example is the SLCheckerTool, which you can use to delete orphaned data.

Security considerations

IBM WebSphere Portal Express provides a flexible delegation model for administering portal resources. This means that a master administrator can delegate administration and configuration work to subadministrators or other users as required in a highly detailed manner. For example, the master administrator can delegate the responsibility and rights for different administrative tasks to different departments in the same business. These departments can be for developing, deploying, and operating software solutions that are based on WebSphere Portal Express.

The delegation model is implemented by access control, which works by access control decisions, which guard the execution of administrative tasks that manipulate portal resources. Users can complete a task only if they have the access permissions that are required for that task. Access permissions are implemented as user rights on actions that are related to portal resources, not on the resources themselves. For more information, refer to the documentation about access control.

The extent to which the portal delegation model and access control is tied in varies between the portal administration tools. Security might therefore influence which tool you use for a certain purpose.

Overview of administration portlets

Portal administrative users can use the administration portlets for the following tasks:

- Completing administrative tasks and actions on portal resources, depending on the access rights that the administrative user has on those resources. These tasks include:
  - Configuring individual portal resources.
  - Configuring individual portal resources, together with their dependent resources. For example, this configuration can be pages and the pages that are derived from them.
- Giving other users, for example subadministrators, limited access rights on selected portal resources. These subadministrators can then complete administrative tasks that their access rights allow. As the master administrator, you can widen or limit that extent by modifying the access rights for these users on the portal resources. This way, you can delegate administrative tasks as required.
- Deploying your own custom developed artifacts, such as portlets, themes, or skins.

You cannot use the administration portlets to complete scripted or automated administration or configuration tasks.

For more information, refer to the documentation about the administration portlets that are supplied with WebSphere Portal Express.

Overview of the XML configuration interface

The XML configuration interface works as follows:

- The XML configuration interface provides a batch processing interface for portal configuration updates. It allows you to export an entire portal configuration or

parts of a configuration, for example-specific pages, to an XML file. You can then re-create the exported configuration from such a file on another portal.

- You access the XML configuration interface by using a command-line tool. This command-line client is a small separate program that connects to the server by using an HTTP connection. You can therefore use it remotely.
- You can use the XML configuration interface to process portal resources, but not portal actions or tasks.
- You can use the XML configuration interface to process the configuration of portal resources that exist, for example pages. In this context, the XML configuration interface processes derived resources, but it does not automatically create them.
- The XML configuration interface does not reflect the access control authorization model with delegated administration. You need only the access permission to use the XML configuration interface. An administrator who works with the XML configuration interface does not need access permission for the portal resources that are processed by the XML request. (The reason for this is that access control gives users access permissions on actions and not on resources.)

You can use the XML configuration interface for the following tasks:

- Exporting, importing, and updating complete or partial portal installations. These tasks can be for the following purposes:
  - Transfer or migration between workstations
  - Back up of the portal configuration
  - Overview of the portal configuration.
  - Cloning of a portal. To complete this step, you use the object ID generation mode of the XML configuration interface.
- Copying parts of a configuration, such as specific pages, from one portal to another.
- Transferring portal configurations from one installation to another. You do this transfer by exporting and importing the portal configuration. This usage scenario includes the case where you try out a new portal configuration on a test portal for evaluation, and then transfer it to a production portal in a separate step by using the portal configuration interface.
- Creating a portal configuration file by XML export. You complete this step by an XML export.
- Installing extra resources on a portal.
- Completing recurring administration tasks in an automated and reproducible manner.
- Completing these administrative tasks remotely, that is, from another server through an HTTP connection.

Security: A user who uses the XML configuration interface to complete administrative tasks needs only the access permission on the virtual resource XML_ACCESS. The user does not need access rights on the portal resources that are updated by the XML configuration interface.

Use of the XML configuration interface for the following tasks is limited:

- Delegating administrative tasks, that is, having other administrative users with specific access permissions complete these tasks.
- Limiting administrative tasks to a particular user or to particular portal resources.

For more information, refer to the documentation about the XML configuration interface.

The XML configuration interface is also used for release staging, that is, for staging a portal from development through test to production. For more information about staging your portal to production, refer to the topics about staging to production and ReleaseBuilder.

Overview of the Portal Scripting Interface

The Portal Scripting Interface works as follows:

- The Portal Scripting Interface is a command-line tool.
- The Portal Scripting Interface behaves just like the portal administration portlets. It provides delegated administration in the same manner as the portal administration portlets and access control. To work with Portal Scripting Interface, a user needs access permission on the WebSphere Portal Express and on the portal resources that the user administers.
- It allows implicit derivation during administrative work. This means when you modify a portal resource, the Portal Scripting Interface creates the derivations of that resource in the same process, depending on your access rights.

You can use the Portal Scripting Interface for the following tasks:

- Making fine-tuned changes to a portal configuration.
- Transferring configuration updates in a safe and controlled manner, and without disturbing the production portal. For example, this process can happen by the following steps:
  1. On a development system, a development team develops configuration updates for the portal, and the script for running these updates.
  2. After the script is completed, a test team tests both the script and the new configuration.
  3. After the script and the new configuration are tested and approved, they can be applied to the production portal.
  4. An operator team processes the scripts that update the production portal.

The Portal Scripting Interface has the following advantages:

- Security: The user IDs and access roles of the involved teams provide separation between the responsibilities for the subtasks:
  - The development and test team do not have access rights on the production portal.
  - The operator who runs the script must have access rights on the resources that are created and updated by the script. Therefore, if you limit the access rights for that user as required, the script cannot affect other resources unintentionally.
- Safety and availability of the production portal:
  - The scripts can be tested and verified before it is put into production.
  - After the scripts are tested and verified, they perform the update in a reliable way. Human errors that might happen when you are working with the administration portlets are not possible.
  - The production portal does not even require an Administration page for performing the update.
  - The update can be performed over night without disturbing production.

Use of the Portal Scripting Interface is limited in the following way:

- The Portal Scripting Interface offers only a subset of the functionality of the portal administration portlets. For details, refer to the Portal Scripting Interface command reference.

Overview of ReleaseBuilder

To generate or stage follow-on releases of IBM WebSphere Portal Express portals, configurations, and artifacts need to be moved between systems. ReleaseBuilder enables management of release configurations independent of user configurations.

Release configuration data are exported to XML files that can be imported using the XML configuration interface (XmlAccess). Using ReleaseBuilder it is possible to stage release configurations between two portals. This allows you to track which configuration entities were removed, added, or changed compared to the previous release generated from a given portal and to apply these differential updates to another portal. Detecting the differences between one configuration and another of the same portal server creates differential updates. A third configuration or "diff", generated by ReleaseBuilder, represents the changes made between the two configurations. The third configuration can be used to apply not only addition and update modifications but also deletions to the target server. This allows two portal servers, for example, a staging server and a production server, to remain in synch. ReleaseBuilder is designed to eliminate the need to generate complete XmlAccess exports to move a partial configuration or to manually create XML response files to export a partial configuration. ReleaseBuilder also helps to prevent the problem of configuration bloat on the target server.

Overview of Configuration Wizard

Use the Configuration Wizard to set up stand-alone servers and new deployments, create clusters, migrate and update to new versions, and add new capabilities to existing deployments.

In the Configuration Wizard, you answer questions about the environment that you are configuring. Based on your answers, the wizard prompts you for custom values that are needed to configure your environment. Finally, the wizard generates custom steps and scripts to set up your environment.

"Portal administration portlets" on page 1047
Administration portlets are supplied with IBM WebSphere Portal Express. Use them to perform administration tasks and actions on portal resources, give other users limited access rights on selected resources, and deploy custom portlets, themes, or skins.

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

"Portal Scripting Interface" on page 1113
You can use the Portal Scripting Interface to configure your portal by running scripts from a command line.

Related concepts:

"ReleaseBuilder" on page 3624
To generate or stage follow-on releases of IBM WebSphere Portal Express portals, configurations, and artifacts need to be moved between systems. ReleaseBuilder enables management of release configurations independent of user configurations.

"Configuration Wizard" on page 232
Use the Configuration Wizard to set up stand-alone servers and new deployments, create clusters, migrate and update to new versions, and add new capabilities to existing deployments.

"Updates using ReleaseBuilder" on page 3624
After setting up your initial staging and production servers, you can use ReleaseBuilder to make updates to your production server.

Related information:

Technotes for administration tools

Portal administration portlets

Administration portlets are supplied with IBM WebSphere Portal Express. Use them to perform administration tasks and actions on portal resources, give other users limited access rights on selected resources, and deploy custom portlets, themes, or skins.

"Administration portlets overview"
WebSphere Portal Express has administration portlets that assist you with managing resources. Get an overview of the administration portlets and the tasks you can complete with each portlet.

"Working with administration portlets" on page 1052
Working with the portal administration portlets allows you to customize the administration portlet and its features, such as, defining the number of rows to display in a table.

Related information:

Technotes for administration portlets

Administration portlets overview

WebSphere Portal Express has administration portlets that assist you with managing resources. Get an overview of the administration portlets and the tasks you can complete with each portlet.

Note: Subadministrators of virtual portals might not have access to all administration portlets listed here. For information, refer to the topics about multiple virtual portals.

Portal User Interface

The following sections describe the portlets that are provided for customizing the user interface for WebSphere Portal Express.

Manage Pages
Use the **Manage Pages** portlet to export page configurations in XML, create, edit, activate, order, and delete pages and external web pages and labels. Available tasks depend on which item is selected.

Themes and Skins
Use the **Themes and Skins** portlet to install, edit, and delete themes and skins. You can also select a default theme and skin with this portlet. See the Themes and Skins portlet help for specific instructions on these tasks.

Page Templates
Page Templates are not accessed by a portlet, but they provide a means for managing portal pages. Page templates include common page elements and make page creation easy. To work with page templates, users can, but

do not have to configure pages and their properties, such as theme or page layout. For example, web site designers can create and manage page templates that match your web site design. Content authors can then use the page templates to create content ready pages for different content types. Working with page templates is not limited to administrators. Users need only limited access permissions to work with page templates.

Portlet Management

The following sections describe the portlets that are provided for working with portlets, web modules, and applications.

Web Modules
Use the **Manage Web Modules** portlet to install new portlets from either a web service or a WAR file. You can also use the portlet to manage existing portlets or view a list of portlet applications for a web module. A web module is a WAR file that contains portlet applications.

Applications
Use the **Manage Applications** portlet to enable a portlet application as a web service or to manage existing portlet applications. It displays a list of all web modules and associated portlet applications that are installed on WebSphere Portal Express. You can view and change portlet application settings from this portlet. Tasks include renaming and deleting portlet applications, and modifying configuration parameters. See the Manage Applications portlet help for steps on completing these and related tasks.

Portlets
Use the **Manage Portlets** portlet to view or manage existing portlets, or enable portlets as web services. It displays a list of all installed portlets. Use the **Manage Portlets** to view and change portlet settings. Tasks include renaming and deleting portlets, and adding, modifying, or deleting portlet configuration parameters. See the Manage Portlets help for steps on completing these and related tasks.

Web Services
Use the **Web Service Configuration** portlet to set up your portal for consuming web services for Remote Portlets (WSRP) by configuring WSRP Producers on the Consumer side. For more details, refer to the topics about Using WSRP services.

You cannot use the **Web Service Configuration** portlet for the following actions:

- Provide web services that make your portlets available to other systems such as remote web services. To provide web services as a Producer, use the **Manage Portlets** portlet.
- Consume web services that integrate web services that are provided by a Producer as remote portlets. To consume web services as a Consumer, use the **Manage Web Modules** portlet.

Virtual Web Application Manager
Use the **Virtual Web Application Manager** portlet to create content provider profiles in your portal site. These profiles are required to integrate web-based content from different providers, such as Microsoft SharePoint. You can then create Web Dock applications based on the content provider application details that are rendered on a portal page in an iFrame.

Portal Access

Users and Groups

Use the **Users and Groups** portlet to search for, edit, and delete existing users and groups. You can also create new users and groups and modify group membership.

Resource Permissions

Use the **Resource Permissions** portlet to set access roles. You can assign access roles to associate users and groups with resources to determine the level of interaction a user can have with a resource.

User and Group Permissions

Use the **User and Group Permissions** portlet to easily assign, view, and modify the roles and permissions that users and groups have on various resources. Refer to the **User and Group Permissions** portlet help.

Credential Vault

Use the **Credential Vault** portlet to complete tasks that are specific to vault management. You can add or manage vault segments and vault slots.

Portal Settings

Global Settings

You can use the **Global Settings** portlet to define what the user sees, including the default language and the **Find** link. The default language that is specified in Global Settings applies to all users when the language preference specified in their browser is not supported. For example, the portal is configured to support English, German, and Spanish, with English as the default language in **Global Settings**. A user, whose browser language preference is set to Italian, would see English because Italian is not supported in this case. A user can also select a preferred language when they register.

The **Global Settings** portlet also determines what users see when they return to the portal. For example, you can choose to display the most recently viewed page rather than a default page. You can allow users themselves to choose what they see when they log on, or if they see the default page or the most recently viewed page. You can also determine a URL for the **Find** link. For more information about the **Find** link, refer to the topic about setting the search engine that opens when users select **Find**.

Note: The **Global Settings** portlet does not work in portal cluster configurations.

Custom Unique Names

WebSphere Portal Express uses object IDs to identify resources unambiguously even between different portals. They consist of an extended alphanumeric string that might be difficult to remember. Use the **Custom Unique Names** portlet to assign unique names to resources. You can select names that are easy to read and remember. These custom unique names make identification of resources easier than the object IDs assigned by WebSphere Portal Express, for example when porting resources from one portal to another.

Supported Markups

Use the **Supported Markups** portlet to determine which markups are

recognized. You can add, edit, activate or deactivate, and delete a markup. The installation default is HTML. See the **Supported Markups** portlet help for detailed instructions.

Note: Leave the default HTML markup enabled. Removing or changing the HTML markup causes access problems. If the default HTML markup is disabled, use the XML configuration interface to re-enable the HTML markup.

Supported Clients
With the **Supported Clients** portlet, you can determine what types of devices and web browsers can access the portal. You can add, edit, order, or delete clients. If you need to test a portlet with a device simulator, you might need to add the user agent string of the device simulator to the portal client list. Consult the documentation included with the device simulator to determine the user agent strings the simulator supports and add them with the **Supported Clients** portlet. Read the **Supported Clients** portlet help for detailed steps on working with clients.

Import XML
Use the **Import XML** portlet to import an XML file. For example, from a staging server you can export pages and portlets into XML with **XML export** on **Manage Pages** and then use the **Import XML** portlet to import the configuration to a production server.

Portal Content

Web Content Libraries
Add, edit, copy, and delete web content libraries to better manage your web content. You can also specify access control settings for the library itself and the types of content it contains.

Syndicators
Create, edit, and delete syndicators that are used to replicate content between Web Content Manager environments. Syndicators identify the libraries that are available for replication by subscribers. You can define which libraries are available for syndication and change the order that libraries are syndicated.

Subscribers
Create, edit, and delete subscribers that are used to replicate content between Web Content Manager environments. Subscribers are associated with a syndicator and receive updates from all libraries that are specified by the syndicator.

Feed Configuration
The Feed Configuration portlet helps you create and manage RSS feed consumers for creating and updating web content.

Feed Jobs
The Feed Jobs portlet helps you create and manage schedules for the web content RSS feed consumers.

Search Administration

Manage Search
Use the **Manage Search** portlet to create and manage search services, search collections, and search scopes. Searchable resources include various document types, for example HTML and text documents. WebSphere

Portal Express sites can also be indexed and searched. Refer to the Manage Search portlet help for detailed instructions about working with search.

Portal Analysis

Theme Analyzer
Use the Theme Analyzer to view, but not edit, all parts of the theme optimization framework of WebSphere Portal Express. With this portlet, you can view which pages have specific profiles that are set or inherited. You can also see which profiles are available and belong to which theme.

Additionally, you can browse and explore all aspects of the available modules: You can see which modules are loaded for a specific profile or all modules of the whole system. You can drill down into the dependency hierarchy to understand interdependencies and get different views on it, such as a parent view. The module explorer also features a rich search set so that you can easily find modules that contribute certain resources or capabilities, or browse all exposed data.

This portlet also provides a number of utilities to help you debug theme issues such as client-side tracing, export module languages, and other debugging tools. In case of problems, you can also export your set of data as a compressed file and share it with others. They can then import your data set and examine your profiles and modules.

Frequent Users
The **Frequent Users** portlet shows how many users are logged in for the past 90 days.

Enable Tracing
Use the **Enable Tracing** portlet to enable or disable the tracing logs. See the Enable Tracing portlet help for detailed instructions on working with logs.

Use the **Enable Tracing** portlet to dynamically enable or disable trace-logging for individual classes and entire packages without restarting WebSphere Portal. In a WebSphere Portal cluster, the portlet lists and changes the currently running trace loggers on only one individual server (not the entire cluster). To dynamically change the trace specification for the entire cluster, you must use the Enable Tracing portlet on each individual server (horizontal or vertical cluster member).

About WebSphere Portal Express
This portlet shows the version and fix level of your WebSphere Portal Express. It also shows the product numbers and the Copyright years.

Manage Virtual Portals

Manage Virtual Portals
Use the **Virtual Portal Manager** portlet to create, list, modify, and delete virtual portals. When you create a virtual portal, it is filled with the initial default content for virtual portals. For more information about the **Virtual Portal Manager**, refer to the portlet help.

Other portlets that are useful for administration

There are other portlets not accessible from Administration that is also useful in administering WebSphere Portal Express.

Site Map

WebSphere Portal Express provides the **Site Map** portlet. It serves two purposes:

- You can use the Site Map portlet to browse the site. It displays a list of the pages and portlets. You can access this portlet by logging in to the portal, clicking **Search Center**, then clicking the **Site Map** tab
- The Site Map portlet enables external search crawlers to collect pages more efficiently.

Properties

Use the **Properties** portlet to modify properties on pages and portlets. You can access this portlet when you create a new page, editing an existing page, or from the **Manage pages** administration portlet.

Note: The current limitations for the **Properties** portlet are:

- The list of shareable pages that are displayed by the **Properties** portlet is limited to those pages that the user can go to. As a result, a user might not be able to create explicitly derived pages from some shareable pages.
- A user must have at least Editor privileges on a page to be able to edit the properties on that page. Privileged users are not able to edit a page's properties or edit personalization rules because Personalization rules are part of a page's properties.

Working with administration portlets

Working with the portal administration portlets allows you to customize the administration portlet and its features, such as, defining the number of rows to display in a table.

To work with the portal administration portlets, click the **Administration menu** icon to open the administration menu. The portal displays the navigation for the administration portlets.

Before you use the portal administration portlets, read the concept and background information for administrative tasks. Refer to the appropriate topics in the *Administering* section of this portal information center. For detailed steps for performing portal administrative tasks, refer to the help of each portlet.

You can customize the administration portlets by using the configure mode of the portlet. For example, you can configure features such as the layout of the portlet and the number of rows to display.

Searching for portal resources in administration portlets

Portal administration portlets that list portal resources, such as pages, portlets, users, or virtual portals, provide a search feature. You can search for resources available in the portlet. You can use an asterisk (*) as a wildcard character for your search. Following is a list of some search types. Their availability differs between portlets. It depends on the portal resource type that the portlet administers.

- **Title starts with:** Select this option to search on the beginning of a string in the title. This setting is the default setting, and the input is expected in string format.
- **Title contains:** Select this option to search on a string in the title. The input is expected in string format.

- **Name starts with:** Select this option to search on the beginning of a string in the name. The input is expected in string format.
- **Name contains:** Select this option to search on a string in the name. The input is expected in string format.
- **Keyword starts with:** Select this option to search on the beginning of a keyword. The input is expected in string format.
- **Keyword contains:** Select this option to search on a keyword. The input is expected in string format.
- **Description starts with:** Select this option to search on the beginning of a string in the description. The input is expected in string format.
- **Description contains:** Select this option to search on a string in description. The input is expected in string format.
- **Unique Name start with:** Select this option to search the beginning of a string in the unique name. The input is expected in string format.
- **Unique Name contains:** Select this option to search on a string in the unique name. The input is expected in string format.
- **Markup starts with:** Select this option to search the beginning of a string in the markup type. This search returns a list of pages that support that markup. The input is expected in string format.
- **Markup contains:** Select this option to search on a string in the markup type. This search returns a list of pages that support that markup. The input is expected in string format.
- **Label:** Select this option to search on a URL context label. The input is expected in string format.
- **Attributes:** Select this option to search on a user or group attribute. The input is expected in string format.
- **Last modified:** Select this option to search by items that were modified on or since a specific date. The input is expected in the format `YYYY MM DD`.
- **All available:** Select this option to return a listing of all items. No input is required.

Notes:

For title search:

By default, searches for **Title starts with** and **Title contains** return only portal resources with titles in the locale of the current portal session. You can configure portal administration portlets to return search results across all supported locales, regardless of the locale of the current portal session or the locale set in the browser. To do this configuration, proceed as follows:

1. Access the **Manage Portlets portlet**.
2. Locate the administration portlet for which you want to configure search of resources across all locales. To locate, use the search feature that is described earlier.
3. Click the **Configure portlet** icon for the portlet.
4. Add the parameter `searchByTitle.all.locales` with the value `true`.
5. Click **Add** and **OK**.

The default value of the parameter `searchByTitle.all.locales` is `false`. Searches for portal resources return results only in the locale of the current portal session or the locale set in the browser.

Note: Configuring search across all locales can have an impact on the search performance.

.

For the Users and Groups portlet:

Searching with wild characters, such as the asterisk (*), is only supported for the User and Groups and User and Groups Permissions portlets. Wildcard character searches for the User and Groups and User and Groups Permissions portlets must have the asterisk (*) at the end or beginning of the search parameter. For example, the following searches are supported for the User and Groups and User and Groups Permissions portlets:

- *user
- user*
- *user*

The following searches are not supported for the User and Groups and User and Groups Permissions portlets:

- us*r

The XML configuration interface

Use the XML configuration interface (XML Access) for exchanging portal configurations.

"About the XML configuration interface" on page 1055
The XML configuration interface provides a batch processing interface for portal configuration updates. It allows you to export an entire portal configuration or parts of a configuration, for example specific pages, to an XML file. You can then re-create the exported configuration from such a file on another portal.

"Changes to the XML configuration interface for this version of WebSphere Portal Express" on page 1059
Learn about the changes that were made to the XML configuration interface from earlier versions to current versions of IBM WebSphere Portal Express. This information can be useful to you if you migrate your WebSphere Portal Express from one version to a later version.

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

"XML configuration reference" on page 1082
Learn more about XML input structure, XML tags for portal resources, attributes for special purposes such as locale data, object IDs, and more. Also find information about action attributes that determine the type of processing that the XML configuration reference applies to the portal resource. There are syntax restrictions that you need to consider as well as information about how to determine the object IDs for portal resources.

Related tasks:

"Using the XML configuration interface to work with Producer definitions" on page 1477
You can use the XML configuration interface to work with Producer definitions in different ways.

"Using the XML configuration interface to consume portlets from a Producer portal" on page 1484
You can use the XML configuration interface (XMLAccess) to consume portlets from a Producer portal.

"Creating a web content page with the XML configuration interface" on page 2074
As with other portal pages, you can create a web content page with the XML configuration interface (**xmlaccess** command). Page definition is similar to a standard portal page. However, there is an additional page parameter that specifies the site area that is associated with the web content page.

Related reference:

"XML configuration interface parameters for the Web Content Viewer" on page 415
As with other portlets in your portal, you can use the XML configuration interface (**xmlaccess** command) to deploy and configure the Web Content Viewer. To simplify the configuration of the portlet with the XML configuration interface, the portlet parameters you can specify accept path values in addition to the standard IDs.

"XML configuration interface and content associations" on page 2076
With the XML configuration interface (**xmlaccess** command), you can perform batch updates of content associations or export associations to import into another portal. Content association information is represented in the XML configuration schema by content-mapping-info elements.

Related information:

 Technotes for XML Access and Release Builder

About the XML configuration interface

The XML configuration interface provides a batch processing interface for portal configuration updates. It allows you to export an entire portal configuration or parts of a configuration, for example specific pages, to an XML file. You can then re-create the exported configuration from such a file on another portal.

How do I access the XML configuration interface?

You access the XML configuration interface using a command line tool. This command line client is a small separate program that connects to the server using an HTTP connection or a secure HTTPS connection with SSL. It is therefore possible to configure the portal remotely.

Tasks that you can perform with the XML configuration interface

These are typical tasks for which you use the XML configuration interface:

- Copy parts of a configuration, such as specific pages, from one portal to another. This usage scenario includes the case where you try out a new portal configuration on a test portal for evaluation, and then transfer it to a production portal in a separate step using the portal configuration interface.
- Install additional resources on a portal.
- Perform recurring administration tasks in an automated and reproducible manner.

Use of the XML configuration interface for backing up or restoring complete portal configurations is restricted by the following limitations:

1. A complete XML export of a portal configuration is not sufficient to re-create the portal. You also need the WAR files for your portlets and possibly additional file resources, such as theme files if they are not part of the standard portal installation.
2. The XML configuration interface is not designed to deal efficiently with large volumes of data. For a backup and restore solution on a production server, you should rely on low-level database and file system backups.

Access and security considerations

To be able to use the XML configuration interface, you need to have the manager role on the virtual resource `XML_ACCESS` and the security administrator role on the virtual resource PORTAL. This implies that you must be a super administrator of the portal, who can perform any action. Consequently, there are no further access control checks that could restrict your actions when you use the XML configuration interface; you may view all resources in the portal and you may update and delete all resources.

When you run the XML command line tool, you must authenticate yourself by specifying your portal user ID and password. When you use an HTTP connection, the user and password are sent to the server unencrypted, therefore you should only connect to the XML configuration interface from inside a protected intranet where you can be sure that the HTTP connection is not compromised. In all other networks configure SSL and use a secure HTTP**S** connection to connect to the XML configuration interface.

Overall structure of the XML input and output

There are two main types of requests that can be sent to the XML configuration interface:

Export requests
: An export request triggers the export of complete or partial portal configurations into XML. It does not modify the configuration of the portal. It results in a response file.

Update requests
: An update request modifies the configuration of the portal according to the values found in the XML script.

A third request type is available for preparing the deletion of orphaned data:

Export-orphaned-data requests
: An export-orphaned-data request exports the complete portal configuration into XML, including orphaned data. It results in a response file.

Requests to and responses from the XML configuration interface use the same XML format. An export request generates an XML response that contains all the configuration data required to re-create the exported configuration part. This means that you can export a portal configuration, save the XML output file and, without modification, send it to another portal to re-create the same configuration there.

Use the XML schema for the XML format that WebSphere Portal Express provides for reference. You will find it in the JAR file `wp.xml.jar` in the WebSphere Portal Express installation directory:

- Linux: *PortalServer_root*`/base/wp.xml/shared/app`
- IBM i: *PortalServer_root*`/base/wp.xml/shared/app`
- Windows: *PortalServer_root*`\base\wp.xml\shared\app`

Unpack the JAR file and you will find the file with the XML schema under the path `com/ibm/wps/command/xml/PortalConfig_8.5.0.xsd`. An XML request contains the following:

- A mandatory `portal` section; it describes the parts of the portal configuration that should be exported or updated

- An optional status section. In an XML response it indicates the success or failure of the requested operation. During the import of configuration data the XML processing ignores this section of the XML input file.

Representation of a portal configuration in XML

The XML hierarchy that is found under the `portal` section in the XML request file represents the structure of a portal as an XML tree. This tree contains resources in the portal, such as portlets or pages, and their configuration data. The XML hierarchy of all supported portal resources is shown in the following table:

Table 136. Tree hierarchy of portal resources in the portal section of an XML request

XML element	Description
`portal`	Main element of every XML request
`global-settings`	Global portal settings
`services-settings`	Global portal settings for portal services
`language`	Languages that are defined in portal
`task`	Tasks that can be used to schedule programs
`action`	Actions that can be used to create action sets
`action-set`	Action sets that can be used to create roles. They are also known as Role Types.
`virtual-resource`	Virtual resources that have associated access control settings
`resource-type`	Resource types that you can use to create custom resources.
`protected-resource`	A resource instance that is protected by Portal Access Control (PAC).
`user`	Users defined in the portal user management system
`group`	Groups defined in the portal user management system
`markup`	Markups that can be supported by portal pages
`client`	Client devices (browsers) that the portal knows about
`device-class`	Device class information
`skin`	Visual appearance settings that can be applied to user interface elements
`theme`	General visual settings that can be applied to the user interface
`wsrp-producer`	Producer of web services as defined in the consumer portal
`wsdl-url`	The URL to the Producer's WSDL document
`porttype`	The URL to the service description, markup, registration, or portlet management of the Producer
`web-app`	Web modules containing portlets
`url`	The WAR file that contains the web application
`context-root`	The context root that is assigned to the web application of the portlet application in the predeployed EAR file (reference: application.xml)
`display-name`	The name that is assigned to the application in the predeployed EAR file (reference: application.xml)

Table 136. Tree hierarchy of portal resources in the portal section of an XML request (continued)

XML element	Description
servlet	Servlets that are defined in the web module
portlet-app	Portlet applications that are defined in the web module
portlet	Portlets that are defined in the portlet application
federation-server	The federation server definition. This server is used to retrieve content nodes.
content-node	Elements of the portal content tree (pages or labels)
supported-markup	The markups that are supported by this content node
allowed-portlet	The portlets that are allowed on this page
component	Layout components of pages
component	Subcomponents in the structure of the page
portletinstance	Occurrences of a portlet on a page with customized settings
cross-page-wire	Property broker wiring between two portlet instances. **Note:** The `wire` tag has been deprecated with WebSphere Portal Express Version 7, as it supports property broker wiring between two portlets *on the same page* only. Use the `cross-page-wire` tag as it supports property broker wiring between portlets on the same page and on different pages.
credential-segment	Segments for storing credentials in the credential vault
credential-slot	Slots in a credential segment that hold a credential
url-mapping-context	User defined URLs that map to pages in the portal
user-resource	Allows exporting and deletion of a specific user resources.
policy-node	Policies that are defined in the portal
application-role	A named set of authorization roles that can be assigned to users or groups.
wsrp-customized-portletinstance	A customized occurrence of a portlet provided by WSRP on a Producer portal
custom-resource	A custom resource that can be tagged or rated by users
category-instance	A category assigned to a custom resource
tag	A tag applied to a resource by a user
rating	A rating applied to a resource by a user
filter-instance	A filter for preprocessing data before the data is finally stored.

Depending on the content of an XML request, these resources can be created, modified, deleted or exported. An XML request can contain any number of such resource definitions. It can therefore create hundreds of new resources in one step or modify only a single configuration setting of one existing resource.

Related concepts:

"Portal administration tools" on page 1042
Learn about the different tools that you can use to administer your portal.

Related tasks:

"Setting up SSL" on page 1596
Get an overview of the tasks that are required to configure SSL for IBM WebSphere Portal Express. Some of these tasks are completed on the IBM WebSphere Application Server and the web server. The steps that refer to the WebSphere Application Server and the web server are summarized here; refer to the WebSphere Application Server and the web server documentation for detailed information. Steps that are unique to WebSphere Portal Express are described in detail here.

"Deleting orphaned data" on page 280
You use the SLCheckerTool to delete orphaned data in the database.

Changes to the XML configuration interface for this version of WebSphere Portal Express

Learn about the changes that were made to the XML configuration interface from earlier versions to current versions of IBM WebSphere Portal Express. This information can be useful to you if you migrate your WebSphere Portal Express from one version to a later version.

Changed XML schema

The XML schema is updated for new versions of WebSphere Portal Express as required. Make sure that your XML scripts specify the correct version of the XML schema according to the version of your portal installation. For example, for portal Version 8.5 specify the current version of the XML schema as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<request
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
 type="export | update">
  . . . configuration . . .
</request>
```

The different versions of WebSphere Portal Express use the following syntax definitions for the XML configuration interface:

Table 137. XML syntax definitions for different WebSphere Portal Express versions

WebSphere Portal Express Version	XML schema
Version 7.0.0	7.0.0
Version 7.0.0.2	7.0.0_2
Version 8.0.0	8.0.0
Version 8.5	8.5.0

Notes:

1. **Backward compatibility:** The later XML schemas are backward compatible with earlier supported versions of WebSphere Portal Express. This means that you can run XML scripts from earlier portal versions that IBM supports under a later version of the portal. For example, you can run an XML script that is based on the `PortalConfig_7.0.0.xsd` under portal Version 8.5. In such cases a warning message is written to the output script, which informs that a previous version of the XML schema was used.
2. **Schema file naming convention:** If the XML schema is enhanced by updates, a new version of the xsd file with a new file name is created according to the following naming convention:

- Starting with portal V 6.0.1 the schema file name includes the portal version number as follows: `PortalConfig_portal_version_number.xsd`. Example: `PortalConfig_7.0.0.xsd`.
- If the schema is enhanced during a portal version, the name includes additional ID information to ensure unique schema file names: `PortalConfig_portal_version_number_id.xsd`. Example: `PortalConfig_7.0.0_2.xsd`.

New XML resources in WebSphere Portal Express Version 8.5

In WebSphere Portal Express Version 8.5 the following new resource tags have been introduced:

Table 138. New resource tags

New XML configuration interface tags in portal Version 8.5	Tag specifies the following type of portal resource
`device-class`	a device class
`global-target-settings`	a section containing the cross-page wire settings that are set as global targets
`target`	a global target

For more details about this tag refer to the XML configuration interface reference.

New XML attributes in WebSphere Portal Express Version 8.5

This section lists the attributes that have been added to WebSphere Portal Express for Version 8.5.

- A new boolean flag `system` has been introduced for content-mapping tags.
- You can now set `parameter` sections for task nodes.
- A new attribute `target-portletdefinitionref` has been introduced on cross-page-wire items,

Removed XML resources in WebSphere Portal Express Version 8.5

The following XML resource is no longer supported in portal Version 8.5:

- `event-handler`

Setting the project scope in WebSphere Portal Express Version 8.5

The managed pages feature enables you to edit portal resources, such as pages, in the scope of a project. By working in a project, you can create, update, and approve pages in a draft state, without affecting the live server. You can specify a project scope for actions performed with the XML configuration interface command by including the object ID of the project in the URL. See *XML configuration interface and managed pages* for details.

Related reference:

"XML configuration interface and managed pages" on page 1271
You can use the XML configuration interface (XML Access) to manipulate managed pages just as you can for other portal resources.

Working with the XML configuration interface

You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Using the XML configuration interface

IBM WebSphere Portal Express provides a highly configurable framework of services to accommodate the different scenarios that portals of today need to address. The framework allows convenient replacement of service implementations as well as modification of the configuration of each service. When you work with the XML configuration interface, the results, especially of configuration imports or updates, are influenced by the property settings in these configuration services. This includes changes that you have made to these settings. For more detail about the configuration services, refer to the topics about the portal service configuration.

There are two ways to use the XML configuration interface to work with portal configuration data:

- By using the XML configuration command line client. The XML configuration command line client provides all the XML configuration interface functions.
- By using administration portlets you can export and import XML configurations.

Both options are described in the following sections.

Terminology

These topics use the following terminology in the context of the XML configuration interface:

export This term can have either of the following meanings, depending on the context:

- In the administrative context: a human administrative task, for example exporting a portal configuration or a part of it
- In the context of the XML configuration interface: an XML request and action, for example to export the data of a portal configuration or a part of it.

import

This term represents only the human task in the administrative context, for example importing a portal configuration or a part of it. It has no corresponding XML request type. In the context of the XML configuration interface, an import is performed by specifying the `update` request type, together with the `create` or `update` action for the resources that are to be imported.

The topics in the documentation describe the portal XML configuration interface. Internally within the portal, this tool is named XMLAccess.

Sample XML scripts

The following topics mention several sample files suitable for different purposes of portal configuration using XML. Before you use them, read the other topics about the XML configuration interface carefully. These sample files are documented here for reference purposes only. If you want to use the XML samples for work on your portal configuration, use the files provided in your portal installation, as they

might be more up to date than this documentation. The XML sample files are located in the following directory of your WebSphere Portal Express installation:

- Linux: *PortalServer_root*/doc/xml-samples
- IBM i: *PortalServer_root*/doc/xml-samples
- Windows: *PortalServer_root*\doc\xml-samples

"Using administrative portlets for XML configuration"
XML configuration through administrative portlets allows you to export and import configurations. You can export a page or an entire page hierarchy by clicking the **Export** icon in the Manage pages administration portlet. You can also import an XML configuration file by using the Import XML portlet.

"Using the XML configuration command line client" on page 1064
You access the XML configuration interface using a command line tool. This command line client is a separate program that connects to the server. Connecting to a **remote** server makes it possible to configure the portal remotely. You use the command line syntax of the XML configuration interface.

"Generating a complete XMLAccess export of a Portal configuration" on page 1081
To help troubleshooting a WebSphere Portal issue, you might be asked by IBM Support to generate or collect a full or complete export of your Portal configuration. The complete export can be generated by using the XML Configuration Interface, commonly known as XMLAccess.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

"Using the XML configuration interface to work with Producer definitions" on page 1477
You can use the XML configuration interface to work with Producer definitions in different ways.

"Using the XML configuration interface to consume portlets from a Producer portal" on page 1484
You can use the XML configuration interface (XMLAccess) to consume portlets from a Producer portal.

Related reference:

"Credential Vault Service" on page 332
You can use the portal Credential Vault Service to configure Vault Adapter implementations that are used by the Credential Vault Service to store credential secrets.

Related information:

"Portal service configuration" on page 289
IBM WebSphere Portal Express comprises a framework of configuration services to accommodate the different scenarios that portals of today need to address. You can configure some of these services.

Using administrative portlets for XML configuration:

XML configuration through administrative portlets allows you to export and import configurations. You can export a page or an entire page hierarchy by clicking the **Export** icon in the Manage pages administration portlet. You can also import an XML configuration file by using the Import XML portlet.

About this task

Notes:

- You must enable support for JavaScript and disable pop-up blocking in your browser settings.
- Do not use the administration portlets to export or import complete portal configurations. For details about how to transfer a complete configuration see the related reference information.

  "Exporting pages or page hierarchies by using the Manage Pages portlet"
  You can do an XML export of a page or an entire page hierarchy by using the Manage Pages portlet.

  "Importing pages or page hierarchies by using the XML Import portlet"
  You can import an XML configuration file by using the XML Import portlet.

Related tasks:

"Transferring a complete configuration" on page 1074
For transferring complete portal configurations, WebSphere Portal Express provides a Release Builder tool.

Exporting pages or page hierarchies by using the Manage Pages portlet:

You can do an XML export of a page or an entire page hierarchy by using the Manage Pages portlet.

Procedure

1. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
2. Locate the portal page that you want to export.
3. Click the **Export** button for that page.
4. The portlet prompts you to indicate whether you want to export the entire page hierarchy or only the selected page. Select one of the following options:
   - Click **Yes** to export the entire page hierarchy.
   - Click **No** to export only the selected page.
   - Click **Cancel** to stop the page export.
5. If you selected Yes or No in the previous step. The portlet prompts you to save the XML configuration file. Type a file name and select the location where you want the file to be saved.
6. When the export is complete, the portlet shows a success message.
7. On the Download complete screen, click **Open** to view the newly saved XML configuration file. Check the content of the file and make sure that it does not contain a `<failure>` tag. If you do not want to view the XML file, close the extra window that is open for viewing the file.

Importing pages or page hierarchies by using the XML Import portlet:

You can import an XML configuration file by using the XML Import portlet.

About this task

To import an XML configuration file by using the Import XML portlet, proceed as follows:

Procedure

1. Click the **Administration menu** icon. Then, click **Portal Settings** > **Import XML**.
2. Locate the XML configuration file that you want to import. Use the **Browse** button to help locate the file.
3. Click **Import** to import the configuration file.
4. When the import is complete, the portlet shows a success message.

Using the XML configuration command line client:

You access the XML configuration interface using a command line tool. This command line client is a separate program that connects to the server. Connecting to a **remote** server makes it possible to configure the portal remotely. You use the command line syntax of the XML configuration interface.

About this task

The remote connection can be either an HTTP connection, which is not secure, or a secure HTTPS connection. Apply care and use the appropriate type of connection that is required for your environment. Use an HTTP connection to connect to the XML configuration interface only from inside a protected intranet where you can be sure that the HTTP connection is not compromised. In all other networks use a secure HTTP**S** connection to connect to the XML configuration interface. For details about how to use an HTTPS connection refer to XML Syntax for using a secure connection with SSL.

You invoke the command line client by using the following shell scripts in the directory *wp_profile_root*/PortalServer/bin :

- Linux: ./xmlaccess.sh
- IBM i: xmlaccess.sh
- Windows: xmlaccess.bat

You can also use the XML configuration interface remotely from a machine that does not have portal installed. In this case copy the required files to the remote machine and configure the portal from there. All you need is a Java run time. However, you have to adapt the path settings in the shell scripts accordingly. These are the required files:

- *PortalServer_root*/base/wp.xml.client/bin/wp.xml.client.jar
- *PortalServer_root*/base/wp.base/shared/app/wp.base.jar
- *PortalServer_root*/base/wp.base/shared/app/wp.engine.impl.jar
- *PortalServer_root*/base/wp.base/shared/app/wp.utilities.streams.jar
- *AppServer_root*/lib/j2ee.jar
- *AppServer_root*/lib/bootstrap.jar
- *AppServer_root*/lib/com.ibm.ws.runtime.jar
- *AppServer_root*/plugins/j2ee.jar
- *AppServer_root*/plugins/com.ibm.wps.emf.jar
- *AppServer_root*/plugins/org.eclipse.emf.ecore.jar
- *AppServer_root*/plugins/org.eclipse.emf.common.jar
- *PortalServer_root*/bin/xmlaccess.sh or *PortalServer_root*\bin\xmlaccess.bat , depending on your operating system.

Note: When you update your portal by installing fix packs, these files might be updated. In such cases make sure that you always use the most recent versions of these files.

"Command line syntax of the XML configuration interface"
The command line syntax for working with the XML configuration interface can vary, depending on your portal installation environment and configuration.

"Transferring portal configuration data by using the XML configuration interface" on page 1072
When you use the XML configuration interface to transfer WebSphere Portal Express configuration data, you export or import an XML script file. In most cases, you can use the result file from an XML export for an XML import. Sometimes you can use the export result file directly, sometimes you must modify it.

"Creating and modifying resources" on page 1077
In addition to copying and restoring configurations of existing portal resources, you can use the XML configuration interface to install new resources in the portal. You can also use the XML configuration interface as an alternative to the portal administrative user interface for running some administration tasks.

"Activating and deactivating portlets, portlet applications, and web applications" on page 1079
You can change the states of portlets, portlet applications, and web applications between active and inactive by using the portal XML configuration interface.

"Scheduling the delayed cleanup of portal pages" on page 1079
You can use the example XML script `Task.xml` to schedule the cleanup of pages that have been marked for deletion.

"Registering predeployed portlets" on page 1079
You can manually predeploy portlet application WAR files using the WebSphere Integrated Solutions Console. You can later register and configure the predeployed portlet applications into WebSphere Portal Express, together with other J2EE resources and artifacts, by using the XML configuration interface.

"Removing users and groups" on page 1080
Depending on circumstances, you might want to remove users or groups from your WebSphere Portal Express that are no longer used or required. You can use the XML configuration interface (XML Access) to list such users and groups. You can also remove only some selected users and groups, and keep others for further use.

"Preparing the deletion of orphaned resources" on page 1081
You can use the XML configuration interface to delete orphaned data from your WebSphere Portal Express.

Related reference:

"XML Syntax for using a secure connection with SSL" on page 1068
You can also use the XML command line client with SSL over a secure HTTPS connection.

Command line syntax of the XML configuration interface:

The command line syntax for working with the XML configuration interface can vary, depending on your portal installation environment and configuration.

"Basic XML command line syntax" on page 1066
The basic command line syntax for the XML configuration interface is as follows.

"XML Syntax for using a secure connection with SSL" on page 1068
You can also use the XML command line client with SSL over a secure HTTPS connection.

"XML Syntax for exporting and importing credential vault data" on page 1070
When you use the XML command line for credential export or import, the command syntax is slightly different than for normal command-line use.

Basic XML command line syntax:

The basic command line syntax for the XML configuration interface is as follows.

```
xmlaccess -user user_ID -password password
          -url myhost:10039/wps/config
          -in input_file.xml -out result_file.xml
```

Note: All data, including the user ID and password, are sent to the server unencrypted. Therefore you should only connect to the XML configuration interface from inside a protected intranet where you can be sure that the HTTP connection is not compromised. In all other networks use a secure HTTPS connection to connect to the XML configuration interface.

Prompting for credentials: You can use the parameter `askForCredential` and leave out the parameters `user` and `password`. The XML configuration interface will then prompt you for the user ID and password. This can be useful in security sensitive environments, as the user credentials are not visible on the console or in the process view. The parameter `askForCredential` requires no value to be specified. Example:

```
xmlaccess -askForCredential -url myhost:10039/wps/config
          -in input_file.xml -out result_file.xml
```

Placing the credentials in a properties file: You can also place the credentials in a properties file and use the option `useEncryptedCredentials`. This option reads the encrypted or unencrypted credentials from the properties file, and then saves the file back using the encrypted password. If you do not want to write the properties file back with the encrypted credentials, use the **additional** flag `noUpdateProperties`. In this case you can use the `PropFilePasswordEncoder` utility to encrypt the password in the properties file. This option reads the following properties out of the file:

- For the user ID: `com.ibm.SOAP.loginUserid = userID`
- for the password: `com.ibm.SOAP.loginPassword = password`

An example of a command line is as follows

```
xmlaccess -in Export.xml -useEncryptedCredentials myProperties.properties
                         -url portal.example.com:10039/wps/config
```

Virtual portals: If you have virtual portals in your configuration, you can access a virtual portal by its host name or its URL mapping context. Example for accessing a virtual portal by its URL mapping context:

```
xmlaccess -user user_ID -password password
          -url myhost:10039/wps/config/URL_mapping_context_of_the_VP
          -in input_file.xml -out result_file.xml
```

Example for accessing a virtual portal by its host name:

```
xmlaccess -user user_ID -password password
          -url my_VP_host:10039/wps/config
          -in input_file.xml -out result_file.xml
```

"Syntax elements for the XML configuration interface command line"
This topic lists the syntax elements for using the XML configuration interface command line client over an HTTP connection.

Syntax elements for the XML configuration interface command line:

This topic lists the syntax elements for using the XML configuration interface command line client over an HTTP connection.

For information about the XML syntax elements for a secure HTTPS connection see the topic about *XML Syntax elements for using a secure connection with SSL*.

Table 139. Descriptions for the Syntax elements

Syntax element	Description
`xmlaccess`	This is the shell script. It is located in directory *`wp_profile_root`*`/PortalServer/bin` . Use one of the following scripts: • Linux: `./xmlaccess.sh` • IBM i: `xmlaccess.sh` • Windows: `xmlaccess.bat`
`-in`	Use this element to specify the name of a file containing the XML request (configuration export or update) that should be processed.
`-user` and `-password`	Use these elements to specify the user identification and password describing the authority under which the request should be processed. For the value for `user` you must specify the short user name as specified during login; full distinguished names (DN) are not supported. The XML configuration interface is only accessible to users that have the manager role on the virtual resource XML_ACCESS and the administrator role on the virtual resource PORTAL.
`-askForCredential`	You can use the parameter `askForCredential` and leave out the parameters `user` and `password`. The XML configuration interface will then prompt you for the user ID and password. The parameter `askForCredential` requires no value to be specified.
`-useEncryptedCredentials`	Use this option if you want to provide the user credentials in a properties file rather than with the XML command.
`-noUpdateProperties`	Use this option additionally with the option `useEncryptedCredentials`, if you do not want to have the encrypted credentials written back to the properties file.
`-url`	Use this element to specify the URL to access the configuration servlet. This URL consists of the host name, the base URI as specified during installation (for example `/wps`), and the servlet extension `/config`.
`-out`	The name of the result file that contains the XML output. This file gives a result status and thereby indicates whether the XML request was performed successfully, or what errors might have occurred. In the case of an XML export, this file contains the exported configuration. You can later use this file to re-import the exported configuration.

XML Syntax for using a secure connection with SSL:

You can also use the XML command line client with SSL over a secure HTTPS connection.

In this case the command syntax is as follows:

```
xmlaccess -user user_ID -password password
          -url https://myhost:10035/wps/config/
          -in input_file.xml -out result_file.xml
          -truststore trustStore -trustpwd trustPassword
          -trusttype trustType [ -keystore keyStore
          -keypwd keyPassword -keytype keyType ]
```

The following rules apply:

1. The `https://` prefix in the URL is required to allow the XML client to detect whether a secure HTTPS connection is required. The appropriate HTTPS port must be provided instead of the HTTP port.
2. The options starting with the string `trust` are mandatory in all configurations where a custom certificate store is used for storing certificates required for secure connections. For configurations that use the Java standard `cacerts` certificate store, the parameters starting with `trust` are optional.
3. The options starting with the string `key` are optional. They are only required when client certificate authentication is used for establishing the SSL connection.
4. The default value for `-keytype` and `-trusttype` is `jks`. Therefore the `-keytype` and `-trusttype` options are optional unless the used keystore or truststore uses a different format.

Note: When your WebSphere Portal Express runs on the Oracle Solaris platform, the default protocol handler for the hybrid IBM JDK is the Sun handler. Therefore, in order to successfully connect by using the XML configuration interface and the IBM JSSE2 provider, you need to add an additional parameter to the file *wp_profile_root*`/PortalServer/bin/xmlaccess.xml` . Edit that file and add the parameter `-Djava.protocol.handler.pkgs=com.ibm.net.ssl.www2.protocol` as follows:

```
   . . . . .
${JAVA}
-Djava.protocol.handler.pkgs=com.ibm.net.ssl.www2.protocol
-classpath ${WPS_HOME}/. . . . .
   . . . . .
```

"XML syntax elements for using a secure connection with SSL"
This topic lists the syntax elements for using the XML command line client with SSL over a secure HTTPS connection.

XML syntax elements for using a secure connection with SSL:

This topic lists the syntax elements for using the XML command line client with SSL over a secure HTTPS connection.

Table 140. Description of the Syntax elements used for a secure connection with SSL

Syntax element	Description
`-truststore`	Use this element to specify the name of the truststore file that contains the server certificates that are required for accepting SSL connections with trusted servers. If no truststore is provided, the XML client will use the default Java `cacerts` truststore.
`-trustpwd`	Use this element to specify the password that is required for accessing the truststore. If the default Java `cacerts` truststore is used, no trust password needs to be provided.
`-trusttype`	Use this element to specify the type of the truststore that is used. The default type is `jks`. As long as the used truststore is of type `jks`, you do not have to provide this parameter.
`-keystore`	Use this element to specify the name of the keystore file that contains client certificates that are required for establishing an SSL connection with a server that requires client certificate authentication. If no keystore is provided, the XML client will use the default Java `cacerts` keystore.
`-keypwd`	Use this element to specify the password that is required for accessing the keystore. If the default Java `cacerts` keystore is used, no key password needs to be provided.
`-keytype`	Use this element to specify the type of the used keystore. The default type is `jks`. If the used keystore is of type `jks`, you do not have to provide this parameter.
`-protocol`	Use this element to specify the protocol, for example `SSL`, `SSLv1`, `SSLv3`, or `TLS` . Note that you can select only protocols that WebSphere Application Server supports and has enabled. The parameter is evaluated only if the URL of the XMLAccess servlet selects a secure connection with HTTP**s**. Otherwise, the parameter is ignored.

See the following examples.

Example 1

The following is an example of how to use the XML configuration interface to establish an SSL connection with a WebSphere Portal Express server, using the default certificate stores that are provided by WebSphere Application Server:

```
xmlaccess.sh -user wpsadmin -password your_password -url https://portalhost:10035/wps/config/
             -in $PortalHome/doc/xml-samples/ExportAllUsers.xml -out result.xml
             -truststore $WASHome/profiles/wp_profile/etc/trust.p12
             -trustpwd WebAS -trusttype PKCS12
```

For this example to run, use the `trusttype` parameter with a value of `PKCS12` to avoid an `invalid file format` error.

Example 2

The following is an example of how to use the XML configuration interface to establish an SSL connection with a WebSphere Portal Express server, using the dummy certificate stores that are provided by WebSphere Application Server:

```
xmlaccess.sh -user wpsadmin -password your_password -url https://portalhost:10035/wps/config/
             -in $PortalHome/doc/xml-samples/ExportAllUsers.xml -out result.xml
             -truststore $WASHome/profiles/wp_profile/etc/DummyClientTrustFile.jks -trustpwd WebAS
```

For this example to be able to run, you need to configure the SSL configuration in WebSphere Application Server using the `DummyServerKeyFile.jks` and the `DummyServerTrustFile.jks` for secure connections. The option **require client authentication** must not be active.

Example 3

If the option **require client authentication** is active, you need to provide a keyfile when establishing the SSL connection with the XML configuration interface:

```
xmlaccess.sh -user wpsadmin -password yourpassword -url https://portalhost:10035/wps/config/
             -in $PortalHome/doc/xml-samples/ExportAllUsers.xml -out result.xml
             -truststore $WASHome/profiles/wp_profile/etc/DummyClientTrustFile.jks -trustpwd WebAS
             -keystore $WASHome/profiles/wp_profile/etc/DummyClientKeyFile.jks -keypwd WebAS
```

This example allows the XML configuration interface to send a client certificate to the server, if the server requests one. Using client certificate authentication is required wherever the number of clients that can administer WebSphere Portal Express needs to be controlled. Only clients with the correct client certificate will be able establish a connection with WebSphere Portal Express.

XML Syntax for exporting and importing credential vault data:

When you use the XML command line for credential export or import, the command syntax is slightly different than for normal command-line use.

Prerequisite configuration: Before you run the `xmlaccess` command to export or import credential vault data, make sure that you added the two properties `export.userDN` and `export.enforceSSL` to the WebSphere Application Server configuration.

When you use the XML command line for credential export or import, you need to add two more parameters: `-credentialexport` and `-passphrase` to the XML command. See the following example:

```
xmlaccess -user user_ID -password password
   -url https://myhost:10035/wps/config/
   -truststore
    wp_profile_root/config/cells/cellname/nodes/nodename/trust.p12
   -trusttype PKCS12    -trustpwd WebAS
   -in input_file.xml -out result_file.xml
   -credentialexport -passphrase encryptionPassphrase
```

Table 141. Additional XML Syntax elements for credential secret migration

Syntax element	Description
`-credentialexport`	This parameter, without a value, indicates that the export of credentials must be enabled.

Table 141. Additional XML Syntax elements for credential secret migration (continued)

Syntax element	Description
`-passphrase`	Use this element to specify the `encryptionPassPhrase` for the encryption. The minimum length of this string is the number of bits set as the export `keylength` in the WP Vault Service Custom properties, which are divided by 8. The `-passphrase` value is used to create a key of the specified length for the encryption. For details about the WP Credential Vault Service, see the topic about the Credential Vault Service. For details about how to configure or determine service configuration properties see the topic about Setting service configuration properties.

Usage notes:

- The following rules apply to these parameters:
  - For export or import of encrypted credential secrets, the options `credentialexport` and `passphrase` are mandatory. For example, during migration you need to specify these options.
  - For all XML Configuration actions that do not export or import encrypted credential secrets during migration, the options `credentialexport` and `passphrase` are optional.
- Use the same `passphrase` for both the export and the import.
- The import might fail if the user DN schema was changed between the previous and the current system or when credentials for users are contained in the XML import file that is not present in the current system. In this case, manually remove the obsolete credential entries from the XML file, then complete the import.
- For security reasons, use an HTTPS connection when you import credentials; however, if you choose not to, set the `export.enforceSSL` configuration property to `false`.

Example

Following is an example of how to use the XML configuration interface to export/import credential secrets by using HTTPS:

```
xmlaccess.sh -user wpsadmin -password your_password -url https://portalhost:10035/wps/config/
     -in ExportedCredentialSecrets.xml -out result.xml
     -credentialexport -passphrase JGD786JHgasdf8a67kjhUIT7sdj7nsh776jasdf786regUFZT756675zufurz
     -truststore $WASHome/profiles/wp_profile/etc/DummyClientTrustFile.jks -trustpwd WebAS
```

"Adding `export.userDN` and `export.enforceSSL` to the WebSphere Application Server configuration" on page 1072
Before running the `xmlaccess` command to export or import credential vault data, you need to add two properties to the WebSphere Application Server configuration.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

"Adding export.userDN and export.enforceSSL to the WebSphere Application Server configuration"
Before running the xmlaccess command to export or import credential vault data, you need to add two properties to the WebSphere Application Server configuration.

Related reference:

"Credential Vault Service" on page 332
You can use the portal Credential Vault Service to configure Vault Adapter implementations that are used by the Credential Vault Service to store credential secrets.

Adding export.userDN and export.enforceSSL to the WebSphere Application Server configuration:

Before running the xmlaccess command to export or import credential vault data, you need to add two properties to the WebSphere Application Server configuration.

About this task

Proceed as follows:

Procedure

1. Log in to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment** > **Providers** > **WP_VaultService** > **Custom properties**.
3. Add the property export.userDN :
   Name: export.userDN
   Value: *administrator_DN*. For example: cn=wpsadmin,o=ibm
   Type: java.lang.String
4. Add the property export.enforceSSL :
   Name: export.enforceSSL
   Value: true
   Type: java.lang.Boolean
5. Save your configuration changes.
6. Restart the portal server.

Transferring portal configuration data by using the XML configuration interface:

When you use the XML configuration interface to transfer WebSphere Portal Express configuration data, you export or import an XML script file. In most cases, you can use the result file from an XML export for an XML import. Sometimes you can use the export result file directly, sometimes you must modify it.

Before you begin

An XML file that you process must always be in UTF-8 encoding. It must specify the root element and schema that is given in the following example code snippet.

```
<?xml version="1.0" encoding="UTF-8"?>
<request xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
```

```
        type="export|update">
      . . . configuration  . . .
    </portal>
  </request>
```

For an XML export, specify `export` for the request type. For an XML import, specify a request type of `update`. The line where you specify either of these request types is highlighted in the example. For more information about the structure of an XML input script file, see the reference topics about the *XML input script file structure*.

Procedure

1. Run the XML command-line interface with a file that has a request type of `export` in it. For example, you can use one of the XML sample files with request type `export` provided with WebSphere Portal Express. The XML command-line interface returns a result file that contains the resources that are specified in the XML file that you used for the export. This file can be, for example a resource and all dependent resources. The file that XML command-line interface returns specifies `update` for the request type and `locate` or `update` for the individual resources actions. This file is ready to be used for an XML import.
2. Optional: Modify the XML result file from the export as required. For example, to create extra resources, use the actions `create` or `update`.
3. Run the XML command-line interface, and specify the XML file that resulted from the XML export and that you might have modified in the previous steps. You can also use one of the XML sample files with request type `update` in it. The XML command-line interface returns a result file that indicates whether the specified resources were imported successfully, or what errors might have occurred.

What to do next

Usage notes:

Using the XML output result file for further processing:
When the XML request finishes processing on the server, the resulting XML output is sent back to the client and written to the standard output. You can write the output to an XML file by using the `-out` command-line option. Using this option always writes the output in UTF-8 encoding, so you can usually use that file for further XML processing. If you do not use this option, the output is written in a console encoding that depends on your operating system and active locale. It might therefore be invalid XML. For more information, see the topics about using the XML command-line client.

Usage notes on the difference between XML exports and imports:

- The command-line syntax and XML processing is the same for both exports and imports. You specify an XML input file to the XML configuration interface, and the XML configuration interface returns a resulting XML export file.
- The difference between export and import is determined by whether you set the request type to `export` or `update` in the XML input file that you specify in the command-line request. When you run an XML import, the resources `action` attribute can have the following values: `locate`, `create`

or `update`. For more information about XML resources, elements, and attributes, see the topics about the *XML configuration reference* and *Types of portal resources*.

For more information about XML exports and imports and transfers of portal configuration, see the following topics. For more information about the structure of an XML input script file, see the reference topics about the XML configuration interface.

"Transferring a complete configuration"
For transferring complete portal configurations, WebSphere Portal Express provides a Release Builder tool.

"Exporting and transferring parts of a portal configuration" on page 1075
You can also export partial configurations.

Related tasks:

"Using the XML configuration command line client" on page 1064
You access the XML configuration interface using a command line tool. This command line client is a separate program that connects to the server. Connecting to a **remote** server makes it possible to configure the portal remotely. You use the command line syntax of the XML configuration interface.

Related reference:

"XML input script file structure" on page 1084
When you use the XML configuration interface command line client, the XML script you use specifies the root element, the XML schema, the portal resources, and actions to be performed.

"Types of portal resources" on page 1085
The portal resources are represented by the following XML tags.

"Sample XML configuration files" on page 1111
Sample files are provided for your reference to help illustrate how to use XML configuration for different portal configuration purposes. Before you use them, read the other topics about the XML configuration interface carefully. Many of the samples need to be modified with valid page or user name before they can be used.

Transferring a complete configuration:

For transferring complete portal configurations, WebSphere Portal Express provides a Release Builder tool.

About this task

To transfer a complete portal configuration, use the WebSphere Portal Express ReleaseBuilder. For information about how to use it see the topics about ReleaseBuilder.

If you want to move a complete configuration from a test to a production server by using the portal ReleaseBuilder tool, use the XML sample file `ExportRelease.xml` provided with the portal . The attribute `domain="rel"` indicates that only shared and no private resources are exported. This sample file exports the complete portal configuration without private resources as required by the portal ReleaseBuilder tool.

Related concepts:

"ReleaseBuilder" on page 3624
To generate or stage follow-on releases of IBM WebSphere Portal Express portals, configurations, and artifacts need to be moved between systems. ReleaseBuilder

enables management of release configurations independent of user configurations.

Related reference:

"Sample XML configuration files" on page 1111
Sample files are provided for your reference to help illustrate how to use XML configuration for different portal configuration purposes. Before you use them, read the other topics about the XML configuration interface carefully. Many of the samples need to be modified with valid page or user name before they can be used.

Exporting and transferring parts of a portal configuration:

You can also export partial configurations.

About this task

To complete this step, you specify the XML hierarchy down to the specific portal resource that you want to export. For the portal resource element itself, you specify an `export` action; for its parents, you specify a `locate` action.

The sample XML request file `ExportPage.xml` provided with the portal exports a page with the unique name `ibm.portal.ssa.SamplePage`. This exported page does not exist in a newly installed portal. You can create it by running the `DeployPortlet.xml` sample file, which is covered in the next section.

Normally, you specify the resources that you want to export by their object ID or by their unique name. You can use the Custom Unique Names administration portlet to look up object IDs and unique names of portal resources. Some resources also support lookup by other attributes; see the XML reference documentation for detailed information.

Running the `ExportPage.xml` example request file that is mentioned earlier results in an XML file similar to `ExportPageResult.xml`. You can use this file to update the page to the exported state, if it still exists in the portal. You can also use this file to re-create the page, in case you delete it later.

When you look at the file, you notice that it includes not only the page itself but also other configuration elements that are referred to by the page, for example the portlet that is placed on the page. These other elements have a `locate` action. The export does not include their full configuration data, but enough information to look them up in the portal, assuming they exist. Note how the configuration of the page makes references to the `objectid` attributes of other resources, for example in the `portletref` attribute of the `portletinstance` elements.

All those references are described by object IDs. Therefore, if the object IDs are correct, the referenced resources can be looked up in the portal even if they were not included in the export. Locating resources before they are referenced is only necessary if you do not know their actual object IDs so that you must find the resources by some other identifying attribute. For more information, see the XML reference documentation. That way, for example a portlet can be identified by its name and by the `uid` attributes of its parents, and the referencing still works, even if the object ID is not available for looking up the portlet.

Exporting resource configurations normally creates `update` actions for all exported elements. This means that if the portal resource exists on the importing system, the

settings are modified, and if it does not yet exist, it is created. This in turn means that if you reimport the page into the portal that you exported it from, nothing changes.

You can import the XML file into another portal to create a copy of the page, this importation requires that the referenced resources (such as the portlet and the content parents) also exist on the target portal and can be found by an identifying attribute. In that case, the page and all contained resources take their object IDs with them so that they have the same object IDs on the source and target system - the resources retain their identity. You can avoid that by using the ID generating mode. For more information, see the XML reference documentation. When you use the ID generating mode, the object IDs in the input are not taken literally, but during the import process the resources obtain new object IDs when they are created on the target system. You apply ID generating mode by adding the following attribute to the main request tag:

```
<request . . . create-oids="true" . . . >
```

You can create a duplicate of the page in the portal from where you exported it by using the ID generating mode and changing the unique name of the page in the XML script. This way, the page, and its changed name, cannot be found for updating by either its object ID or its unique name, therefore a new page with the same settings is created. Change the page title so that you can distinguish between the two pages. The `CopyPage.xml` sample shows how this script would look.

When you are exporting resources to XML scripts, it is possible and often useful to export several resources by using one request. The `ExportPortletAndPage.xml` example extends the `ExportPage.xml` example by including also the portlet that is contained on the page. The resulting XML file contains the complete configuration data of the portlet and the page.

The `ExportSubTree.xml` example shows how you export subtrees of the portal content hierarchy. It exports part of the predefined administration page hierarchy that was created during the portal installation.

Using wildcard characters:

When you export portal resources, you can specify the asterisk (*) as a wildcard character for tag attributes. Be aware of the following limitations:

1. The asterisk wildcard character is supported for attributes of top-level tags only, that is, subtags of the `portal` tag.
2. Specify the asterisk wild character for the object ID attribute of tags as follows: `objectid='*'`, except for `policy-node` tags, where you can specify it for the `path` attribute.
3. Specify only the asterisk alone: **"*"**. The asterisk does not work in combination with partial strings that precede or follow it. For example, you cannot specify `"abc*"` or `"*xyz"`.
4. If you specify the asterisk as a wildcard character, all other attributes of that tag are ignored, except for the following tags, where the listed attributes are interpreted as filters:
   - The tag `content-node`, attribute `create-type`
   - The tag `tag`, attribute `locale`.

The `ExportAllPortlets.xml` example shows the use of the asterisk character (*) as a wildcard to export all resources of a given type. This example exports all the web modules that were installed in the portal and their contained portlets.

Related reference:

"XML configuration reference" on page 1082
Learn more about XML input structure, XML tags for portal resources, attributes for special purposes such as locale data, object IDs, and more. Also find information about action attributes that determine the type of processing that the XML configuration reference applies to the portal resource. There are syntax restrictions that you need to consider as well as information about how to determine the object IDs for portal resources.

"Sample XML configuration files" on page 1111
Sample files are provided for your reference to help illustrate how to use XML configuration for different portal configuration purposes. Before you use them, read the other topics about the XML configuration interface carefully. Many of the samples need to be modified with valid page or user name before they can be used.

Creating and modifying resources:

In addition to copying and restoring configurations of existing portal resources, you can use the XML configuration interface to install new resources in the portal. You can also use the XML configuration interface as an alternative to the portal administrative user interface for running some administration tasks.

About this task

In these cases, you cannot export and reimport XML scripts, but you must edit them. In most cases, it is still useful to start with an XML export and only partially modify it, rather than writing complete new XML scripts. The following scripts show examples for modifying different resources in the portal configuration by using XML scripts.

All the examples use the ID generating mode and do not specify literal object IDs. Therefore, you can run them on any portal installation, as they do not depend on hardcoded object ID values. As noted earlier, using literal object IDs makes sense only if you really want to create two instances of the *same* resource, and if you have a controlled environment where you can guarantee that all object IDs that your resources depend on have exactly the required values. As object IDs are difficult to use for identifying the resources, the examples assign unique names to most top-level resources. This way you can reference them later, and the resources are not duplicated if you run the scripts twice.

The first example file `DeployPortlet.xml` shows how you deploy a portlet and create a simple test page to display the portlet. Some of the attributes in the XML must match the corresponding settings that were defined in the `portlet.xml` deployment descriptor in the portlet WAR file. This is necessary so that the XML processing can properly identify the contents of the WAR file. When you want to deploy a different portlet, you must not only specify a different WAR file but also adapt those attributes. Also, the configuration that is specified for the portlet is less than what you see in an XML export result for the portlet. For example, the localized titles are not included in the XML script. This is because those settings are specified in the `portlet.xml` deployment descriptor; you do not have to override them with the XML configuration interface.

Note: As Windows limits the maximum path length to 260 characters, the name of the WAR file must be 25 characters or less. Deploying a WAR file with a name that is more than 25 characters results in an error.

The CreatePage.xml sample shows the following extra possibilities:

1. It assumes that the portlet is already installed. Therefore, it uses only a locate action for the web module, not an update action.
2. It sets a specific skin for displaying the portlet on the page.
3. It shows how you can specify localized titles in properties files rather than include them in the XML script: the titles and descriptions for the page are now loaded from two properties files for two different languages.

Both examples use a simple page layout with just one row and one column. If you want to generate more complex page layouts, you can use the administration portlets to create them. You can export the result to generate a template for your XML scripts.

When you create new resources, you might want to define specific access control settings for them, for example to make them visible to all portal users. The UpdateAccesscontrol.xml example shows the syntax for specifying different access control settings. This sample updates existing resources, but you can use the same syntax to define access control settings for new resources while you are creating them in an XML script. This sample also shows how you can specify access control user roles on virtual resources. This allows you to give a user access to all resources of a specific type that exist in the portal.

The CreateURL.xml sample defines a URL mapping for the sample page that was created with the DeployPortlet.xml example mentioned earlier. After you create the URL mapping, you can access the page directly by entering that URL in the browser.

The DeployTheme.xml example shows how you can use XML scripts to install new themes and skins into your portal. The XML scripts create these resources only in the portal database, so that they can be used in the portal. In addition, you must write the JSPs that run the actual visualization and copy them to the resource directory specified in the XML before you can use the theme in the portal.

The ModifyPortlet.xml example changes settings of a portlet instance that is shown on a page. Such settings are normally set in the edit mode of the portlet. It depends on the code of the portlet which settings are stored and how they are used.

The CreateUser.xml example imports a new user into the portal. It also creates a group that contains only that one user.

To add a language to the portal, use the CreateLanguage.xml example.

Note: To prepare for running this XML script, you must insert resource bundles and, where applicable, JSPs for the new language. For details about how to do this, refer to the topic about how to support new languages in the WebSphere Portal Express information center.

The UpdateVault.xml example demonstrates how to create new resources in the portal credential vault with an XML script.

The ClonePortlet.xml example shows how you can use the XML configuration interface to add new portlets with different settings to existing applications.

The `Transaction.xml` example demonstrates the effect of using different transaction levels for the execution of an XML import.

The `MovePage.xml` example shows you how to move a page to another node.

Note: The actual move of the page is done by the last two lines in the sample file.

Related tasks:

"Supporting a new language" on page 1421
To support a new language to IBM WebSphere Portal Express you add resource bundles and, where applicable, JSPs for the new language.

Activating and deactivating portlets, portlet applications, and web applications:

You can change the states of portlets, portlet applications, and web applications between active and inactive by using the portal XML configuration interface.

About this task

The `ActivatePortlet.xml` example shows you how to complete this step.

Scheduling the delayed cleanup of portal pages:

You can use the example XML script `Task.xml` to schedule the cleanup of pages that have been marked for deletion.

About this task

Notes:

1. If you delete a page with an object ID and then use the XML configuration interface to re-create the same page with the same object ID, you might receive an error message indicating the operation was canceled because it would have caused a duplicate key value.
2. When you run the cleanup task, the XML configuration interface only schedules the task to be run in WebSphere Application Server and returns. This does not necessarily mean that WebSphere Application Server runs the task immediately. To determine when a task started and ended, check the portal log `SystemOut.log` for the EJPDE0005I and JPDE0006I messages. These messages confirm that the cleanup task has successfully completed. After you have confirmed this, you can run the XML script for re-creating a page with the same object ID as it had before the deletion.

Registering predeployed portlets:

You can manually predeploy portlet application WAR files using the WebSphere Integrated Solutions Console. You can later register and configure the predeployed portlet applications into WebSphere Portal Express, together with other J2EE resources and artifacts, by using the XML configuration interface.

About this task

To install a predeployed portlet, use the sample file `RegisterPreDeployedEAR.xml`. You might have to change this sample for your requirements. For more information about how to predeploy portlets refer to the topic about *Deploying J2EE resources*.

Removing users and groups:

Depending on circumstances, you might want to remove users or groups from your WebSphere Portal Express that are no longer used or required. You can use the XML configuration interface (XML Access) to list such users and groups. You can also remove only some selected users and groups, and keep others for further use.

About this task

IBM WebSphere Portal Express stores users and groups that exist in the user registry as entries in the database. When you use the XML configuration interface or the **Manage User and Groups** portlet to delete users and groups, they are deleted from both the user registry and from the database. Deleting a user or group directly from the configured user registry does not remove the database entry. Also, WebSphere Portal Express does not remove entries from its database when users or groups are muted in the user registry, for example, users with too many wrong password attempts. You can manually remove the users and groups from the database.

Examples for removing users or groups can be the following cases:

- Portal users or groups were removed from the user registry, but not from the portal database.
- User IDs were deactivated, for example after too many wrong password attempts.

Note: After you delete these entries by using the modified XML script, all customization is lost for the deleted users and groups.
To remove users and groups from your portal, proceed as follows:

Procedure

1. Make a backup copy of your portal database.
2. To identify and list these users and groups, run an XML export and use the `cleanup-users` attribute.

   Specify the `cleanup-users` attribute with the `request` tag of type `export`, and set its value to `true`. You also need to set the `export-users` attribute to `true`.

   The resulting output file lists the affected users and groups with their action set to `delete`.

   The XML sample file `CleanupUsers.xml` shows an example of how you can export such users and groups. For information about the sample XML configuration files and their location, read *Sample XML configuration files*.

   Note: CF03 If the number of invalid users is very high, the XML export step can fail with an out-of-memory exception. For such cases, APAR PI23109 introduces a new XML element `threshold`. In case of such out-of-memory exceptions, add `threshold="10000"` to the `<request ... >` element in the `CleanupUsers.xml` script. This option limits the number of exported users to 10,000. When you use this approach, repeat the export step and all following steps until the exported file contains no entries any more. You need to have APAR PI23109 or fix pack CF03 installed to use this XML element.
3. Check the output file from the previous step and **remove** all users and groups that you want to **keep** in the portal database. For example, you might want to keep the muted users and re-enable their passwords. All users and groups that remain in the file are removed from the database in the following import step.

4. Import the modified XML file into your portal. The portal removes all users and groups that you retained in the XML file during the previous step from the portal database.

Results

After you delete these entries by using the modified XML script, all customization is lost for the deleted users and groups.

Related reference:

"Sample XML configuration files" on page 1111
Sample files are provided for your reference to help illustrate how to use XML configuration for different portal configuration purposes. Before you use them, read the other topics about the XML configuration interface carefully. Many of the samples need to be modified with valid page or user name before they can be used.

Preparing the deletion of orphaned resources:

You can use the XML configuration interface to delete orphaned data from your WebSphere Portal Express.

Procedure

1. To prepare for deleting orphaned data, use the example XML script `ExportIncludingOrphanedData.xml`. This script runs an export that includes all orphaned data. The resulting XML output file lists the affected portal resources with their action set to `delete`.
2. Check the output file from the previous step and remove all resources that you want to keep in the portal.
3. Import the modified XML file into your portal. The portal removes all resources that you retained in the XML file during the previous step.

Related tasks:

"Deleting orphaned data" on page 280
You use the SLCheckerTool to delete orphaned data in the database.

Generating a complete XMLAccess export of a Portal configuration:

To help troubleshooting a WebSphere Portal issue, you might be asked by IBM Support to generate or collect a full or complete export of your Portal configuration. The complete export can be generated by using the XML Configuration Interface, commonly known as XMLAccess.

To generate a full export of your Portal's configuration, do the following steps:

1. Open a terminal or command window. Change directory (cd) to `<WP_Profile_root>/PortalServer/bin`
2. Run the following command all on one line:

   ```
   xmlaccess -user Portal_admin_user -password Portal_admin_password -url
   http://<myhost>:<port>/wps/config -in <Portal home>/doc/xml
   -samples/Export.xml -out result.xml
   ```

   - Substitute `Portal_admin_user`, `Portal_admin_password`, `myhost`, and `port` with the correct values for your environment. It is recommended to use the direct URL and port so the request can bypass any webserver or load balancer that might be present.

- The file name that is specified after the `-out` parameter contains the Portal configuration as XML. The output file can have any name.
- The protocol can be omitted after the -url in the value for the parameter except in the case of SSL.

If you receive an error that any one of the variables JAVA, WPS_HOME or WAS_HOME are not defined run the script `<Appserver_home>/bin/setupCmdLine` in the current window.

You might also start XMLAccess without any parameters on the command line to test it in your environment.

If the file `Export.xml` is missing from your installation, contact IBM Support or take a copy from another Portal host.

Note: The file `Export.xml` is used as the input file in these examples to generate an export for diagnostic purposes. However, when Portal artifacts are to be imported to or updated in a Portal configuration, or for use by ReleaseBuilder the export file is generated by using `ExportRelease.xml` as the input file rather than `Export.xml`.

Virtual Portals

XMLAccess exports the base or default Portal for the specified host by using the `/config` URI. If virtual portals are defined, each must be exported separately. The virtual portal URL mapping context must be specified in the URL.

For example, given a Portal 7.0 virtual portal VP1 an example XMLAccess command to export the content of VP1 is:

```
xmlaccess -user Portal_admin_user -password Portal_admin_password -url
http;//host.raleigh.ibm.com:10039/wps0config/VP1 -in Export.xml -out
result_VP1.xml
```

If your Virtual Portal is defined by using an optional host name, specify the host name and `/config` URI to export the Virtual Portal content:

```
xmlaccess -user Portal_admin_user -password Portal_admin_password -url
http://MyVirtualPortalHost.com:10039/wps/config -in Export.xml -out
result_MyHost.xml
```

XML configuration reference

Learn more about XML input structure, XML tags for portal resources, attributes for special purposes such as locale data, object IDs, and more. Also find information about action attributes that determine the type of processing that the XML configuration reference applies to the portal resource. There are syntax restrictions that you need to consider as well as information about how to determine the object IDs for portal resources.

"XML input script file structure" on page 1084
When you use the XML configuration interface command line client, the XML script you use specifies the root element, the XML schema, the portal resources, and actions to be performed.

"Types of portal resources" on page 1085
The portal resources are represented by the following XML tags.

"Syntactic restrictions on the input syntax" on page 1099
There are certain restrictions on the allowed values for the `action` attribute.

"Hints and tips for using the portal XML configuration interface" on page 1110
In an example configuration, you might have two WebSphere Portal Express environments that are both configured for security with an LDAP server. However, the two LDAP servers have different directory structures.

"Sample XML configuration files" on page 1111
Sample files are provided for your reference to help illustrate how to use XML configuration for different portal configuration purposes. Before you use them, read the other topics about the XML configuration interface carefully. Many of the samples need to be modified with valid page or user name before they can be used.

XML input script file structure:

When you use the XML configuration interface command line client, the XML script you use specifies the root element, the XML schema, the portal resources, and actions to be performed.

The main level structure of an XML request or response is always as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
   <request
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
    type="export|update">
      <portal . . . >
      definition of configuration parts to be exported or updated
      </portal>
      <status . . . >
      success or failure indication for the processing
      </status>
   </request>
```

The main request element specifies the XML schema used by the XML configuration interface. You must always use the schema reference that is shown in the example, that is a reference with no namespace to the schema `PortalConfig_8.5.0.xsd`. All XML requests must conform to this schema. For your reference, you can find the schema declaration in the JAR file `wp.xml.jar` under the location `com/ibm/wps/command/xml/PortalConfig_8.5.0.xsd`. The JAR file `wp.xml.jar` is located under the following directory:

- **For Linux:** *PortalServer_root*`/base/wp.xml/shared/app`
- **For IBM i:** *PortalServer_root*`/base/wp.xml/shared/app`
- **For Windows:** *PortalServer_root*`\base\wp.xml\shared\app`

All other XML sample files are located in the following directory:

- **For Linux:** *PortalServer_root*`/doc/xml-samples`
- **For IBM i:** *PortalServer_root*`/doc/xml-samples`
- **For Windows:** *PortalServer_root*`\doc\xml-samples`

Before you send requests to the portal, you can verify them against this schema using a suitable editor or parser to ensure syntactic correctness. The schema also contains annotations that give detailed information on the meaning and possible values of all configuration entries.

The `type` attribute indicates whether the XML request contains specifications for exporting or for updating portal resources.

The `portal` section describes the parts of the portal configuration that should be exported or updated. The contents of the hierarchy used are described in more detail in the following sections.

The `status` section is optional; in an XML response it indicates success or failure of the requested operation. If a status element is present in a XML request, the server simply ignores it.

The simplest request that you can send to a server is the following:

```
<?xml version="1.0" encoding="UTF-8"?>
   <request xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
         type="export">
      <portal action="export"/>
   </request>
```

This request exports the entire configuration of the portal. You can look at the contents of the response to see how the configuration of individual portal resources, such as portlets or pages, is represented in XML elements and attributes.

Additional to the `export` and `update` request types, a third request type `export-orphaned-data` is available for the special scenario of preparing the deletion of orphaned data.

Types of portal resources:

The portal resources are represented by the following XML tags.

request
The main tag of every XML request. This element must always appear exactly once and must enclose the request. You can use this attribute to export release data from the portal database to later feed that data it into the staging process.

portal The main element of every XML request. This element must always appear exactly once and supports only `locate` and `export` actions.

global-settings, services-settings
Settings for global portlet configuration values and for specific portal services. These elements support only `locate` and `export` actions.

Note: Leave the XML scripts for exporting and updating these settings unchanged. Do not manually change the values in the XML scripts, as this action might result in invalid portal configurations. To modify these settings, use the appropriate administration portlets instead after the XML update of a configuration.

language
Use this tag to add languages to the list of supported languages that are defined in the portal or to delete languages from that list. Use the following attribute for bidirectional languages.

bidi For bidirectional languages, specify this attribute as `true`.

task Schedules the cleanup of portal resources and the administration of federated tags. *Cleanup* refers to the deletion of portal pages and all page-dependent resources that are marked for deletion. Specify one or more tasks to be run immediately or at specified intervals. The scheduling interval parameters **`dayOfMonth`** and **`dayOfWeek`** are optional and are

mutually exclusive. Each scheduling interval parameter requires a value for the parameter **startTime**. If you want to run a task daily, use only the parameter **startTime**.

name Specifies the scheduling task to be run:

com.ibm.portal.datastore.task.ResourceCleanup
This task cleans up portal resources. If you specify this task without a scheduling interval parameter, portal resource cleanup is done immediately when you run the XML script.

com.ibm.portal.cp.SynchronizationTask
This task synchronizes collaborative data, including tag categorization information that is provided by Web Content Manager tagging. If you specify this task without a scheduling interval parameter, the portal does a resource cleanup immediately when you run the XML script.

com.ibm.portal.services.RefreshIWidgetDefinitionsTask
This task refreshes iWidget definitions that are stored in WebSphere Portal Express. Refreshing iWidget definitions refers to reloading the iWidget definition XML files and updating the corresponding iWidget Wrapper portlet clones.

Tip: To call this task directly, run the following portal configuration task: `refresh-iwidget-definitions`. See the topic about the *Task refresh-iwidget-definitions* for instructions.

com.ibm.portal.cmis.TransientSlotCleanupTask
This task removes temporary credential vault slots that were created by the federated documents wizard and were not used for at least 3 hours.

com.ibm.wps.cp.tagging.federation.taskhandler.FederationTaskHandler
This task retrieves tags and related data from federated tagging providers. Connections is an example of a federated tagging provider.

com.ibm.wps.cp.tagging.federation.taskhandler.FederationDeleteTaskHandler
This task removes federated tags and related data from WebSphere Portal Express. It is recommended to start the task when federation of tasks is no longer required.

dayOfMonth
If you want the task to run monthly, specify a number from *1* to *31*. If the number you specify is higher than the last day of the month, the cleanup is done on the last day of the month. For example, specifying a value of *31* sets the task to run on January 31, the last day of February, March 31, April 30, and so on.

dayOfWeek
If you want the task to run weekly, specify a number from *1* to *7*, where *1* is equivalent to Monday and *7* is equivalent to Sunday.

startTime
If you specified a scheduling interval of **dayOfMonth** or **dayOfWeek**, you must specify the time of day at which you want the task to start. Use the format *HH:MM* to specify a value from *0:00* to *23:59*.

You do not need to include leading zeros, for example *4:45*. To run the task daily, use this parameter only; do not use the parameters **dayOfMonth** or **dayOfWeek**.

The `Task.xml` sample shows you how to schedule the cleanup of portal resources.

Notes:

1. When you delete a page with an object ID, and re-create the same page by using the XML configuration interface. If you use the same object ID, you might receive an error message. The error message indicates that the operation was canceled because these actions created a duplicate key value.
2. When you run a cleanup task, the XML configuration interface schedules only the task to be run in IBM WebSphere Application Server. This action does not necessarily mean that WebSphere Application Server runs the task immediately. To determine when a task started and ended, check the portal log `SystemOut.log` for the *EJPDE0005I* and *JPDE0006I* messages. These messages confirm that the task was successfully completed. After you confirmed the completion of the task, you can run the XML script for re-creating a page with the same object ID that it had before the deletion.

action Use this tag to define the required action. For more information about actions, see the topic about *Actions on portal resources*.

virtual-resource
Virtual resources in the access control subsystem. Virtual resources have access control definitions that are attached, but are not otherwise represented in the portal. The virtual resource PORTLET_APPLICATIONS, for example, allows you to give users access to all installed portlets, but does not correspond to an actual portlet. The `virtual resource` element supports only `update` and `export` actions.

user Users in the portal user repository. The definitions include their properties, for example the user's preferred language, and portal access control settings that apply to users, for example, who is allowed to administer them. The XML configuration interface allows setting the password for a user, but it does not export the password. The password attribute is required for creating new users. Therefore, you cannot directly use the response file from an XML export request to create new users; you need to add a password attribute to the XML first.

You can specify the name attribute of users and groups with a full DN (distinguished name) as in `uid=wpsadmin,cn=users,...`. Or you can specify the name attribute with a short ID as it is used for portal login, for example `wpsadmin`. An XML response file from a portal export request always contains full DNs.

With regards to the user tag, there are two special cases for which you must specify particular attributes, depending on the task you want to do:

- Exporting users and groups
- Deregistering users and groups from the portal.

Both cases are described in the following sections.

Exporting users and groups: A full portal export does not normally include user and group information, but only if you explicitly specify in your XML

request that the user is to be exported. If you want to export all user and group information, set the `export-users` flag on the main request tag to `true` as follows:

```
<request . . . export-users="true" . . .>
```

If you want to export groups without the members, set the `export-users` flag to `no-member`.

Notes:

1. Searching for users or groups is a time consuming task. A search might time out or return more results than the system can handle or the user might expect. To prevent this behavior, you can limit searches for users or groups by setting a timeout or a maximum number of search results.
2. The values of some attributes of the tag `user` correspond to settings in included `parameter` tags. If you include both in your export request, but specify different values for them, then the value set by the `parameter` tag overwrites the value set by the attribute, and is exported as the attribute. The values of the attributes of the `user` tag correspond to the included `parameter` tags as follows:

Table 142. User tag: corresponding attributes and included parameter tags

Attribute	parameter tag with name=" "	Comments
firstname	givenName	
lastname	sn	
name	uid	
n/a	cn	This tag is a mandatory parameter tag for representing the LDAP attribute.

Example: You can define the first name of a user at creation by using either the attribute `firstname` or the `parameter` tag with the `givenName` attribute. If you use both and specify two different names, then the value that is specified by `givenName` is exported as the attribute `firstname`. Example XML export request:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Description of this command file: Create 1 users -->
<request xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
         type="update" create-oids="true">
     <portal action="locate">
         <user action="update" firstname="John_1" password="password"
               lastname="Miller" name="John's name">
             <description>John Miller</description>
             <parameter name="cn" type="string" update="set">John Miller</parameter>
             <parameter name="givenName" type="string" update="set">John_2</parameter>
         </user>
     </portal>
</request>
```

This request sets the value `John_2` for the `firstname` attribute.

On some LDAP servers, where the `cn` is part of the distinguished name, you might not be able to update the `cn`.

Deregistering users and groups from the portal: If portal users or groups are removed from the user registry, but not from the portal database, or if users are muted, for example after too many wrong password attempts, you can export these users and groups for later removal. They can be

removed later by using the `cleanup-users` attribute with the `request` tag. To export these users and groups, set the `cleanup-users` flag on the main request tag to `true` as follows:

```
<request . . . cleanup-users="true" . . .>
```

If you set this property to `true`, all users and groups that no longer exist in the configured user repository, such as LDAP, are exported with their action set to `delete`. You also need to set the `export-users` attribute to `true`.

Before you reimport the file, you need to review and edit the result file and remove all users and groups that you want to keep in the portal database. During XML import, all users and groups that remain listed in the file are removed from the portal database.

Note: After you delete these entries through the modified XML script, all customization is lost for the deleted users and groups.

The sample XML file `CleanupUsers.xml` is an example of exporting users and groups.

group Groups in the portal user database. The definitions include portal access control settings and membership information, for example, which users belong to the group. For more information about how to handle groups in XML, see the preceding tag user.

markup
Definition of markup types that the portal pages support.

client Definition of client devices, how they are detected, and which markups and features they support.

device-class
Definition of a device class.

skin Describes the visual appearance, for example, the border of individual user interface elements, for example, of portlets on a page. Skins can be assigned to content nodes and components.

theme Describes basic properties, for example, colors and font type, of the appearance of an entire group of pages. Themes might restrict the selection of possible skins on the associated pages. Themes can be assigned to content nodes.

wsrp-producer
The wsrp-producer is the Producer definition on the Consumer side for Web Services for Remote Portlets (WSRP). It allows an administrator to use and integrate remote portlets that are provided by the WSRP Producer.

After you integrate a WSRP service, the portal handles it like a regular local portlet. For example, you can add the portlet to pages.

web-app
Web modules, which correspond to a deployed WAR file. To use them in the portal, one or more (concrete) portlet applications must be defined that describe specific settings. In a `portlet.xml` deployment descriptor, this element corresponds to the `portlet-app` tag.

Web modules for standard API-compliant portlets can contain only one single portlet application, whereas web modules for IBM API-compliant

portlets can contain more than one portlet application. For IBM API portlets, the `uid` attribute must match the`uid` attribute that is defined in the deployment descriptor.

Note: IBM API portlets are deprecated since WebSphere Portal Express Version 7.0, but are still supported.

For standard portlets, the `uid` attribute must be constructed by the `id` attribute of the `portlet-app` subelement and a `.webmod` suffix. If the `id` attribute is not specified, then the `uid` attribute is constructed by the WAR file name and a `.webmod` suffix. Example:

```
<web-app action="update" active="true" removable="true" uid="jsr_bookmarks_id001a.webmod">
     . . . . .
<portlet-app action="update" active="true" uid="jsr_bookmarks_id001a">
```

Depending on the path location of your WAR files, previously exported WAR files might not be found during an XML import due to incorrect path information. For information about how to use the `<url>` subtag with the `<web-app>` tag see the topic about *Importing WAR files*.

servlet

Servlets that are defined in the web module. They are created as part of the WAR file deployment and cannot be created or deleted explicitly, therefore the `create` and `delete` actions are not supported. In a standard `portlet.xml` deployment descriptor, this element corresponds to the `portlet` tag. In an IBM `portlet.xml`, one servlet is created for each `concrete-portlet` tag.

portlet-app

Portlet applications that are contained in a web module and contain specific settings. Usually, applications and their contained portlets are defined in the WAR file of the web module and are created by the portal during deployment. For IBM API-compliant portlets, this element corresponds to the `concrete-portlet-app` tag of the portlet deployment descriptor.

For standard compliant portlets, the `uid` attribute must match the `id` attribute of the `portlet-app` element that is defined in the deployment descriptor. If the `id` attribute is not set in the deployment descriptor, specify the WAR file name.

When the application is used in a WSRP context, the portlets that are contained in the application are defined remotely and can be integrated by using the XML configuration interface. In this case, portlet applications need to define a `groupid` attribute.

After you deploy a WAR file locally or integrated a WAR file as a WSRP service, you can also create more portlet applications (and the contained portlets) with different settings. This action is also known as copying or cloning the portlet application.

portlet

Portlets that can be placed on a page and contain specific settings. Normally, portlets are defined in the WAR file of the web module and are created by the portal during deployment. When the portlets that are contained in the web module are used in a WSRP context, they are defined remotely and are integrated by the XML configuration interface. In this case, portlet applications need to define a `handle` attribute. A new flag, `provided`, is introduced for providing portlets for remote invocation and withdrawing them.

When you create a new portlet in an extra application, it must refer to one of the servlets that was defined in the web module. In a `portlet.xml` deployment descriptor, this element corresponds to the `concrete-portlet` tag.

Note: If you write standard API-compliant portlets, you must not use the `parameter` tag to add parameters; use the `preferences` tag instead.

content-node
An element in the content hierarchy of the portal. The portal supports several types of content nodes:

- A page is a content node that is made up of nested layout elements and displays portlets.
- A label is a content node that serves for organizing the content hierarchy but does not display portlets.
- A static page is a content node that contains a static HTML file or an HTML fragment.
- An internal URL is a content node that points to other portal content by referencing a URL.
- An external URL is a content node that points to a web page outside the portal.

All content nodes in the portal are organized in a hierarchy; at the root of this hierarchy is the special content node wps.content.root. A content node of the type `page` can be derived from another parent content node so that it partially overrides or extends the layout of its parent. The portal and the portlet for Working with pages always display an aggregation of a composition layer and all of its ancestors. But the XML configuration interface must manage every layer separately.

Note: It is recommended to always export and replace an entire stack of page layers and *not* to use XML requests to modify individual layout components or derived page layers. In particular, do not try to manually create XML scripts for the definition of derived pages, as the reference structure is complex. Instead, use the portlet for Working with pages to edit page layouts, and then export the result into an XML response file.

component
A layout component inside a page. The portal supports two types of components:

- A container is a row or column container that aggregates child containers.
- A control component contains a portlet instance.

If you update an existing page with an XML script and the script specifies components inside that page, the `layout-processing` attribute of that page defines how those new components interact with the existing layout of the page.

portletinstance
An individual occurrence of a portlet on a page. The portlet instance includes the user-defined portlet data that was set by using the edit mode of the portlet.

Note:

1. A portlet instance is always contained in a component of type `control`; deleting a portlet instance automatically deletes the component in which the portlet was contained.
2. Instances of standard portlet API-compliant portlets must not use the `parameter` tag to add parameters; they must use the `preferences` tag instead.

For personalized content, where only portlet parameters but no page structure is changed, use the following attributes with the `portletinstance` tag:

owner Use this attribute to define the owner of the portletinstance.

parentref
Use this attribute to define the parent of the portletinstance.

cross-page-wire
Represents a property broker wiring between two portlet instances on either the same page or on different pages. A wire connects a source and a target portlet instance so that values that change in the source are propagated to the target. This tag has the `source-pageref` and `target-pageref` as the only extra attributes to the wire tag. When you export a page with cross-page wires that are connected, then the `cross-page-wire` tag is exported, even if there is no direct reference to or from the page or the wire.

Note: A wire can be created only if the wiring endpoints of the corresponding portlets exist. Legacy portlets that are not compliant with JSR 168 or 286 might create those endpoints programmatically on their first rendering. Therefore, the XML configuration interface cannot create a new wire for those portlets unless they are rendered the first time. To create this wire, first view the page that contains the portlet with a web browser and then create the wire by using the XML configuration interface.

wire Represents a property broker wiring between two portlet instances on a page. A wire connects a source and a target portlet instance so that values, which change in the source are propagated to the target.

Notes:

1. The `wire` tag is deprecated with WebSphere Portal Express Version 7.0, as it supports property broker wiring between two portlets on the same page only. Use the `cross-page-wire` tag as it supports property broker wiring between portlets on the same page and on different pages.
2. A wire can be created only if the wiring endpoints of the corresponding portlets exist. Legacy portlets that are not compliant with JSR 168 or 286 might create those endpoints programmatically on their first rendering. Therefore, the XML configuration interface cannot create a new wire for those portlets unless they are rendered the first time. To create this wire, first view the page that contains the portlet with a web browser and then create the wire by using the XML configuration interface.

credential-segment
Groups a collection of credential entries for a specific back-end credential store (vault). The configuration of credential segments cannot be modified after they are created.

credential-slot
A single credential entry that describes information that is required to

connect to a protected resource outside the portal. The XML configuration interface covers only the definition of the credential slot. The actual credential, for example the password for an application, is stored in the back-end credential store. It can be set or updated, but not exported by using the XML configuration interface.

For more information about setting user credentials by using the XML configuration interface, see the XML sample file `UpdateVault.xml` provided with the portal.

url-mapping-context
Use this tag to define arbitrary URL spaces that map to portal content.

user-resources
Use this tag to do export or delete actions for specific users.

policy-node
Use this tag to define arbitrary URL spaces that map to portal content. When you use the XML configuration interface to work with policies, some limitations apply.

application-role
Use this tag to define a compound role that combines multiple authorization roles and is specific for a set of users.

The following tags and attributes are for portal internal use only. If you encounter these tags or attributes in an XML export script that you want to use for later update, do not change these tags or their content in any way.

action-set
This tag is for portal internal use only.

category
This tag is for portal internal use only.

federation-server
This tag is for portal internal use only.

protected-resource
This tag is for portal internal use only.

resource-type
This tag is for portal internal use only.

transformation
This tag is for portal internal use only.

transformation-app
This tag is for portal internal use only.

transformationinstance
This tag is for portal internal use only.

serverref
The serverref is an attribute for the `content-node` tag; it is for portal internal use only.

content
This tag is for portal internal use only.

supported-processing-event
This tag is for portal internal use only.

supported-publishing-event
This tag is for portal internal use only.

qname
This tag is for portal internal use only.

alias This tag is for portal internal use only.

class-name
This tag is for portal internal use only.

The following tags are available for portal resources for tagging and rating: `tag`, `rating`, `custom-resource`, and `category-instance`. For more information about these tags, see the topic about *Using the XML configuration interface to administer tags and ratings*.

The XML configuration interface manages only resources of the portal core and not the resources of extra components, such as Portal Personalization.

"Special configuration data entries" on page 1095
Additional to the tags that represent portal resources as listed in the preceding section, XML provides the following tags or attributes for special purposes.

"Actions on portal resources" on page 1098
All XML elements that represent portal resources have a required `action` attribute. The `action` attribute of XML elements determines what type of processing is applied to the portal resources.

Related reference:

"Task refresh-iwidget-definitions" on page 1252
Use this configuration task to refresh iWidget definitions in the portal. This task affects all iWidget definitions that are referenced through absolute HTTP or HTTPS URLs in addition to iWidget definitions that are referenced through WebDAV URIs.

"Actions on portal resources" on page 1098
All XML elements that represent portal resources have a required `action` attribute. The `action` attribute of XML elements determines what type of processing is applied to the portal resources.

"Importing WAR files" on page 1107
To create new portlet applications, you need additional resources, the WAR files.

"Sample XML configuration files" on page 1111
Sample files are provided for your reference to help illustrate how to use XML configuration for different portal configuration purposes. Before you use them, read the other topics about the XML configuration interface carefully. Many of the samples need to be modified with valid page or user name before they can be used.

"Using the XML configuration interface to administer tags and ratings" on page 1343
You can use the XML configuration interface to manage tagging and rating in the portal. For example, you can move tagspaces and ratings between portal versions or for staging purposes.

"Using the XML configuration interface to administer analytics tags" on page 1729
You can use the XML configuration interface to manage analytics tags and site promotions in the portal.

Special configuration data entries:

Additional to the tags that represent portal resources as listed in the preceding section, XML provides the following tags or attributes for special purposes.

localedata
Describes locale specific data associated with portal resources. The `localedata` element allows a number of child elements (title, description and keywords), but not all of them are supported for all portal resources. Unsupported child elements are ignored. The `charset` attribute is only used in `markup` elements.

When you create XML files that contain lots of translated texts in different languages, it is sometimes more convenient to specify those texts in properties files instead of including them in the XML script. Therefore you can alternatively use a URL to specify a properties file. XML will then read the title, description, and keyword texts from that file.

parameter
Describes name-value pairs associated with portal resources. The `parameter` element supports a `type` attribute, but for all elements except portlet instances, the type must be `string`. Portlet instances additionally support the `binary` type which treats the parameter contents as Base 64 encoded binary data. User definitions support setting multiple values for the same parameter name. With all other resources, setting a parameter overwrites any previous value stored under the same parameter name.

If the WSRP Producer requires a registration of the WSRP Consumer with certain registration properties, you must specify these properties as parameters.

preferences
This is a derivation from the `name/value` pair. It relates to the `parameter` element. It describes name/multivalue combinations, that is, combinations of a `name` and one or more `value` child elements that are associated with the portal resources `wsrp-producer`, `portlet`, and `portlet-instance`.

Note:

1. The `value` elements support only string type attributes, no binary type attributes.
2. Setting a `preferences` value overwrites all values that were previously stored under the same `preferences` name.
3. `portlet` and `portlet-instance` can only have the `preferences` element if they comply with the standard portlet API. For all other portlets, use the `parameter` element.
4. If the WSRP Producer requires a registration of the WSRP Consumer with certain registration properties, you must specify these properties with the `parameter` tag.

Optionally, you can use the `preferences` element to define which user attributes you want to have transferred in the WSRP communication with the Producer. These user attributes are the user attributes that are defined in LDAP.

access-control
Describes the access permissions associated with portal resources. By specifying access-control subsections for resources in the XML, you can, for

example, define which users or groups are allowed to manage a resource. The access control definition for a resource includes all of the following information:

- The owner of the resource
- The roles defined on that resource
- The mapping of users or groups to a role
- Whether inheritance of a role or from the parent resource or to child resources is blocked.

Alternatively, a resource can be a private resource of a specific owner, or it can be managed externally.

When you specify users or groups in the access-control section of a resource, you can either use the full DN as in `uid=wpsadmin,cn=users,...` , or the short ID as it is used for portal login, for example `wpsadmin`. An XML response file from a portal export request always contains full DNs.

Note: When you change the access control state of a resource from public to private or vice versa, this also affects all the resources that inherit access control from this resource. You can never have a public resource that inherits access control from a private resource.

objectid

Almost all resources in the portal have an object ID, which identifies the resource. The object ID allows addressing the resource unambiguously. It is also used to express references from one resource to another. For example, when a theme is assigned to a page, the configuration of the page includes the object ID of the theme. For more information about object ID handling in XML scripts see the topic about *Object IDs in XML scripts*.

uniquename

A resource that has an object ID can optionally also have a unique name. The unique name can then also be used to identify the resource unambiguously, because a unique name can be used only once in a portal installation.

If you run an XML update that assigns a unique name which is already used on the system, the execution will fail. You can delete the unique name for a resource by setting it to the value `undefined`. When you create unique names in XML scripts that you run on many different portal installations, for example to install add-on portlets and pages, you should use a specific prefix for any unique names that you assign, to minimize the chances that they clash with existing unique names on the system. For example, the prefix `wps.` is used for all unique names that are created as part of the portal installation.

Note: When you create a nested element, for example a component, with a `uniquename` attribute, the whole hierarchy upward from that element must also have `uniquename` attributes. Example XML export request snippet:

```
<content-node ...
     <component uniquename="component_1"...
          <component uniquename"component_2"...
               <component uniquename"component_3"...
                    . . . . .
               </component>
          </component>
     </component>
</content-node>
```

Failing to do so might result in the following error message:

```
XMLC0142E: Unique name unique_name is already used in the portal.
```

ordinal

The client and component resources take an `ordinal` attribute which represents the sorting order. (In the case of client resources, if more than one client entry matches a connecting device, the entry with the higher ordinal takes precedence.)

When ordinals for resources are set in an XML script, you have the following options for specifying them:

- As a plain integer value. In this case, the value is written to the database. The resource is sorted into the position that this ordinal value has relative to the ordinals of existing siblings. For example, if you create a new page with `ordinal="350"` under an existing label, and there are already other pages under that label with the ordinals 100, 300 and 500, the new created page will appear in the third position. If you specify an ordinal which has already been assigned to an existing resource, the order of the two resources cannot be predicted.
- As a position indicator. Specify with a hash mark (**#**), followed by an integer value. In this case the resource is inserted in the sequence of resources at the position indicated by the specified value. For example, when you create a new page with `ordinal="#2"`, it appears in the second position in its content parent.
- The special values `first` and `last`. In this case, a value is chosen so that the resource is sorted at the first or last position in its content parent.

An XML export response file always contains the literal ordinal values as they are stored in the portal database. Note that specifying a position indicator or special value will not necessarily produce the required effect if you are working with derived pages, because in these cases the order of elements depends on the individual user viewing the portal.

domain

Use this attribute to specify in which database domain a specific resource should be stored. You can also use the `domain` attribute to export release data from the portal database by specifying `domain="rel"` to later feed that data it into the staging process. Use one of the following values with the `domain` attribute:

rel This value exports only the release data.

cust This value exports the release data with the customization data. This includes all referenced release data as `locate` statements.

export-release

This attribute is for use with the `request` tag. Do not use this attribute any more. Instead, use the preceding attribute `domain` by specifying `domain="rel"`.

The following two attributes have been added for the `portletinstance` tag. Use them for personalized content, where only portlet parameters are changed, but not the page structure.

owner Use this attribute to define the owner of the portlet instance.

parentref

Use this attribute to define the parent of the portlet instance.

global-target-settings

Contains those property broker actions that are defined as global targets. By using this tag, portlet instances can make Communication Targets available to Click-To-Action menus or for cross-page wiring. A `portletinstance` tag section can contain `target` elements that define a Communication Target that has been defined as global. The `target` definition specifies the communication target that is to be exported. Example:

```
<portletinstance ...>
   <global-target-settings>
      <target actionref="orderDetails" update="set"/>
   </global-target-settings>
</portletinstance>
```

You can find additional information on the meaning and possible values for configuration elements and attributes in the schema annotations.

Related reference:

"Object IDs in XML scripts" on page 1100
All resources in WebSphere Portal Express, except for the resources `portal` and the `settings`, have an object ID that uniquely identifies them in the portal. That ID is generated by the portal when the resource is created. These object IDs are represented by the `objectid` attributes in an XML export.

Actions on portal resources:

All XML elements that represent portal resources have a required `action` attribute. The `action` attribute of XML elements determines what type of processing is applied to the portal resources.

You can specify the following actions:

Table 143. Actions allowed to be set for the Action attribute of XML elements

Action	Resulting processing
`locate`	Identify the portal resource corresponding to this XML element (usually required as a context for other actions).
`create`	Create a new portal resource with the given attributes. A new resource is always created, even if another resource with the given name already exists.
`update`	Update the configuration of the corresponding portal resource with the given configuration data (attributes and dependent configuration data elements); if no corresponding portal resource can be found, it is created.
`delete`	Delete the portal resource corresponding to this XML element.
`export`	Include an XML representation of the portal resource corresponding to this element in the output of the XML command.

Example: the following XML snippet sets the portlet named "MySpecialPortlet" to inactive status:

```
<portlet name='MySpecialPortlet' action='update' active='false'/>
```

Syntactic restrictions on the input syntax:

There are certain restrictions on the allowed values for the `action` attribute.

They are listed here:

- If the request type was specified as `update`, only the actions `locate`, `create`, `update` and `delete` are permitted.
- If the request type has been specified as `export`, only the actions `locate` and `export` are permitted.
- `Create` actions (and `update` actions for non-existing elements) might require that certain attributes of the element are defined that are not required for other `update` or `delete` actions.
- Actions of nested elements must not be contradictory. If you choose to `delete` a resource, you cannot `create` any other resources inside it. The following table lists all commands, together with the allowed commands for subelements:

Table 144. Actions, together with the allowed actions for subelements

Action	Allowed actions in subelements
`locate`	`locate, create, update, delete, export`
`create`	`locate, create, update`
`update`	`locate, create`
`delete`	`no subelements allowed`
`export`	`no subelements allowed`

You can specify two or more XML elements that refer to the same portal resource. For example, you can have one element that creates a portal resource and another that updates the same resource with new configuration data.

Configuration data elements do not have an associated action, but most of them have an `update` attribute that determines the type of update that is applied. The following values are possible:

Table 145. Values for the Update attribute of configuration data elements

update	Resulting processing
`set`	The corresponding configuration data (for example, parameter) is set, or, if it does not yet exist, it is created.
`remove`	The corresponding configuration data (for example, parameter) is removed

Note that configuration data elements are processed only if their parent has an `update` action. For example, the following fragment will *not* update the capability information for the given page:

```
<client uniquename="smart.browser" action= "locate">
     <client-capability update="set">HTML_JAVASCRIPT</client-capability>
</client>
```

All specified actions are processed in the textual order in which they are specified in the XML input (document order). If there are any interdependencies between the actions, the user or program providing the XML input is responsible for ordering the elements correctly.

Object IDs in XML scripts:

All resources in WebSphere Portal Express, except for the resources `portal` and the `settings`, have an object ID that uniquely identifies them in the portal. That ID is generated by the portal when the resource is created. These object IDs are represented by the `objectid` attributes in an XML export.

References between resources are represented by these object IDs: One resource has a reference attribute that contains the object ID of another resource. For example, a portlet instance that is to be displayed on a page must reference a portlet. Therefore the `portletinstance` tag has a `portletref` attribute that corresponds to the `objectid` attribute of the portlet. Consequently, you see the following snippets in an XML export:

```
<portlet action="update" . . . objectid="Z3_G0Q03FH200A5202QRHAG4320G0" . . . >
    . . .
<portletinstance action="update" . . . portletref="Z3_G0Q03FH200A5202QRHAG4320G0" . . . >
```

All resources get an object ID assigned in the portal when they are created. That object ID can not be altered later. When you create new resources in the portal administrative user interface, they automatically get a new object ID generated by the portal. When you create a new resource with XML, it also always gets a new object ID, if you do not specify one in the XML. Note that you can not simply "invent" object IDs for new resources, because they must conform to a correct internal representation. The only way to get valid object IDs is from an XML export.

In XML scripts the `objectid` attribute of a resource is used for the following purposes:

- To look up the resource, if the action is `locate`, `export`, `update` or `delete`.
- To set the object ID for a new resource, if the action is `create`.
- To set the object ID for a new resource, if the action is `update` and no resource with that object ID exists.
- To describe links from one resource to another.

You can use object IDs to uniquely specify resources that you want to administer. For example, the following snippet deletes a specific known page. (You would normally get the object ID of the page from an XML export.)

```
<content-node action="delete" objectid="Z6_G0Q03FH200A5202QRHAG4320O0"/>
```

The following snippet looks up a page with a specific object ID. If it cannot find the object ID, it creates it. If it already exists, it updates it.

```
<content-node action="update" objectid="Z6_G0Q03FH200A5202QRHAG4320O0" type="page" . . . >
```

The next snippet creates or updates a theme with a specific object ID and then assigns that theme to a label:

```
<theme action="update" objectid="ZJ_G0Q03FH200A5202QRHAG4320S1" . . .>
    . . .
<content-node action="update" objectid="Z6_G0Q03FH200A5202QRHAG4320O0"
            type="label" themeref="ZJ_G0Q03FH200A5202QRHAG4320S1" . . . >
```

If the theme already exists with the specified object ID, you can directly use that object ID in references without having to include the theme in the XML script. The next snippet assumes that the theme has already been created. For example, it might have been copied from another server in a previous step. Therefore the snippet only assigns the theme to the label:

```
<content-node action="update" objectid="Z6_G0Q03FH200A5202QRHAG4320O0"
     type="label" themeref="ZJ_G0Q03FH200A5202QRHAG4320S1" . . . >
```

In this case, the theme is looked up in the portal data store using its object ID `ZJ_G0Q03FH200A5202QRHAG4320S1`. If no theme with that object ID is defined in the portal, you get an error during the XML validation.

An object ID is globally unique. Two object IDs that were automatically generated by different portal installations can never be the same. Therefore you can exchange resources between different portal installations using XML export and update requests without having to worry about possible object ID conflicts. The only way that you can ever duplicate an object ID is by transferring a resource (including the object ID) to another portal with an XML export and update.

In many cases, this is the required behavior. However, if you do not want to copy the same resource to another portal, but you want to create a new resource instead, regardless of existing OIDs and without any chance of causing conflicts, you must either use symbolic object IDs or delete the `objectid` attribute from the XML script. In the latter case the portal creates a new object ID.

"Symbolic object IDs and ID generating mode"
In some cases, you might need to use object ID attributes to express references between resources in your XML script, but you do not want these to be read from or written to the portal database. In this case, the object ID would be only a symbolic reference inside the XML script.

"Lookup of portal resources" on page 1103
XML elements with `locate`, `export`, `update` and `delete` actions need to refer to existing resources in the portal. Those resources must be identified by specific attributes.

Symbolic object IDs and ID generating mode:

In some cases, you might need to use object ID attributes to express references between resources in your XML script, but you do not want these to be read from or written to the portal database. In this case, the object ID would be only a symbolic reference inside the XML script.

For example, you might want to create a new theme and page, and reference the theme in the page. Nevertheless you want to let the portal chose an object ID for it because you do not want to accidentally overwrite an existing resource.

There are two ways to achieve this:

- You can switch the XML processing to **ID generating mode** by setting the `create-oids` flag on the main request tag:

  ```
  <request . . . create-oids="true" . . . >
  ```

  In this mode, all object IDs are not written to or read from the portal database. Instead, their only purpose is to express the linking between resources in the XML. When the XML processing creates new resources, they are created with a new system generated object ID, not with the object ID specified in the XML.
- When you use a value for an `objectid` attribute that cannot be decoded as an object ID, such as a simple string name, the XML processing treats it as symbolic and does not try to use it for looking up the object or write it to the database. This makes it possible to selectively treat individual object IDs as symbolic.

Note that even in symbolic object IDs, anything after the first space is not significant for processing. For example, you could use the following snippet to create a portlet and put it on a page using a symbolic object ID:

```
<portlet action="update" . . . objectid="Welcome_Portlet" . . . >
    . . .
<portletinstance action="update" . . . portletref="Welcome_Portlet" . . . >
```

As the object ID values are purely symbolic, you cannot use them in a reference without first "defining" them in the same XML script. Before you can use the `portletref="Welcome_Portlet"` attribute specification in an XML update, you must also have a portlet with `objectid="Welcome_Portlet"` defined in the same XML; otherwise a syntax error is reported.

Of course, the object IDs in the XML are also not used for looking up a resource. If you want to refer to an existing resource, you need to use a unique name instead. For details see the information later in this section and in the topic about *Lookup of portal resources*. In ID generating mode, the following snippet locates an existing portlet by its name, "defines" a symbolic object ID for it and places the portlet on a page:

```
<portlet action="locate" name="Welcome Portlet" objectid="Welcome_Portlet">
     . . .
<portletinstance action="update" . . . portletref="Welcome_Portlet" . . . >
```

As object IDs are not used to identify existing resources in ID generating mode, it is good practice to define unique names for all resources that are created in such scripts. That way, if the script is executed twice, the second execution can find and update the resource by its unique name, instead of creating two identical resources.

When you create resources using symbolic object IDs, it can sometimes be useful to know the actual object IDs of the new resources. You can set the export-mapping flag on the main request attribute to obtain this information:

```
<request . . .export-mapping="true" . . . >
```

When you set this flag, a mapping section is appended to the XML response. For every symbolic object ID given in the input, this mapping shows the actual object ID in the portal data store.

The ID generating mode is useful if you want to create an XML script that installs a group of new resources and is executed on many different portal installations. This can be, for example, a part of the installation procedure of a portal add-on that you give out to other parties. In this case, you have no control over the systems on which the XML script is executed, and it is of no interest to you which object IDs the resources actually get.

Note: Use the ID generating mode for all the examples under Working with the XML configuration interface, because they should work on any portal installation.

The identity of the objects that are configured in the XML script is expressed by their object ID. Therefore you should use "real" object IDs in your scripts, when you want the objects in your XML to retain their identity. For example, when you use an XML export request and the resulting response file to copy a resource from a staging to a production system, the resource is created on the production system with the same object ID as on the staging system. This way you can establish a correspondence between the two resources. When you later transfer the same

resource again, the XML processing looks up the resource by its object ID, finds that it already exists, and updates the existing resource instead of creating a new one.

When setting up the target system in such scenarios, you should use only the XML configuration interface to copy resources from the source system. If you deploy portlets on the target system during the portal installation or if you deploy them using the administration portlets, they will not have the same object IDs as on the source system, so you can run into problems when you later copy other resources that reference them.

Lookup of portal resources:

XML elements with `locate`, `export`, `update` and `delete` actions need to refer to existing resources in the portal. Those resources must be identified by specific attributes.

The relevant attribute to identify a resource in the portal is its object ID. Every resource must have an object ID and it must always be unique. Therefore, if you specify an `objectid` attribute for a resource, and you do not use symbolic object IDs as described earlier, the resource is looked up by that object ID.

Of course, there are cases where you do not have literal object ID values available when you write your scripts, especially if you are writing scripts that are executed on installations that you are not administering yourself. Therefore you can also specify other identifying attributes to look up resources. If the lookup by object ID fails, the XML processing also attempts to find the resource using other attributes.

An alternative method for looking up portal resources is to use a unique name. Every resource that has an object ID can also have an optional unique name, and the unique name must unambiguously identify the resource. Unique names are useful if you need a symbolic way to identify certain resources. They allow easy porting of configurations between portal installations. In contrast to object IDs, it is possible to modify unique names of resources, which can be an advantage in certain situations. To set a unique name for a resource, use the Custom Unique Names portlet under Administration, Portal Settings.

If a unique name is not given or cannot be found, some resources can also be searched using other attributes. Some resources can be looked up without any attribute information, because they exist only once in their context.

The following table shows the relationship between resources and the attributes you can use for locating them:

Table 146. Resources and attributes for locating them

Resource key	**Attributes used for locating the resources**
`portal, global-settings, services-settings`	None; these items always exist only once.
`markup, virtual-resource, user, group, credential-segment, credential-slot, portlet`	`name`
`web-app, portlet-app`	`uid`
`servlet`	`name` **Note:** The `refid` is used as fallback for XML imports from earlier portal versions that do not contain the `name` attribute.

Table 146. Resources and attributes for locating them (continued)

Resource key	Attributes used for locating the resources
portletinstance	None; there is at most one portletinstance per component.
url-mapping-context	label

In any case the lookup process first tries to find the resource by its object ID, if specified, and then by its unique name, if that is specified. Only when those attempts fail, other attributes are used for locating the resource.

Note: If an objectid attribute is specified in the XML input, but the corresponding resource cannot be found by that object ID but only by another attribute, and if that object ID is used in other parts of the XML script as a reference, those references are mapped to the actual object ID for the resource that was found. In this case the objectid attribute behaves like a symbolic object ID as described earlier.

Replacement variables in XML configuration interface script files:

XML script files that were created by an XML configuration interface export or that can be imported by the XML configuration interface can contain URLs to portal files. These URLs reference files that are in WebSphere Portal Express server directories. Depending on the installation directory of your WebSphere Portal Express installation, these file locations can differ. You can avoid the dependency on the file location by using variables.

For example, the web application of the portal login module contains the following XML element:

```
<url>file://localhost/$archive_root$/login.war.webmod/login.war</url>
```

On a Linux installation of WebSphere Portal Express, this URL might be interpreted as follows:

```
<url>file://localhost//opt/WebSphere/PortalServer/login.war.webmod/login.war</url>
```

On other WebSphere Portal Express installations, the login.war file can be in a different directory.

As the XML configuration interface uses different values for the variable replacement, the XML input file is independent of the specific WebSphere Portal Express installation directories.

The following table shows the available variables and provides examples of the replacement values they might have on a Linux installation of WebSphere Portal Express:

Table 147. Replacement variables in XML configuration interface script files

Variable for WebSphere Portal Express installation directory	Example replacement value
$app_install_root$	/opt/WebSphere/wp_profile/installedApps
$archive_root$	/opt/WebSphere/wp_profile/PortalServer/deployed/archive
$predeployed_root$	/opt/WebSphere/wp_profile/installedApps/wpsbvt

Table 147. Replacement variables in XML configuration interface script files (continued)

Variable for WebSphere Portal Express installation directory	Example replacement value
$profile_install_root$	/opt/WebSphere/wp_profile/installedApps
$server_root$	/opt/WebSphere/PortalServer
$user_install_root$	/opt/WebSphere/wp_profile
$wp_profile_root$	/opt/WebSphere/wp_profile

Exporting sets of resources:

You can specify more than one resource with an export action in the same request and thus generate an export response file that contains a selected group of resources, for example several portlets and pages.

The XML configuration interface provides two additional features that allow you to export selected subsets of the portal resources:

- When you export a content node, you can specify the `export-descendants` attribute for that resource. If you set this attribute to `true`, the export response file also includes the entire subtree in the content hierarchy that is located under that node. In addition, all derived pages that override the layout of pages in that subtree, are also exported. In other words, this exports all content nodes that can be reached via a chain of `content-parentref` or `derivation-parentref` attributes.

  Example: the following fragment exports a subtree of the content hierarchy that starts with the label as specified:

  ```
  <content-node uniquename="MyPages" action="export"
          export-descendants="true" />
  ```

- All first level resources that can take an `objectid` attribute, such as `markup`, `virtual-resource`, `user`, `group`, `client`, `event-handler`, `web-app`, `theme`, `skin`, `content-node`, `credential-segment` and `url-mapping-context` support the asterisk (*) as a wild card symbol that you can use as a value for the object ID. The asterisk can be used as a wild card symbol only with the export and delete actions. Depending on the action with which it is specified, it exports or deletes all resources of the respective type. For example: the following fragment exports the complete client configuration of the portal:

  ```
  <client objectid="*" action="export"/>
  ```

Note: A combination of a partial search string and the asterisk is not valid. The asterisk also has no special meaning if it is used as a value for any other attribute.

The XML configuration interface offers no other "query" features, that allow you to export resources based on specific criteria. The only other possibility to export a selected subset of resources is to specify all the resources individually with their object IDs or other identifying attributes in your XML input.

Mandatory and optional attributes:

Depending on the action that you perform by using the XML configuration interface, some attributes can be mandatory or optional.

Normally, you to specify only that part of the configuration data for an XML element that is necessary for the required operation. For example, when you delete

a portlet, it is sufficient to specify its reference ID to identify it; it makes no sense for this operation (although it is not forbidden), to specify a new active state, since the portlet is removed anyway.

When creating a new portal resource, some required attributes (depending on the type of resource) must be specified. Others can be omitted. They are then set to a default value.

When you update an existing portal resource, all attributes are optional, except those required to locate the element. The omitted attributes remain unchanged. In a few cases of page layout attributes, there is the possibility of explicitly specifying an "undefined" value. This means that the attribute is not defined at the respective level, but inherited. For example, if the skin for a component is undefined, it will be inherited from the setting of its page.

Note that there is a semantic difference between the following XML fragments:

```
<content-node uniquename="MyPages" action="update" active ="true"/
```

and

```
<content-node uniquename="MyPages" action="update" active ="true" skinref="undefined"/>
```

The first fragment only modifies the active attribute of the page and leaves its skin setting unchanged; the second fragment additionally resets the skin to the `undefined` value (whatever the previous skin setting was), so that the page will always display in the portal default skin.

Page layout modifications:

When you use an XML script to update an existing page with a new layout, you create or update child elements of type `component` for the `content-node` element of the page. Normally you use the XML script to define a complete new layout of the page rather than combine the existing layout with your new definitions. In such cases the XML configuration interface applies special processing.

It proceeds as follows: After it has updated the page layout, it deletes all components in the page that existed before but were not updated by the script. As a result, the page contains only those layout components that are specified in the XML script and no remainders of its previous layout. Otherwise you could easily end up with an invalid component structures.

In particular, this means that if you update a page layout in ID generating mode, all existing components of the page are deleted, and a new layout is created instead, even if the new layout is identical to the old one. This happens because components can only be looked up by their object IDs, and lookup by object ID is not possible in ID generating mode. Therefore all the components specified in the XML script are created, because they cannot be found for updating, and all existing components are deleted because they were not updated.

In the rare case that you actually want to update specific components in the page but do not want to delete the existing page layout, you can turn off this special processing by specifying the attribute `preserve-old-layout="true"` for the content node.

Marking pages as hidden under the content root:

By default, pages that you create under the content root display in the main menu. If you do not want a page that you create to appear in the main menu, you can hide the page.

You do this by setting the `hidden` flag for the page parameter for the `content-node` tag in XML. Use the following XML snippet:

Note: You can still view and work with pages that are marked as hidden in Administration portlets. You can also create a direct URL to the hidden page so that the page can be accessed from other areas of the site, such as the page menu.

```
<content-node action="update" ...>
  ....
<parameter name="com.ibm.portal.Hidden" type="string" update="set"><![CDATA[true]]></parameter>
 ...
</content-node>
```

Importing WAR files:

To create new portlet applications, you need additional resources, the WAR files.

You cannot include those WAR files in the XML input. Instead, you can add references to external URL locations to the XML input. See the following example:

```
<web-app uid="MySpecialPortlet" action="create">
        <url>file://localhost/C:/myportlets/Special.war</url>
</web-app>
```

The WAR files are not used in the running portal. However, when you process the XML request, the WAR files referenced in the XML script must be accessible to the portal. When you update a package and specify a `<url>` subelement, the WAR file is re-deployed, just as if you had selected the update of a portlet application in the browser. If you intend to deploy the same configuration into several new portals, you can set the URL to `http://deploymentserver/path/filename.war` . This way there is no need to copy all WAR files to each server machine. The `deploymentserver` machine needs to be set up properly so that the WAR files can be accessed by http.

An XML export request does not create any archive files that might be required. Instead, it only creates pseudo-references in the form of file URLs that rely on the assumption that the file resides in the `/installableApps/` subdirectory of the portal installation. If these assumptions are not met, an exported portal configuration cannot be successfully re-created without editing the generated URLs manually. A back up of a portal configuration requires that the WAR files required for redeployment are saved in addition to the XML export.

Importing static page content from archive or compressed files:

You can import the content of static pages from an external archive or compressed file by using the XML configuration interface.

The following example imports static page content from the file `index1.zip` :

```
<content-node action="update" active="true" allportletsallowed="true"
              content-parentref="homepage" create-type="explicit"
              domain="rel" ordinal="1500" themeref="ibm.theme"
              type="staticpage" uniquename="samplestaticpage1">
          ...
          <pagecontents markup="html" display-option="inline">
```

```
            <url>file:///$server_root$/doc/xml-samples/index1.zip</url>
        </pagecontents>
        ?
</content-node>
```

The referenced archive or compressed files must be accessible to the portal when the XML request is processed. For details about the administration of static pages with the XML configuration interface refer to the topics about *Using the XML configuration interface to work with static pages*.

Related reference:

"Exporting and importing static pages" on page 1872
You can work with static portal pages by using the portal XML configuration interface. Learn about the tasks that you can perform and the XML elements for working with static pages.

Viewing updates and changes made with the XML configuration interface:

For your users to be able to view updates that you made by using the XML configuration interface, they might have to log out and log back in again, or you might have to restart the portal. This depends on the type of update that you made.

Most portal data is cached on a per-user basis, therefore many modifications become visible to users only after a new logout and login. For example, a page that is created on behalf of users does not immediately become visible to those users if they are currently logged in. Other modifications only become visible after some timeout when internal caches are refreshed with current data. For some settings the portal needs to be restarted to activate the updates. Therefore, if an update you made does not become visible, restart the portal.

In general, the effect of an XML update is the same as if the update was made using administration portlets using a new browser and login session.

Transactionality:

When you use XML scripts to create, update or delete resources, the changes in the portal database are grouped into transactions. All changes that are part of one transaction are either executed completely or not at all. The XML configuration has two different levels of grouping database updates into transactions.

The grouping is defined by the `transaction-level` attribute of the main request element, which can have the following values:

resource
: Every top-level resource in the XML script is processed in one separate transaction. For example, this can be a content node with its complete layout. If an error occurs, all resources up to the one where the error was encountered have been fully processed; the resource where the error was encountered is not created, or, if it already existed, it remains unchanged.

request
: The entire XML script is executed in one transaction. If an error occurs, all database changes caused by the script are made undone, and the original state is restored. Note that using this level of transactionality might cause large and long-running database transactions if used in large XML scripts. As a result, you might encounter database errors caused by exceeding

database limits on transaction duration or transaction log size, depending on the configuration of your database.

none No explicit transactions will be opened for the processing. This is the default value.

Transactionality applies only to changes in the portal database. The following aspects of resources are not stored in the portal database and therefore not included in transactions:

- Enterprise applications for portlets that are deployed into WebSphere Application Server
- User and group information
- Role assignments in an external access control system.

An example of what this means is as follows: When you deploy a WAR file in an XML script that uses `transaction-level="request"` and an error occurs later in the execution of the XML script, the transaction is canceled, so the entries for the portlet are removed from the portal database. However, the corresponding enterprise application has already been deployed into WebSphere Application Server and is not removed. This will not further affect the operation of the portal; you can simply deploy the portlet again later. You will just have an unused enterprise application in WebSphere Application Server. Remove it manually.

Error recovery:

If errors occur during the processing of an XML script, the XML result file contains an error message. After fixing the cause of the error, you have two options to continue.

Perform one of the following two options:

1. Run the entire XML script once again.
2. Remove all resources before the point where the error occurred from the XML script and run only the rest of the XML script.

If the error occurs during the validation of the XML script, and no resources have actually been processed so far, you can simply run the entire script again. You can verify this by reviewing the progress reporting comments in the XML response.

If the error occurs after some resources have actually been processed, the best option depends on several circumstances:

- If you used the `request` transaction level, you must run the entire script once more, because all changes have been undone.
- If you used the `resource` transaction level, the preferable option is to run only the rest of the XML script, and not to repeat changes that have already been made; otherwise you might duplicate resources that were already created. You can only use this option if the rest of the XML script does not contain references to the resources that you are removing, that is, if you do not refer to symbolic object IDs of these resources. See Symbolic object IDs and ID generating mode for more information. You can always remove resources that have already been processed from the XML script, if your script uses only hard-coded object IDs for references, because in this case all references can be resolved by looking up the object IDs in the portal database.

To make error recovery easier, use scripts that can be run again partially or completely without the possibility of duplicating resources. To do that, specify an

object ID or another identifying attribute on every resource in the script and use only `update` actions. This way resources are simply overwritten with the same configuration if they have already been created. For more information about how to specify attributes, see the topic about *Mandatory and optional attributes*.

Related reference:

"Symbolic object IDs and ID generating mode" on page 1101
In some cases, you might need to use object ID attributes to express references between resources in your XML script, but you do not want these to be read from or written to the portal database. In this case, the object ID would be only a symbolic reference inside the XML script.

"Mandatory and optional attributes" on page 1105
Depending on the action that you perform by using the XML configuration interface, some attributes can be mandatory or optional.

Hints and tips for using the portal XML configuration interface:

In an example configuration, you might have two WebSphere Portal Express environments that are both configured for security with an LDAP server. However, the two LDAP servers have different directory structures.

For example, this can be different LDAP suffixes for the users or groups. To transfer such a portal configuration from one portal to the other, you can use the following XML script:

```
<?xml version="1.0" encoding="UTF-8"?>
<request
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
    type="export"
    export-users="false">
    <portal action="export"/>
</request>
```

When you use this XML script to transfer the configuration data between these two environments, be aware of the following:

1. By setting the tag `export-users` to `false` you only prevent the export of the LDAP hierarchy. Ownership and access control rules are still exported.
2. During the transfer all user-related information is lost, as the target portal does not know the user information from the source portal. For example, this affects access rights or ownership of private pages. You might see a warning about missing user or group information, but it should not prevent a successful import.
3. If you use this script for your export, you might find that your XML import fails with an exception and references one of the following two items:
   - Credential slots and segments. To avoid exceptions centering around the credential slots and segments, remove the references to these elements from your XML prior to running your XML import.
   - Private pages. The destination server cannot use information about private pages. To address exceptions centering around the private pages, use the following script for the XML export:

```
<?xml version="1.0" encoding="UTF-8"?>
<request type="update"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd">
```

```
    <portal action="locate">
    <content-node action="export" name="*" create-type="explicit"/>
</portal>
</request>
```

This procedure exports all pages which are not private, along with the information that is required to put the portlets on the pages. However, you must either deploy the portlet applications on the target portal prior to running the XML import, or you must modify the XML script to deploy the portlets in the same run.

Sample XML configuration files:

Sample files are provided for your reference to help illustrate how to use XML configuration for different portal configuration purposes. Before you use them, read the other topics about the XML configuration interface carefully. Many of the samples need to be modified with valid page or user name before they can be used.

Sample file location

The sample XML configuration files provided with the portal are located in the following directory: *PortalServer_root*/doc/xml-samples.

Sample file list

Some of the XML samples are listed in the following. This list is not complete. All samples are located under the directory given in the previous section.

- Export.xml
- ExportRelease.xml
- ExportPage.xml
- ExportPageResult.xml
- CreatePage.xml

  Note: If you do not want to set the page title for all portal supported locales, you need to set it at least for the default system locale of your portal.
- CreateCsaPage.xml
- CreateLegacyPage.xml
- CreatePageFromZip.xml
- CreatePageFromTemplate.xml
- DeployPortlet.xml
- ClonePortlet.xml
- ModifyPortlet.xml
- ExportPortletAndPage.xml
- ExportSubTree.xml
- UpdateAccesscontrol.xml
- UpdateVault.xml
- CopyPage.xml
- CreateURL.xml
- CreateUser.xml
- CreateLanguage.xml:

This XML sample adds a new language for the portal or removes an existing language from the portal.

Note: To define a new language for your portal, set the title for the new language in all locales that are supported for your portal in the XML file. If you do not want to set the title for all portal supported locales, you need set the title at least for the default system locale of your portal.

- DeployTheme.xml
- ExportAllPortlets.xml
- Transaction.xml
- MovePage.xml

  Note: The actual move of the page is done by the last two lines.
- ActivatePortlet.xml

  Use this sample to change the states of portlets, portlet applications, or Web applications between active and inactive by setting the attribute `active` of the appropriate tag to `true` (for active) or `false` (for inactive). The sample activates the respective resources.
- Task.xml

  This sample creates a scheduler task for cleaning up portal resources, com.ibm.portal.datastore.task.ResourceCleanup.
- RegisterPreDeployedEAR.xml

  Use this sample to install a predeployed portlet. You might have to change this sample for your requirements.

  Notes: For the *`deploy_target_directory`* in the `url` tag specify the directory to which you deployed the EAR file on the WebSphere Application Server. The default target directory is `AppServer`, but when deploying portlets for your portal it is a good option to specify `PortalServer` as the target directory.

  An Enterprise Application Archive (EAR) can hold more than one Web Application Archive (WAR) files. To configure the pre-deployed portlet resources into portal you need to reference each Web Application resource separately in the XML configuration script. Provide a dedicated `<web-app>` statement with the corresponding data in the XML script for every WAR file that contains a portlet application in that EAR file.
- CleanupUsers.xml

  You can use this sample to identify users and groups in your portal database who have been removed from the user registry, but not from the portal database. In order for the file to work properly, you must set both attributes `cleanup-users` and `export-users` to `true` . Running this sample file results in a file that lists those users and groups and marks them for deletion. The result file also lists all users who have been muted, for example after too many wrong password attempts. Before you re-import the file, check the file and remove all users and groups that you want to keep in the portal database. During XML import all users and groups that remain listed in the file will be removed from the portal database.

  Note: After deleting these entries via the modified XML script, all customizations are lost for the deleted users and groups.
- ExportIncludingOrphanedData.xml

You can use this sample file to perform an export that includes all orphaned data. You do this to prepare for deleting the orphaned data. Note that this sample uses the request type `export-orphaned-data`.

Related tasks:

"Exporting a Producer definition by using the XML configuration interface" on page 1482
You can use the XML configuration interface to export a Producer definition. You might, for example, export the Producer from a test portal to update your production portal with it later.

"Creating a Producer definition and consuming a portlet by a single XML script" on page 1482
You can use a single XML script to create a Producer definition and then consume portlets from that same Producer.

Related reference:

"XML samples for creating or removing language definitions" on page 1427
You can modify these XML samples and use them to create or remove language definitions from your portal.

"XML samples for creating Producer definitions" on page 1480
You can modify use these XML samples and use them to create Producer definitions,

Portal Scripting Interface

You can use the Portal Scripting Interface to configure your portal by running scripts from a command line.

"About the Portal Scripting Interface"
IBM WebSphere Portal Express provides a scripting interface that enhances the possibilities for automated solution deployment and administration of the portal. The Portal Scripting Interface allows you to create scripts that portal administrators can use to perform administrative tasks from a command line.

"Getting started with the Portal Scripting Interface" on page 1115
With the Portal Scripting Interface, you can administer your portal installation by executing commands. You can do that either interactively by typing the commands into a console window with interactive scripting and pressing the Enter key, or you can write the commands into a file and run that file. Here are some hints and examples for using the Portal Scripting Interface.

"Working with the Portal Scripting Interface" on page 1120
Learn more about the different modes that you can use with the Portal Scripting Interface.

"Command reference for the Portal Scripting Interface" on page 1126
The IBM WebSphere Portal Express Scripting Interface component provides a scripting interface for the administration functions.

Related reference:

"Portal Scripting Interface and content associations" on page 2078
With the Portal Scripting Interface, you can create scripts to automate the management of content associations. Using the ContentMapping bean with the Portal Scripting Interface, you can add, modify, and remove content associations.

About the Portal Scripting Interface

IBM WebSphere Portal Express provides a scripting interface that enhances the possibilities for automated solution deployment and administration of the portal. The Portal Scripting Interface allows you to create scripts that portal administrators can use to perform administrative tasks from a command line.

The Portal Scripting Interface allows portal solution development teams to write scripts that are later executed by operation teams for solution deployment. These scripts have the same functionality as the portal administration user interface. This allows you to implement automated configuration management for various kinds of configuration changes.

Scripts can help you split the administrative workload between solution development and solution operation teams. Even if the solution development teams cannot work interactively with the productions system, they can apply the same administrative actions through the use of scripts. At the same time, the use of scripts enhances availability and quality of the solution as developers can write and test the scripts without interfering with the production system. Scripts provide repeatability and avoid user errors that are likely in manual administration procedures.

In addition to these benefits, portal scripts provide the following advantages:

- The Portal Scripting Interface provides delegated administration in the same manner as the portal administrative user interface. This allows distributed portal administration as follows:
  - Different development teams can work on related portal updates without interfering with each other's work.
  - Different administrative teams can perform the tasks of developing a solution, and deploying and operating that solution for production. These teams can be within the same organization (for example in the same enterprise), or in independent companies, such as independent solution centers for the operation. In a typical scenario, the solution development team may have in-depth knowledge about the software solution internals, the operation is focused on the external characteristics of the solution. The operation solution team can receive the solution from the development team as a black box that can be operated without much knowledge of the solution internals. This allows enterprises to use automated solution deployment and distributed staging processes.

  In this regard the portal scripting interface goes beyond the XML configuration interface that has been provided by WebSphere Portal Express for several releases now. The XML configuration interface does not allow for easy separation of distributed portal administration.
- You can use the portal scripting interface for staging and integration of new releases A new release can be developed and tested on a test system and can then be integrated into the production system while the system is running.
- You can use the scripting interface for all of the following:
  - Release staging and integration
  - Updates of portal content, portal configuration, releases. This includes, for example, adding, replacing, or removing components.
- Interactive portal administrative tasks do not require a Web browser.
- Operators can enhance productivity and quality by scripting repeated administrative tasks for automated administration and maintenance.
- You can apply portal configuration updates in real time.
- The scripting interface is an extension of the WebSphere Application Server scripting interface. This means that if you are already familiar with the interface, you should easily learn how to use Portal Scripting Interface.
- You use the scripting interface through the use of a user ID, similar to the portal UI.

- They are easy to use:
  - You can use any text editor to develop scripts.
  - It provides its own online help as part of the scripting environment. You do not need to leave the script window for command syntax descriptions.

Change from JACL to Jython syntax

In previous versions of WebSphere Portal Express the Portal Scripting Interface was based on JACL syntax. Starting with WebSphere Portal Express Version 6.1 the Portal Scripting Interface is based on Jython syntax. The JACL Version 7.0 syntax is still available and supported.

The Jython syntax can be derived from the JACL syntax in a generic way as follows:

JACL: `$Object method arg1 arg2` Example: `$Portal login myuserid mypassword`

Jython: `Object.method(arg1, arg2)` Example: `Portal.login('myuserid', 'mypassword')`

Getting started with the Portal Scripting Interface

With the Portal Scripting Interface, you can administer your portal installation by executing commands. You can do that either interactively by typing the commands into a console window with interactive scripting and pressing the Enter key, or you can write the commands into a file and run that file. Here are some hints and examples for using the Portal Scripting Interface.

About this task

Note: To simplify things, the following examples are based on a Linux portal installation in the directory `/opt/WebSphere`. If you use a different system or a different installation path, adopt the examples so.

"Opening a console window for interactive scripting" on page 1116
To start the Portal Scripting Interface from a console window, use the procedure described here.

"Logging in to the portal" on page 1116
Before you can work with portal scripting commands, you need to log in to the portal server instance where you want to work with the Portal Scripting Interface.

"Working with portal pages and other resources " on page 1116
The following exercise shows you some steps how to work with portal resources, such as pages and portlets, search for resources and information about them, and add a portlet to a page.

"Getting help for a command" on page 1118
The Portal Scripting Interface provides more commands and variants of commands than shown in the previous topics. To learn more about these commands and their syntax and function, and beans and parameters, refer to the integrated help of the Portal Scripting Interface.

"Logging out of the portal" on page 1120
After you complete working with the Portal Scripting Interface, you log out and exit the interactive scripting console.

Opening a console window for interactive scripting:

To start the Portal Scripting Interface from a console window, use the procedure described here.

Procedure

1. Change into the directory where your Portal Scripting Interface is installed: `cd /opt/WebSphere/PortalServer/bin`
2. Call the Portal Scripting Interface startup command file: `wpscript.sh|bat`.

   Note: You must indicate that you want to enter the commands in Jython as follows: `-lang jython`. If you prefer to use the JACL Syntax, you can type `-lang jacl`.

   a. Type the command as follows: `./wpscript.sh -lang jython`. This command starts the interactive scripting console. The Portal Scripting Interface prompts you for a user ID and password. The Portal Scripting Interface is an extension to the WebSphere Application Server `wsadmin` tool.

   b. Use a valid WebSphere Application Server administrator user ID. The Portal Scripting Interface returns the following response:

   ```
   WASX7209I: Connected to process "WebSphere_Portal" on node wpsbvt using SOAP connector;
              The type of process is: UnManagedProcess
   WASX7031I: For help, enter: "print Help.help()"
   ```

3. You can now enter Portal Scripting commands as required. You can also use all the available WebSphere Application Server `wsadmin` commands.

Logging in to the portal:

Before you can work with portal scripting commands, you need to log in to the portal server instance where you want to work with the Portal Scripting Interface.

About this task

At the command prompt `wsadmin>` use the following command: `Portal.login("your_userid", "your_password")`:

```
wsadmin> Portal.login(your_userid, your_password)
```

If you logged in successfully, the Portal Scripting Interface returns a message that starts with `logged in as . . .`. The console now shows the following lines:

```
wsadmin> Portal.login(your_userid, your_password)wsadmin>
logged in as "uid=your_userid,o=defaultWIMFileBasedRealm"
```

You can now administer the portal instance in a similar way as by using the Portal Administration user interface. The following examples show how you can work with pages.

Working with portal pages and other resources :

The following exercise shows you some steps how to work with portal resources, such as pages and portlets, search for resources and information about them, and add a portlet to a page.

About this task

Searching for a portal page:

Search for the **Getting Started** page. Enter the following command and specify the page by its unique name:

```
wsadmin>Content.find("page", "un", "ibm.portal.Home.Getting Started")
```

If a page with the specified unique name exists, the Portal Scripting Interface returns the object ID of the page as follows:

```
wsadmin>Content.find("page", "un", "ibm.portal.Home.Getting Started")
   'Z6_CGAH47L00OJCC0I6U1NESJ2GK0'
```

If you are not sure whether the resource is a page or a label you can specify the type `any` instead of `page`. In this case, you get the following response:

```
wsadmin>Content.find("any", "un", "ibm.portal.Home.Getting Started")
   'Z6_CGAH47L00OJCC0I6U1NESJ2GK0'
```

To find out more about the different ways of searching for portal resources, use the command `help()` that the Portal Scripting Interface provides for each command. For an example, see the section about Getting help for a command.

Finding out information about a portal page:

To display some standard information about a portal resource, use the `details()` command. You need to specify for which resource you want more detailed information. The following example can be the portal page Getting Started:

```
wsadmin>Content.find("page", "un", "ibm.portal.Home.Getting Started", "select")
'Z6_CGAH47L00OJCC0I6U1NESJ2GK0'
wsadmin>Content.details()
```

This returns the following response:

```
wsadmin>Content.find("page", "un", "ibm.portal.Home.Getting Started", "select")
'Z6_CGAH47L00OJCC0I6U1NESJ2GK0'
wsadmin>Content.details()
name: ibm.portal.Home.Getting_Started
id  :Z6_CGAH47L00OJCC0I6U1NESJ2GK0
type: staticpage
      no children
```

To find information about a portal resource, you need to specify the resource. If you do not specify the resource, for example, by entering only `wsadmin>Content.details()`, you get a response such as the following one:

```
WASX7015E: Exception running command: "Content.details()"; exception information:
   com.ibm.bsf.BSFException: exception from Jython:
Traceback (innermost last):
   File "<input>", line 1, in ?
EJPXD0020W: No object has been selected.
```

Here are some more examples for requesting information about a portal page:

```
wsadmin>Content.get("type")
   'staticpage'
wsadmin>Content.get("uniquename")
   'ibm.portal.Home.Getting Started'
wsadmin>Content.get("allportlets")
   'true'
```

```
wsadmin>Content.nlsget("title", "en")
   'Getting Started'
wsadmin>Content.parmget("com.ibm.portal.bookmarkable")
   'Yes'
```

The following example shows the hierarchy of a page:

```
wsadmin>Content.path()'Z6_00000000000000000000000A0 Z6_CGAH47L00GS790I6U1M1F020A3 Z6_CGAH47L00OJCC0I6U1NESJ2GK0'
```

This command returns a 'list' of object IDs of the nodes from the root node to your currently selected node. As you can see, it is not a 'true' list in the Jython sense, it is rather a string with entries separated by blanks. To get a list that is better readable, the command `split()` as shown in the following example:

```
wsadmin>for id in Content.path().split():
wsadmin>  Content.get(id, "un")
wsadmin>
'wps.content.root'
'ibm.portal.Home'
'ibm.portal.Home.Getting Started'
```

With this simple loop you can print out the whole hierarchy of nodes up to our currently selected 'Getting Started' page.

Creating a page:

To create a page as a child page to the currently selected page, use the following command:

```
wsadmin>Content.create("page", "Title of my first page", "html", "select")'
Z6_CGAH47L0082M00I6T9E0NL3001'
```

This command creates a page for HTML markup and returns its portal object ID. You can now start adding attributes or metadata to the page.

Adding a portlet to a page:

To add a portlet to the page that you created previously, use a the following command, for example:

```
wsadmin> myportlet = Portlet.find("portlet", "un", "wps.p.Information")
wsadmin> Layout.create("container", "horizontal", "select")'
Z7_CGAH47L008C970I6NA7U4300G2'
wsadmin> Layout.create("portlet", myportlet)'
Z7_CGAH47L008C970I6NA7U4300G1'
```

This adds the Information portlet to the page and places it in a horizontal container.

Getting help for a command:

The Portal Scripting Interface provides more commands and variants of commands than shown in the previous topics. To learn more about these commands and their syntax and function, and beans and parameters, refer to the integrated help of the Portal Scripting Interface.

About this task

Use the command `help()` and specify the command for which you want to get help. Start by entering the following command:

```
wsadmin>print Content.help("find")
```

This returns the following help output about how to use the help command:

```
> find <what> [<by> <value>] [select]
Finds a particular content node. The search scope is the subtree starting
at the current selection, or the whole tree if nothing is selected. Unlike
the 'search' command, this operation expects a single matching node as the
result of the search. An error is generated if there are multiple matches,
or if there is no match at all. If the keyword "select" is specified as the
last argument, the found node will be selected.
All other arguments are the same as for the 'search' command. See help on
'search-types' for the first argument. See help on 'search-criteria' for
the optional second and third arguments.
Example:> find page uniquename "My page"
Returns the id of a portal page node with the given unique name. If the page
with the given unique name does not exist, or if the node with the specified
unique name is not a page node, the command throws an exception.
```

To find out more detail about the search types, enter the following command:

```
wsadmin>print Content.help("search-types")
```

This returns the following help information about the search criteria:

```
> search <what>
> find <what>
The following is a list of supported keywords for content node types.
These keywords can be used as the first argument in 'search' and 'find'
operations.
- any, all
Any type of content node.
- label, labels
Only labels.
- page, pages, composition, compositions, comp
Only compositions, also called pages.
- url, urls, anyurl, allurls
Any type of URL node.
- iurl, iurls, internalurl, internalurls
Only internal URL nodes.
- xurl, xurls, externalurl, externalurls, eurl, eurls
Only external URL nodes.
```

To find out about the search criteria, use the following command:

```
wsadmin>print Content.help("search-criteria")
```

To find out which command the `Content` bean supports, enter the following command:

```
wsadmin>print Content.help()
```

This returns the following help output:

```
This bean provides access to the content hierarchy of the portal. The
content hierarchy consists of labels, pages, and links. Pages are also
called "compositions". Links can be internal or external URLs. Content
nodes can be accessed, created, and deleted with this bean. URLs of a
link can be modified with this bean.
The layout of a page is a component hierarchy of containers and controls.
That hierarchy must be accessed through the Layout bean rather than this
Content bean. When a page (composition) is selected in the Content bean,
the Layout bean can be used to access the component hierarchy of that page.
See the help for the Portal bean to learn about other available beans.
Invoke help with one of these methods as argument for further help:
help, resync, select, deselect, current, csn, root, parent, children, path,
index, details, get, set, nlsget, nlsset, nlsimport, urlget, urlset,
list, add, drop, empty, search, find, move, create, derive, delete,
pageget, pageset, parmget, parmset, transfer, deletecustomization
Other available help topics:
attribute-names, nls-files, list-names,
search-types, search-criteria, create-types
```

Logging out of the portal:

After you complete working with the Portal Scripting Interface, you log out and exit the interactive scripting console.

About this task

Enter the following commands:

```
wsadmin> Portal.logout()
wsadmin> exit
```

To find out more about the Portal Scripting Interface commands and usage refer to the other topics about the Portal Scripting Interface.

Working with the Portal Scripting Interface

Learn more about the different modes that you can use with the Portal Scripting Interface.

Prerequisite information

The Portal Scripting Interface provided by IBM WebSphere Portal Express is based on the wsadmin scripting tool that is provided by IBM WebSphere Application Server. Therefore, before you use the Portal Scripting Interface, familiarize yourself with how to use the WebSphere Application Server wsadmin tool.

Interactive mode

Use the interactive mode if you want to interact directly and dynamically with the portal to perform simple administrative tasks that should only be executed once. For example, the administrator wants to modify the permissions of a page for a certain principal, or the administrator wants to add a portlet to a page. Use the interactive mode if you do not intend to repeat the operation.

Before initiating a session in interactive mode, make sure that WebSphere Portal Express is running. The portal script client is located in the WebSphere Portal Express installation directory:

- Linux: *wp_profile_root*/PortalServer/bin
- IBM i: *wp_profile_root*/PortalServer/bin
- Windows: *wp_profile_root*\PortalServer\bin

Log in using the administrative user ID, and invoke the portal script client using the following commands:

- Linux: ./wpscript.sh
- IBM i: wpscript.sh
- Windows: wpscript.bat

The following procedures provide examples:

1. If WebSphere Application Server security is enabled, specify a user ID and password during login as shown in the following example:
   - Linux: ./wpscript.sh -port *port_number* -user *user_id* -password *password*
   - IBM i: wpscript.sh -port *port_number* -user *user_id* -password *password*
   - Windows: wpscript.bat -port *port_number* -user *user_id* -password *password*

The most basic parameters are explained briefly in the following table.

Table 148. Description of the basic parameters used with the `wpscript.bat|sh` task

Parameter	Description
`-lang`	Specifies the language of the script file, command, or an interactive shell. Specify one of the following values for the `-lang` parameter: • jacl • jython This parameter is optional and has no default value. This option overrides language determinations that are based on script file names, profile script file names, or the **`com.ibm.ws.scripting.defaultLang`** property. **Important:** If you do not specify the script language in the command line or as a parameter, and the wsadmin tool cannot determine the script language, an error occurs. If you do not specify the script language as the value for `-lang`, the wsadmin tool determines the script language as follows: • If you specify the `-f` *`script_file_name`* argument, the wsadmin tool determines the language from the name of the target script file. • If you specify the `-profile` *`profile_script_name`* argument, the wsadmin tool determines the language from the name of the profile script.
`-conntype`	The type of connection that should be established between scripting. Valid connection types include: • SOAP • RMI • NONE The default value is SOAP. This parameter is optional. Use the `-conntype NONE` option to run in local mode. The result is that the scripting client is not connected to a running server. If the connection type NONE is selected, the scripting beans are inactive and cannot be used for administration, with the exception of the help command.
`-port`	The connection port number. This parameter is optional. The port number depends on values chosen during installation. You can verify the value that is set for the **`WasSoapPort`** property in the `wkplc.properties` file found in the appropriate directory given here: • Linux: *wp_profile_root*`/ConfigEngine` • IBM i: *wp_profile_root*`/ConfigEngine` • Windows: *wp_profile_root*`\ConfigEngine` If you are running **`wpscript`** on a server that is part of a cell managed by a deployment manager, the *port_number* can vary depending on what ports are in use on the system when the deployment manager is created. To verify the value, check the setting for `SOAP_CONNECTOR_ADDRESS` in `serverindex.xml` located in the appropriate directory given here: • Linux: `dmgr_profile_root/config/cells/`*`cell_name`*`/nodes/`*`node_name`* • IBM i: `dmgr_profile_root/config/cells/`*`cell_name`*`/nodes/`*`node_name`* • Windows: `dmgr_profile_root\config\cells\`*`cell_name`*`\nodes\`*`node_name`*
`-user`	The user ID under which you establish the connection. This parameter can be mandatory, depending on your security configuration.
`-password`	The password for the user ID under which you establish the connection.

2. Log on to the portal using one of the following script commands:
   Jython: `Portal.login("`*`user_ID`*`", "`*`password`*`")`
   JACL: `$Portal login `*`user_ID password`*

3. Issue portal script commands as required.
4. After you have completed all tasks by the portal scripting interface, close and exit the script processor. All changes that you committed are applied to the portal configuration.

Script mode

Use the script mode to apply predefined changes to the configuration of a portal.

The wpscript tool executes a Jython or JACL script that contains the administrative operations. The scripting client inherits the script processor from wsadmin, so an administrator can exploit the Jython or JACL scripting language, in order to write re-usable, extendable administration scripts. This mode is typically preferred if reproducible administration tasks are created: For example, the administrator can write a script that produces a complete page subtree, and adds individual page layouts and portlets on each page.

Users who have access permission to perform XML configuration interface requests can change configurations of all resources. The Portal Scripting Interface is mostly consistent with the administration model that is exposed by the Portal user interface.

Before using script mode, make sure that WebSphere Portal Express is running and a portal script file is available. You must be logged in using the WebSphere administrative user ID. Use the following procedure:

1. Update the script file with the appropriate credentials, if required.
2. Use one of the following commands to launch the script processor tool:

   Jython: `wpscript.sh -port` *`port_number`* `-f` *`script_file_name.py`*

   JACL: `wpscript.sh -port` *`port_number`* `-f` *`script_file_name.jacl`*This initializes the interactive script environment of the portal JACL or Jython script processor.
3. Check the output from the script processor to ensure that no errors occurred during the execution of the script.

All changes committed by the script are immediately applied to the portal configuration.

The following script example creates a new page with the title **A page**. This page resides beneath the **Home** label. The page contains two portlets that are arranged horizontally.

Portal Jython script example

The following example is a Jython script file named `testme.py`:

```
# Scripting bean example: create a simple page (multi-column Layout)
#

# Procedure: create a multi-column page under the page that is currently
# selected, and place the given portlets into the layout.
#
# parameters:
#   name The name of the page
#   portlet_names A list of portlet names.
# returns:
#   oid The id of the page that has been created
def create_multi_col_page(name, portlet_names):
    thePage = Content.create("page", name, "html")
    Content.select(thePage)
```

```
    lyt0 = Layout.create("container", "horizontal", "select")
    for pn in portlet_names:
        pid = Portlet.find("portlet", "cn", pn)
        Layout.create("control", pid)

    return thePage


# main code starts here

# set User ID/ pwd for portal Login command
# Hint: User ID and passwords should normally not be placed inside a
# configuration script; better use property files or command line arguments
user = "user_ID"
pwd = "password"
Portal.login(user, pwd)

# determine and select the parent of the page to be created.
# In this example, This is the "Home" label.
Content.select(Content.find("all", "uniquename", "ibm.portal.Home")
# Invoke the page creation procedure. The label of the page is "My test page",
# portlets to be added are the reminder portlet and the welcome portlet.
newbie = create_multi_col_page("A Page", ["Reminder", "Welcome_to_WebSphere_Portal"])

print "ok, we are done."
```

Portal JACL script example

The following example is a JACL script file named `testme.jacl`:

```
# Scripting bean example: create a simple page (multi-column Layout)
#

# Procedure: create a multi-column page under the page that is currently
# selected, and place the given portlets into the layout.
#
# parameters:
#   name              The name of the page
#   portlet_names     A list of portlet names.
# returns:
#   oid               The id of the page that has been created
proc create_multi_col_page { name portlet_names } {
  global Content Layout Portlet
    set thePage [$Content create page $name html]
    $Content select $thePage
    set lyt0 [$Layout create container horizontal select]
    foreach pn $portlet_names {
      set pid [$Portlet find portlet cn $pn]
      $Layout create control $pid
    }
  return $thePage
}

# main code starts here

# set User ID/ pwd for portal Login command
# Hint: User ID and passwords should normally not be placed inside a
# configuration script; better use property files or command line arguments
set user user_ID
set pwd  password
$Portal login $user $pwd

# determine and select the parent of the page to be created.
# In this example, This is the "Home" label.
$Content select [$Content find all uniquename "ibm.portal.Home"]

# Invoke the page creation procedure. The label of the page is "My test page",
```

```
# portlets to be added are the reminder portlet, and the welcome portlet.
set newbie [create_multi_col_page "A Page" { "Reminder" "Welcome_to_WebSphere_Portal" } ]

puts "ok, we are done."
```

The scripts can receive parametric information externally, by using one of the following features:

- Profiles.
- Command line arguments.

Scripts can access command line arguments with the following variables:

argc Use this variable in JACL scripts to specify the number of command line arguments.

argv Use this variable in Jython and JACL scripts to specify the command line arguments.

Jython: In the preceding example script, `testme.py`, delete the following statements:

```
user = "portaladmin"
pwd = "adminpwd"
```

and replace them with the following statements:

```
if len(sys.argv) != 2:
   print "invocation syntax: wpscript testme.py <user> <pwd>"
   sys.exit(1)

user = argv[0]
pwd = argv[1]
```

JACL: In the preceding example script, `testme.jacl`, delete the following statements:

```
set user portaladmin
set pwd  adminpwd
```

and replace them with the following statements:

```
if { $argc != 2 }  {
    puts "invocation syntax: wpscript testme.jacl <user> <pwd>"
   exit
  }
  set user [lindex $argv 0]
  set pwd [lindex $argv 1]
```

The security-sensitive username and password are removed from the script. The modified code expects the user ID and password to be specified as command line arguments, for example:

Jython: `wpscript.sh -port` *port_number* `-f testme.py` *user_IDpassword*
JACL: `wpscript.sh -port` *port_number* `-f testme.jacl` *user_ID password*

Run scripting commands in a profile

A profile is a script that runs before the main script, or before entering interactive mode. Profiles can be used to set up environment specific behavior or user specific data. Profiles are specified when invoking wpscript, using the `-profile` parameter. For example, the `login` command can be placed in a profile.

Jython profile script example

The following example is a Jython profile script named mylogin.py:

```
# scripting profile
# contains log-in procedure on portal with disabled security
if len(sys.argv) != 2:
   print "invocation syntax: wpscript -f testme.jacl -profile
mylogin.py user_ID password"
   sys.exit(1)

user = argv[0]
pwd = argv[1]
Portal.login(user, pwd)
```

Remove or comment out the following statements in the testme.py script file:

```
if len(sys.argv) != 2:
   print "invocation syntax: wpscript testme.py user_ID password"
   sys.exit(1)

user = argv[0]
pwd = argv[1]
Portal.login(user, pwd)
```

To invoke mylogin.py, enter the following command: wpscript.sh -port *port_number* -profile mylogin.py -f testme.py *user_ID password*

JACL profile script example

The following example is a JACL profile script named mylogin.jacl:

```
# scripting profile
  # contains log-in procedure on portal with disabled security
  if { $argc != 2 }  {
    puts "invocation syntax: wpscript -f testme.jacl -profile mylogin.jacl user_ID password"
   exit
  }
  set user [lindex $argv 0]
  set pwd [lindex $argv 1]
  $Portal login $user $pwd
```

Remove or comment out the following statements in the testme.jacl script file:

```
  if { $argc != 2 }  {
    puts "invocation syntax: wpscript testme.jacl user_ID password"
    exit
  }
  set user [lindex $argv 0]
  set pwd [lindex $argv 1]
  $Portal login $user $pwd
```

To invoke mylogin.jacl, enter the following command: wpscript.sh -port *port_number* -profile mylogin.jacl -f testme.jacl *user_ID password*

The benefit of this change is that the environment-specific login procedure is removed from the administration script. For systems with enabled WebSphere Application Server security, the login procedure is:

Jython scripts

```
# scripting profile
# contains log-in procedure on portal with enabled security
Portal.login()
```

JACL scripts

```
# scripting profile
# contains log-in procedure on portal with enabled security
$Portal login
```

Related concepts:

"Configuring portal behavior" on page 248
Configure various options related to your portal.

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related reference:

"Scripting for static pages" on page 1875
You can work with static portal pages by using the Portal Scripting Interface, which enables you to use administration function through the Jacl scripting language. Get familiar with the scripting commands for working with static pages.

"Portal Scripting Interface" on page 1113
You can use the Portal Scripting Interface to configure your portal by running scripts from a command line.

"Command reference for the Portal Scripting Interface"
The IBM WebSphere Portal Express Scripting Interface component provides a scripting interface for the administration functions.

Related information:

 http://www.ibm.com/software/webservers/appserv/was/library/

Command reference for the Portal Scripting Interface

The IBM WebSphere Portal Express Scripting Interface component provides a scripting interface for the administration functions.

With the Portal Scripting Interface, you can access the portal with any portal user ID and work within the access rights of that user ID.

Jython or Jacl are the two scripting languages that you can use for the scripting syntax and that are supported by the wsadmin tool of the WebSphere Application Server.

Jython

Jython is a general-purpose high-level programming language. It uses code indentation as block delimiters.

- The hash character (#) starts a comment that extends to the end of the line.
- By default, each line is interpreted as one statement.
- You can write multiple statements on one line by separating the statements with semicolons.
- Jython is case-sensitive.

Example:

```
# here is a comment
single_statement(with_arguments)
first_statement_in_line() ; second_statement()
outer statement [first inner] [second inner statement]
```

A variable can contain an object. In a statement, an object method starts with the object followed by a dot (.), the method name, and arguments that are passed to the method in parentheses ().

Example:

```
# variable 'Object' holds the object to invoke
# invoke the double argument version of the method
Object.method(arg1, arg2)

# and now the single argument version
Object.method(arg)

# there may be a version with three arguments
Object.method(arg1, arg2, arg3)

# invoke the single argument version
# the single argument is provided as a nested statement
# the nested statement invokes the double argument version
Object.method(Object.method(argInner1, argInner2)
```

Jacl

Jacl is an interpreted language without strong typing. It is a procedural language with some object-oriented concepts that are used by the scripting component.

- The number sign character (#) starts a comment that extends to the end of the line.
- By default, each line is interpreted as one statement.
- You can write multiple statements on one line by separating the statements with semicolons.
- You can nest statements by using brackets [] . The brackets are interpreted like back quotation marks in most AIX HP-UX Linux Solaris shells. The statement within the brackets is run, and its result is substituted in place of the bracketed statement before the surrounding statement is interpreted.
- Jacl is case-sensitive.

Example:

```
# here is a comment
single statement with arguments
first statement in line ; second statement
outer statement [first inner] [second inner statement]
```

The value of a Jacl variable is accessed by placing a $ in front of the variable name. A variable can contain an object. An object method is started by using the object as the first part of a statement, followed by the method name, and any arguments that are passed to the method. Since there is no strong typing, a method can be overloaded only by varying the number of arguments.

Example:

```
# variable 'Object' holds the object to invoke
# invoke the double argument version of the method
$Object method arg1 arg2

# and now the single argument version
$Object method arg

# there may be a version with three arguments
$Object method arg1 arg2 arg3

# invoke the single argument version
# the single argument is provided as a nested statement
# the nested statement invokes the double argument version
$Object method [$Object method argInner1 argInner2]
```

"Examples" on page 1173
The following are examples for deleting portlets and adding portlets.

"Troubleshooting" on page 1174
The following solutions help solve the troubleshooting issues.

"Property file format" on page 1174
You can provide locale-specific attributes for a set of locales in a Java property file. The generic format of Java property files is described in the Java API documentation for method load in class `java.util.Properties`. The description here covers the particular properties that are interpreted when locale-specific attributes are loaded from a portal script.

"Index paths" on page 1175
Index paths are used to refer to components in the component hierarchy. They are based on the index or position of a component in the surrounding container. An index path is a multi-dimensional index of a component, where the number of dimensions is equal to the depth of the component in the tree. Index paths are absolute or relative, depending on whether there is a leading slash. Absolute paths start with a leading slash and are resolved from the root component. Relative paths start with a number and are resolved from the selected component. Trailing slashes are irrelevant.

Related reference:

"Portal Scripting Interface and project support" on page 1260
With the Portal Scripting Interface, you can create Jacl or Jython scripts to automate the management of projects.

"Portal Scripting Interface and web content libraries" on page 1270
With the Portal Scripting Interface, you can create Jacl or Jython scripts to automate the management of web content libraries. Using the DocumentLibrary bean with the Portal Scripting Interface, you can create and delete libraries, retrieve a list of libraries, and retrieve library attributes.

Script beans:

The portal scripting component adds Script beans to the wsadmin tool. These Script beans are objects with methods that work on the portal data.

Portal objects are not represented by Jython or Jacl objects. Jython or Jacl object that represents a particular page or an individual portlet is not present. Rather, a fixed number of script beans provide access to specific areas of the portal data.

The available beans are

- Portal
- Content
- Layout
- Portlet
- Look
- Access
- PacList
- Application
- ArchivedApplication
- ApplicationCategory
- TemplateCategory
- Publish

Most method names are single English words, such as `get` or `search` or `parent`. All method names must be written in lowercase. Each bean has a help method. If started with a method name, it prints help for that method in the bean. If started without an argument, it prints general help for that bean, including a list of method names and other help topics. The general help message of the Portal bean includes an overview of the available beans and their responsibilities.

Jython example:

```
# get help - for the completely lost
Portal.help()

# get help on a particular bean
Portlet.help()
Access.help()

# get help on a method of a bean
Portal.help("login")
Layout.help("select")

# get help on an extended help topic of a bean
Content.help("search-criteria")
```

Jacl example:

```
# get help - for the completely lost
$Portal help

# get help on a particular bean
$Portlet help
$Access help

# get help on a method of a bean
$Portal help login
$Layout help select

# get help on an extended help topic of a bean
$Content help search-criteria
```

Portal objects:

Most portal objects are represented in the script by an object identifier string, which is based on the object ID in the portal. For example: `_6_00KJL57F9D02H456_A`.

These IDs are expected by methods as arguments and returned as results. Since an ID never contains white spaces or characters that would be misinterpreted by Jacl, it is a convenient handle for a portal object. If a method returns several objects, the IDs are separated by white spaces. The results are then used directly as a Jacl list.

Note: In Jython, you use the method split() on the result string to create a list.

Unlike with the GUI, where geometric arrangement and a locale-specific title provide information about an object, the IDs used in the script are unintelligible to the user. Most of the Script beans therefore provide a `details` method that prints information about an object. The `details` method of a bean works only for the objects that are handled by that bean. For example, the content bean cannot provide details about IDs returned by the layout bean.

Jython example:

```
# 'search' returns a list of objects
for child in Content.search("all").split():
  # details get printed, they are not returned as a result
  Content.details(child)
}
```

Jacl example:

```
# 'search' returns a list of objects
foreach child [$Content search all] {
  # details get printed, they are not returned as a result
  $Content details $child
}
```

The scripting component tries to generate a common name for the portal objects. The common name is based on a global unique name, an object name, or a title that is assigned to the object. The common name never contains white space, special characters, or characters outside of the US-ASCII range. Hence, even a terminal window that does not support national character sets can display the common name. If suitable input data is available, the generated common name might provide an indication of what an object represents.

Tree navigation:

The Content, Layout, and Portlet beans each represent a tree hierarchy. The basic navigation methods are the same for all three. A tree bean provides methods to access the root node to look up the parent and children of a node, and to maintain a cursor that points to a selected node in the tree.

The code examples use the Content bean, but you can do the same operations on the other tree beans as well. The command root returns the ID of the root node, as a fixed starting point for navigation.

Jython: `Content.root()`

Jacl: `$Content root`

You can select a node by its ID by using the `select` command. You can clear the current selection by using `deselect` or by using `select` without an argument. You can return the ID of the selected node by using the `current` command. For interactive use, `csn` is an alias for `current`.

Jython example:

```
Content.select ID
Content.deselect
Content.current
Content.csn

# example: select the root node
Content.select(Content.root())
```

Jacl example:

```
$Content select ID
$Content deselect
$Content current
$Content csn

# example: select the root node
$Content select [$Content root]
```

The `path` command returns a list of all IDs from the root to the currently selected node. In a similar way, the `children` command returns the children of the selected node. You can obtain the ID of the parent of the selected node by using `parent`.

Jython example:

```
Content.path
Content.parent
Content.children

# example: select a node and print its children
Content.select(node_ID)
for child in Content.children().split():
    print child
```

Jacl example:

```
$Content path
$Content parent
$Content children

# example: select a node and print its children
$Content select node_ID
foreach child [$Content children] { puts "$child" }
```

You can also use the commands `path`, `parent`, and `children` with an explicit ID instead of implicitly referring to the currently selected node.

Jython example:

```
Content.path(ID)
Content.parent(ID)
Content.children(ID)
```

Jacl example:

```
$Content path ID
$Content parent ID
$Content children ID
```

For simplicity, there are dedicated select commands for the root node and for the parent of the currently selected node. In the following example, the first argument is a dummy that is used to distinguish the method from the select with an ID argument. The dummy argument is not interpreted. The second argument is a keyword, which is not case-sensitive. Alternative, shorter keywords are documented in the bean help.

Jython example:

```
Content.select("the", "root")
Content.select("the", "parent")
```

Jacl example:

```
$Content select the root
$Content select the parent
```

Search:

All beans with tree navigation support identical commands for searching, but the available search criteria are different for each bean.

The generic examples use the Content bean. Searches in trees are scoped. The search scope is the subtree under the selected node, including the selected node itself. If nothing is selected, the search scope is the full tree that starts at the root.

There are two different commands for searching:`search` and `find.search` returns a list of matches, whereas `find` succeeds only if there is a single, unique match for the search. It fails if there is more than one match, or no match at all. `find` is used in cases where a script must end if the search result is not a unique match. If the keyword `select` is passed to `find`, the search result becomes the selected node.

Jython example:

```
Content.search(type)
Content.search(type), "by", (value)

Content.find(type)
Content.find(type), "by", (value)

Content.find(type, "select")
Content.find(type, "by", (value, "select")
```

Jacl example:

```
$Content search type
$Content search type by value

$Content find type
$Content find type by value

$Content find type select
$Content find type by value select
```

The first argument for all searches is the type of the nodes to look for. The type is specified by a keyword, which is not case-sensitive. The available types and corresponding keywords depend on the bean. In all beans, the keywords `all` and `any` are used to search regardless of the type. There is a dedicated help topic for the search types.

Jython example:

```
# example: return all nodes in the search scope
Content.search all

# example: get help on the available type keywords
Content.help search-types
```

Jacl example:

```
# example: return all nodes in the search scope
$Content search all

# example: get help on the available type keywords
$Content help search-types
```

You can combine the type selection with an extra search criteria, which is specified by a keyword (`by`) and a value to match against (*value*). The available search criteria and corresponding keywords depend on the bean. There is a dedicated help topic for the search criteria.

Jython example:

```
# example: get help on the available by keywords
Content.help("search-criteria")
```

Jacl example:

```
# example: get help on the available by keywords
$Content help search-criteria
```

The following are common search criteria. Alternative, shorter keywords are described in the help text on search criteria of the respective bean.

Table 149. A description of the common search criteria

Value	Description
id	The value is an ID. The search is for the object with that ID.
uniquename	The value is a string. The search is for the object with the string as its unique name.
commonnamehas	The value is a string. The search is for objects with the string as a substring in their common name. Comparison is not case-sensitive.
commonnameis	The value is a string. The search is for objects with the string as their common name. Comparison is case-sensitive.

Jython example:

```
# example: find and select by unique name
Content.find("any", "uniquename", "ibm.portal.Portlets", "select")
```

Jacl example:

```
# example: find and select by unique name
$Content find any uniquename "ibm.portal.Portlets" select
```

Attributes:

All beans use similar commands to query and modify attributes. Attributes are identified by a name, such as uniquename, title, or markup.

There are different commands for the different types of attributes. Similar to the `details` command in Portal objects, the commands must start on the bean responsible for the object type. For example, attributes of content nodes are accessed only through the Content bean, not through the Layout or any other bean.

The set of attributes that is supported for an object depends on the type of the object. Many attributes are read-only. Attribute values are always represented as strings in the scripting language. They are mapped from and to portal data types by the respective script bean. Exceptions are triggered if the mapping fails.

"Plain attributes" on page 1135
Plain attributes have a single value that is queried by using the `get` command. The object for which to query the attribute is specified by an ID, and the attribute is specified by name.

"List valued attributes" on page 1136
List valued attributes can have multiple values. They are queried by using the `list` command, which returns all values, which are separated by white space.

"Locale-specific attributes" on page 1137
Locale-specific attributes have different values for different languages and countries. They can be queried by using the `nlsget` command, where nls stands for National Language Support.

Plain attributes:

Plain attributes have a single value that is queried by using the `get` command. The object for which to query the attribute is specified by an ID, and the attribute is specified by name.

If the attribute is not read-only, the value is set by using the `set` command, which expects the new value as the last argument. If the bean supports a current selection, the ID is omitted for both commands to refer to the selected object.

Jython example:

```
Content.get(ID, attribute)
Content.set(ID attribute value)

# only for beans with a current selection
Content.get(attribute)
Content.set(attribute value)

# example: get unique name of a content node
Content.get(ID, "uniquename")

# example: get type of the selected content node
Content.get("type")

# example: set theme of a content node
themeid = Look.find("theme", "commonnameis", "Science")
Content.set(ID, "theme", themeid)
```

Jacl example:

```
$Content get ID attribute
$Content set ID attribute value

# only for beans with a current selection
$Content get attribute
$Content set attribute value

# example: get unique name of a content node
$Content get ID uniquename

# example: get type of the selected content node
$Content get type

# example: set theme of a content node
set themeid [$Look find theme commonnameis "Science"]
$Content set ID theme $themeid
```

The following are standard attribute names available for all objects. Names for more attributes of individual portal object types are documented with the respective bean. Alternative or shorter names are documented in the bean help.

Table 150. A description of the standard attributes names available for all objects

Value	Description
id	The identifier of the object.
type	The type of the object.
uniquename	The unique name of the object if it is assigned.
commonname	The common name of the object, if it is generated.

List valued attributes:

List valued attributes can have multiple values. They are queried by using the `list` command, which returns all values, which are separated by white space.

The object is specified by an ID and the attribute of the object, which is queried, is specified by name. If the bean supports a current selection, the ID is omitted to refer to the selected object.

You can modify list valued attributes by adding or removing a particular value, or by removing all values from the list. The respective commands are `add`, `drop`, and `empty`. With all commands, the object is specified by ID and the attribute by name. The `add` and `drop` commands also require the value to be added or removed.

These commands are appropriate if the values do not contain white space and the order of the elements is not important. Currently, all list valued attributes satisfy these restrictions.

You cannot modify all list valued attributes by all of these commands. For example, some lists might not be empty, in which case the `empty` command is not available. However, if an operation is supported for an attribute, the operation uses the command as described here. List valued attributes might also change as a side effect of other operations. For example, if a title is set for a previously undefined locale, the new locale shows up in the list of locales. For more information, see *Local specific attributes* for details on titles and locales.

Jython example:

```
Content.list(ID, attribute)
Content.add(ID, list, value)
Content.drop(ID, list, value)
Content.empty(ID, list)

# only for beans with a current selection
Content.list(attribute)
Content.add(list, value)
Content.drop(list, value)
Content.empty(list)

# example: add a new markup for the selected node
Content.add("markup", "wml")

# example: drop the american locale for the given node
Content.drop(node_ID, "locale", "en_US")

# example: drop all locales for the given node
Content.empty(node_ID, "locale")
```

Jacl example:

```
$Content list ID attribute
$Content add ID list value
$Content drop ID list value
$Content empty ID list

# only for beans with a current selection
$Content list attribute
$Content add list value
$Content drop list value
$Content empty list

# example: add a new markup for the selected node
$Content add markup wml
```

```
# example: drop the american locale for the given node
$Content drop node_ID locale en_US

# example: drop all locales for the given node
$Content empty node_ID locale
```

Related concepts:

"Locale-specific attributes"
Locale-specific attributes have different values for different languages and countries. They can be queried by using the `nlsget` command, where nls stands for National Language Support.

Locale-specific attributes:

Locale-specific attributes have different values for different languages and countries. They can be queried by using the `nlsget` command, where nls stands for National Language Support.

The object is specified by an ID and the object's attribute, which is queried is specified by name. If the bean supports a current selection, the ID is omitted to refer to the selected object. The following are the usual locale-specific attributes.

- `title`
- `description`
- `shorttitle`
- `keywords`

For the Content bean, only the locale-specific attributes `title` and `description` exist. For the Portlet bean, all four of the attributes exist.

The language and country are given as a locale, which consists of a language identifier, an optional country identifier, and an optional variant. Language and country are specified by standard two letter abbreviations, the language in lowercase and the country in uppercase letters. The components are separated by underscore characters. Here are some example locales:

Table 151. Example locales and their description

Locale	Description
en	General English
en_US	American English
de_CH	Swiss German
pt_BR	Brazilian Portuguese

The list valued attributes *locales* holds all locales for which a locale-specific attribute might be defined. However, all locale-specific attributes are optional for all locales. No fallback algorithm that would, for example, return the general Portuguese value if the Brazilian Portuguese value is not set is present. Such fallback algorithms are used when pages are assembled by the portal, but not for the administrative access that is provided by the portal scripting component.

Locale-specific attributes that are not read-only are set by using the `nlsset` command. Specify the new value of the attribute as the last argument in the arguments for the `nlsget` command. By setting an attribute for a locale that was not used before, the new locale is added to the list valued attribute *locales*.

Jython example:

```
Content.nlsget(ID, attribute, locale)
Content.nlsset(ID, attribute, locale, value)

Content.nlsget(attribute, locale)
Content.nlsset(attribute, locale, value)

# example: get american title of a specific content node
Content.nlsget(node_ID, "title", "en_US")

# example: set german description of current selection
Content.nlsset("description", "de_DE", "Kurze Beschreibung")

# example: set general english title of a specific node
Content.nlsset(node_ID, "title", "en", "English Title")
```

Jacl example:

```
$Content nlsget ID attribute locale
$Content nlsset ID attribute locale value

$Content nlsget attribute locale
$Content nlsset attribute locale value

# example: get american title of a specific content node
$Content nlsget node_ID title en_US

# example: set german description of current selection
$Content nlsset description de_DE "Kurze Beschreibung"

# example: set general english title of a specific node
$Content nlsset node_ID title en "English Title"
```

As locale-specific attributes are often translated independently from script development, the `nlsimport` command is used to read a separate property file that defines attribute values for a set of locales. By specifying an appropriate prefix, you can load values from the same property files that are used by the XML configuration interface (XMLAccess).

Jython example:

```
Content.nlsimport(ID, file_name)
Content.nlsimport(ID, file_name, prefix)
```

Jacl example:

```
$Content nlsimport ID file_name
$Content nlsimport ID file_name prefix
```

To delete all locale-specific attributes before a set of values is imported, empty the list valued attribute *locales* as described in *List valued attributes*.

Organization:

For some beans, in particular the Content and Layout beans, the order of nodes is significant. In tree beans, the parent relationship of the nodes defines the hierarchy. Only nodes with the same parent node are in a particular order in trees.

Sequence

The `move` command is used to reorder nodes. The order of nodes is tracked as a non-negative integer position assigned to each node, with the first node at position

0. For tree beans, the positions compare only among the children of a common parent node. You can query the position as a read-only plain attribute.

The order of nodes is changed by the `move` command. It is used on a single node to assign an absolute position or to displace by a distance. The positions of the other affected nodes are updated automatically. In both cases, the new position of the node always remains within bounds. The position or displacement argument is adjusted by the bean.

The `move` command expects the ID of the node to be moved, a keyword that indicates whether the change is absolute or relative, and the new position or distance. If the bean supports a current selection, the ID is omitted to move the selected object.

Jython example:

```
Layout.move(ID, "to", position)
Layout.move(ID, "by", distance)

# only for beans with a current selection
Layout.move("to", position)
Layout.move("by", distance)

# example: move selected node to the first position
Layout.move("to", 0)

# example: move given node one down in the list
# if it already is the last node, do nothing
Layout.move(node_ID, "by", 1)

# example: move selected node 4 up in the list
# if it is at position 0 to 4, it becomes the new head
Layout.move("by", -4)

# example: move given node to the last position
# negative absolute positions are interpreted as max int
Layout.move(node_ID, "to", -1)
```

Jacl example:

```
$Layout move ID to position
$Layout move ID by distance

# only for beans with a current selection
$Layout move to position
$Layout move by distance

# example: move selected node to the first position
$Layout move to 0

# example: move given node one down in the list
# if it already is the last node, do nothing
$Layout move node_ID by 1

# example: move selected node 4 up in the list
# if it is at position 0 to 4, it becomes the new head
$Layout move by -4

# example: move given node to the last position
# negative absolute positions are interpreted as max int
$Layout move node_ID to -1
```

Hierarchy

In some tree beans, you can change the parent of a node in the tree, moving the whole subtree under that node to another part of the tree.

This change of the parent of a node in the tree is achieved by the commands `transfer` or `adopt`. With `transfer`, the ID of the node and new parent are specified explicitly, whereas `adopt` uses the selected layout mode as the new parent. To improve readability and avoid confusion about the interpretation of the two IDs, the arguments of the transfer command are separated by the keyword `"to"`.

Jython example:

```
Content.transfer(child_ID, "to", parent_ID)
Layout.transfer(child_ID, "to", parent_ID)
Layout.adopt(child_ID)

# example: move a layout node under the root component of the page
Layout.select("the", "root")
Layout.adopt(node_id)
```

Jacl example:

```
$Content transfer child_ID to parent_ID
$Layout transfer child_ID to parent_ID
$Layout adopt child_ID

# example: move a layout node under the root component of the page
$Layout select the root
$Layout adopt node_ID
```

Content hierarchy accessed through Content bean:

The content hierarchy is a tree of content nodes. Content nodes can be labels, compositions, and links. Compositions are also called pages. Links can be internal or external. In the GUI, links represent the nodes in the Favorites list. Internal links point to a portal page, external links can point to any URL.

The content hierarchy is accessed and modified by using the Content bean, referenced as `$Content` in Jacl. The Content bean offers the following functions:

- Methods that allow to browse in the content tree hierarchy. The navigation method for the Content bean is a tree bean. For more information, see *Tree navigation*
- Methods that allow to locate a content node, or to search for particular content nodes. For more information, see *Search* .
- Methods for getting and setting attributes or metadata. The following attribute types are supported by the Content bean.
  - Plain attributes
  - List valued attributes
  - Locale-specific attributes
  - URL attributes
  - Metadata attributes
- Methods to create or delete pages, labels, or URL links, or to derive pages. For more information, see *Lifecycle* .
- Methods to move pages, labels, or URL links. The sequence of nodes can be modified as described in *Sequence*. For more information, see *Organization*. It is not possible to reorganize the content hierarchy..

"Search"
The generic search syntax is documented in Search. The Content bean supports the default search criteria, and the following keywords for node types in searches. Alternative, shorter keywords are documented in the bean help.

"Plain attributes" on page 1142
In addition to the default attributes, content nodes have the following attributes.

"List valued attributes" on page 1143
The content nodes have three list valued attributes, `locales`, `markups`, and `allowedportlets`. See the bean help for alternative, shorter names for these lists.

"Locale-specific attributes" on page 1144
The content nodes have two locale-specific attributes, `title` and `description`. See the bean help for alternative, shorter names for these attributes. The title must be defined for each locale, though you can set it to the empty string. A new locale is defined by setting the title for it.

"URL attributes" on page 1145
For a URL content node, the `urlget` command obtains the URLs. It requires the markup name as an argument. If an ID is given, the requested URL of that content URL node is returned. If the ID is omitted, the requested URL of the currently selected URL node is returned.

"Metadata attributes" on page 1145
Content nodes can own metadata, which are name-value pairs of data that is associated with the content node. Metadata are used by the portal, for example to set display attributes, or by the user. However, you must ensure that none of the metadata information that is set by the portal is overridden.

"Lifecycle" on page 1146
The `create` command creates a new content node. The `derive` command creates a new content node for a page that is derived from another page. The `delete` command removes a content node.

Related concepts:

"Tree navigation" on page 1131
The Content, Layout, and Portlet beans each represent a tree hierarchy. The basic navigation methods are the same for all three. A tree bean provides methods to access the root node to look up the parent and children of a node, and to maintain a cursor that points to a selected node in the tree.

"Organization" on page 1138
For some beans, in particular the Content and Layout beans, the order of nodes is significant. In tree beans, the parent relationship of the nodes defines the hierarchy. Only nodes with the same parent node are in a particular order in trees.

Search:

The generic search syntax is documented in Search. The Content bean supports the default search criteria, and the following keywords for node types in searches. Alternative, shorter keywords are documented in the bean help.

- `label`
- `composition or page`
- `internalurl`
- `externalurl`
- `url (either internal or external)`
- `all or any`

Jython example:

```
Content.help("search-types")

# example: search all content nodes
Content.search("all")

# example: find and select page by unique name
Content.find("page", "uniquename", "ibm.portal.Portlets", "select")

# example: find all pages under label "Administration"
Content.find("label", "uniquename", "ibm.portal.Administration", "select")
Content.search("composition")
```

Jacl example:

```
$Content help search-types

# example: search all content nodes
$Content search all

# example: find and select page by unique name
$Content find page uniquename "ibm.portal.Portlets" select

# example: find all pages under label "Administration"
$Content find label uniquename ibm.portal.Administration select
$Content search composition
```

Plain attributes:

In addition to the default attributes, content nodes have the following attributes.

Table 152. Description of content node attributes

Attribute	Description
`position`	The numeric position among the siblings, zero-based.
`themeid`	The identifier of the theme for the content node.
`themename`	The name of the theme for the content node.
`allportlets`	A flag that indicates whether all portlets are allowed for the page or not. If this flag is set true, the list of allowed portlets is ignored. Refer to *List valued attributes*.

The `themeid` attribute is writable. The `themename` attribute is not writable, but the value depends on the `themeid` attribute. The `position` attribute is not writable either, but the value depends on the organization of the content tree.

To get attributes of a static page content node, run the following command

Jython:

```
Content.get(oid, attribute, markup)
```

Jacl:

```
$Content get oid attribute markup
```

Valid attributes are as follows:

filename
Gets the file name of the static page layout file that is contained in the ZIP archive.

displayoption
Specifies markup languages such as HTML.

Jython example: `Content.get(6_CGAH47L00G2N802TJFV58Q3000, filename, html)`
Jacl example: `$Content get 6_CGAH47L00G2N802TJFV58Q3000 filename html`

The preceding example returns the file name of the entry point for the page display.

To set attributes for a static page, run the following command:

Jython example:
`Content.set(`*`oid`*`, `*`attribute`*`, `*`value`*`, `*`markup`*`)`

Jacl example:
`$Content set `*`oid attribute value markup`*

Valid attributes are **filename** and **displayoption**.

filename
Gets the file name of the static page layout file that is contained in the ZIP archive. For example, to set the entry point to display anotherindex.html for the specified markup run the following command:

Jython: `Content.set(6_CGAH47L00G2N802TJFV58Q3000, filename, anotherindex.html, htm`
Jacl: `$Content set 6_CGAH47L00G2N802TJFV58Q3000 filename anotherindex.html html`

displayoption
Specifies markup languages such as HTML. Valid options for **displayoption** are inline, iframe, and ajax. For example, to set the display option to iframe for the specified markup run the following command:

Jython: `Content.set(6_CGAH47L00G2N802TJFV58Q3000, displayoption, iframe, html)`
Jacl: `$Content set 6_CGAH47L00G2N802TJFV58Q3000 displayoption iframe html`

Related concepts:
"List valued attributes"
The content nodes have three list valued attributes, locales, markups, and allowedportlets. See the bean help for alternative, shorter names for these lists.

List valued attributes:

The content nodes have three list valued attributes, locales, markups, and allowedportlets. See the bean help for alternative, shorter names for these lists.

The locales list holds the locales for which locale-specific attributes are defined. For more information, see *Locale-specific attributes*. The commands list, drop, and empty are available for locales. New values can be added by setting a title for the respective locales.

The markups list holds the markups that are supported by the content node. The commands list, add, and drop are available for markups. At least one markup must be supported.

The allowedportlets list holds the portlets that are registered as allowed portlets for that page. The total list of portlets that can be used for the page is this allowedportlets list, plus the lists of allowed portlets of all parent pages.

Jython example:

```
# example for manipulating locales of a content node:
# select node, list locales, remove and re-create a locale
# "locale" is an alternative name for the "locales" list
Content.select(ID)
Content.list("locales")
Content.drop("locale", "en")
Content.nlsset("title", "en", "New English Title")

# example for manipulating markups of a content node:
# select node, list markups, replace wml by chtml
# "markup" is an alternative name for the "markups" list
Content.select(ID)
Content.list("markups")
Content.add("markup", "chtml")
Content.drop("markup", "wml")
Content.list("allowedportlets")
```

Jacl example:

```
# example for manipulating locales of a content node:
# select node, list locales, remove and re-create a locale
# "locale" is an alternative name for the "locales" list
$Content select ID
$Content list locales
$Content drop locale en
$Content nlsset title en "New English Title"

# example for manipulating markups of a content node:
# select node, list markups, replace wml by chtml
# "markup" is an alternative name for the "markups" list
$Content select ID
$Content list markups
$Content add markup chtml
$Content drop markup wml
$Content list allowedportlets
```

Related concepts:

"Locale-specific attributes"
The content nodes have two locale-specific attributes, title and description. See the bean help for alternative, shorter names for these attributes. The title must be defined for each locale, though you can set it to the empty string. A new locale is defined by setting the title for it.

Locale-specific attributes:

The content nodes have two locale-specific attributes, title and description. See the bean help for alternative, shorter names for these attributes. The title must be defined for each locale, though you can set it to the empty string. A new locale is defined by setting the title for it.

Jython example:

```
# example for manipulating locales of a content node:
# select node, remove all, import, add one manually
Content.select(ID)
Content.empty("locales")
Content.nlsimport("nls/content.nls", "page.visualization")
Content.nlsset("title", "en_GB", "Visualisation")
Content.nlsset("description", "en_GB", "A page for...")
```

Jacl example:

```
# example for manipulating locales of a content node:
# select node, remove all, import, add one manually
$Content select ID
$Content empty locales
$Content nlsimport nls/content.nls page.visualization
$Content nlsset title en_GB "Visualisation"
$Content nlsset description en_GB "A page for..."
```

URL attributes:

For a URL content node, the `urlget` command obtains the URLs. It requires the markup name as an argument. If an ID is given, the requested URL of that content URL node is returned. If the ID is omitted, the requested URL of the currently selected URL node is returned.

Jython example:

```
Content.urlget(markup)
Content.urlget(ID, markup)
# example: get WML URL of a specific content URL node
Content.urlget(node_ID, "wml")
```

Jacl example:

```
$Content urlget markup
$Content urlget ID markup

# example: get WML URL of a specific content URL node
$Content urlget node_ID wml
```

Metadata attributes:

Content nodes can own metadata, which are name-value pairs of data that is associated with the content node. Metadata are used by the portal, for example to set display attributes, or by the user. However, you must ensure that none of the metadata information that is set by the portal is overridden.

Jython example:

```
Content.parmget(ID, name)
Content.parmset(ID, name, value)
Content.drop(ID, "parm", name)
Content.list(ID, "parm")

# only for beans with a current selection
Content.parmget(name)
Content.parmset(name, value)
Content.drop("parm", name)
Content.list("parm")

# example: set the metadata for an instance property named #
"MyUserData" on the selected node
Content.parmset("MyUserData", "A_User_Value")

# example: get the metadata for an instance property named #
"MyUserData" (should return "A_User_Value")
print Content.parmget("MyUserData")

# example: list all metadata names
for pname in Content.list("parm").split():
    print pname
```

```
#example: Drop the metadata with the name "MyUserData"
Content.drop("parm", "MyUserData")
Organization
```

Jacl example:

```
$Content parmget ID name
$Content parmset ID name value
$Content drop ID parm name
$Content list ID parm

# only for beans with a current selection
$Content parmget name
$Content parmset name value
$Content drop parm name
$Content list parm

# example: set the metadata for an instance property named
# "MyUserData" on the selected node
$Content parmset MyUserData A_User_Value

# example: get the metadata for an instance property named
# "MyUserData" (should return "A_User_Value")
puts "[$Content parmget MyUserData]"

# example: list all metadata names
foreach pname [$Content list parm] {
  puts "$pname"
}

#example: Drop the metadata with the name "MyUserData"
$Content drop parm "MyUserData"
```

Lifecycle:

The `create` command creates a new content node. The `derive` command creates a new content node for a page that is derived from another page. The `delete` command removes a content node.

When you create a new content node, the parent for the new node must be selected. The first argument for creating is the type of the new node. Supported types are `label`, `page`, and `externalurl`. See the bean help for alternative and shorter names. It is not possible to create a node of type `internalurl` from a script.

The second argument is a name for the new node. It is set as the provisional English title, and the common name of the new node is computed from it. The last argument is one markup that is supported by these nodes. Extra markups can be enabled by manipulating the list of markups. For more information, see *List valued attributes*. The `create` command returns the ID of the newly created node. If the keyword `select` is appended to the command, the created node becomes the current selection.

Optionally, the script can specify a `shared-flag`, which indicates whether the new page is a shared page (`shared`) or a non-shared page (`nonshared`). The optional flag `private-flag` indicates whether the new page is private or public (possible values are `private` and `public`). This flag is only valid if the content node type is `page`.

When you derive a page, the parent for the new page in the content tree must be selected. Only pages that are flagged as shared and public can be used as base pages for derivation. The first argument is the name of the new page. The type of the node is implicit, since only pages can be derived.

The second argument is the keyword *from* and the third argument is the identifier of the page to derive `from`. The supported markups are the same as for the base page. The `derive` command returns the ID of the newly created node. If the keyword `select` is appended to the command, the created node becomes the current selection.

Do not rely on the name that is set as the English title, since this behavior might change in the future. Set the English title explicitly if you plan to support the *en*locale. However, the common name is computed from the name argument.

Jython example:

```
Content.create(type, name, markup)
Content.create(type, name, markup, "select")
Content.create(type, name, markup, [shared_flag,]
[private_flag,] "select")

Content.derive(name, "from", ID)
Content.derive(name, "from", ID, "select")

# example: create and select a label at the first level,
#          then create a derived page under the new label
Content.select("the", "root")
Content.create("label", "Leisure", "html", "select")
Content.derive("Movies", "from", node_ID)
```

Jacl example:

```
$Content create type name markup
$Content create type name markup select
$Content create type name markup [shared_flag] [private_flag]
select

$Content derive name from ID
$Content derive name from ID select

# example: create and select a label at the first level,
#          then create a derived page under the new label
$Content select the root
$Content create label "Leisure" html select
$Content derive "Movies" from node_ID
```

To create a static page, run the following command:

Jython:

```
Content.create(staticpage, title, markup, zip_file_name, filename, displayoption, "select")
```

For example, `Content.create(staticpage, MyStaticPageTitle, html, c:/tmp/StaticContentPage.zip, index.html, inline, "select")`

Jacl:

```
$Content create staticpage title markup zip_file_name filename [displayoption] [select]
```

For example, `$Content create staticpage MyStaticPageTitle html c:/tmp/StaticContentPage.zip index.html [inline] [select]`

The preceding example creates a static page beneath the currently selected content node for the HTML markup with the page title MyStaticPageTitle. The content of the page is read from `c:/tmp/StaticContentPage.zip`. The entry point for the page display is read from `index.html`, which must be contained in the `ZIP` archive. To specify the display method, you can use the optional parameter **`displayoption`**. This parameter takes one of the following values `inline`, `iframe`, or `ajax`. The

default value is `inline`. To make the newly created static page the currently selected content node, use the optional parameter **`select`**.

To get the static page content in the format of a `ZIP` archive, run the following command:

Jython:

```
Content.pageget(oid, markup, zip_file_name)
```

For example, `Content.pageget(6_CGAH47L00G2N802TJFV58Q3000, html, c:/tmp/MyStaticContentPage.zip)`

Jacl:

```
$Content pageget oid  markup  zip_file_name
```

For example, `$Content pageget 6_CGAH47L00G2N802TJFV58Q3000 html c:/tmp/MyStaticContentPage.zip`

The preceding example writes the content of the specified static page to `c:/tmp/MyStaticContentPage.zip`.

To set the static page content by specifying a `ZIP` file name, use the following command:

Jython:

```
Content.pageset(oid, markup, zip_file_name, filename)
```

For example, `Content.pageset(6_CGAH47L00G2N802TJFV58Q3000, html, c:/tmp/NewStaticContentPage.zip, index.html)`

Jacl:

```
$Content pageset oid markup zip_file_name filename
```

For example, `$Content pageset 6_CGAH47L00G2N802TJFV58Q3000 html c:/tmp/NewStaticContentPage.zip index.html`

The preceding example updates the specified static page content with the content of `c:/tmp/NewStaticContentPage.zip`. The entry point for the page display is read from `index.html`, which must be contained in the `ZIP` archive.

The command `delete` removes a content node that has no children. Nodes that have children cannot be deleted. To reduce the risk of accidental deletion, this command always requires the ID as an argument, even if the node to be deleted is selected.

When you delete a page, all pages that are derived from that page are also deleted. This action also affects other users that have pages that are derived from that base page. If the currently selected node is deleted, either directly or by deleting a base page, the bean clears.

Jython example:

```
Content.delete(ID)
```

Jacl example:

```
$Content delete ID
```

Related concepts:

"List valued attributes" on page 1143
The content nodes have three list valued attributes, `locales`, `markups`, and `allowedportlets`. See the bean help for alternative, shorter names for these lists.

Component hierarchy:

The component hierarchy is a tree of components on a page. Components can be containers and controls. A container holds other components, a control displays a portlet. The component hierarchy is accessed and modified by using the Layout bean, referenced as `$Layout` in Jacl.

Only pages have a component hierarchy. The Layout bean always refers to the page that is selected in the Content bean. For more information, see *Content hierarchy*. When the Content bean has no current selection, or the current selection is not a page, the Layout bean cannot be used. Whenever a node is selected in the Content bean it clears the current selection of the Layout bean.

In the GUI, the component hierarchy of a page is manipulated by the customizer. The operations of the Layout bean allow for the definition of component hierarchies that the customizer cannot handle. The script developer must take care if the customizer is to be used alongside scripting.

The Layout bean offers the following functions:

- Methods that allow to browse in the layout hierarchy. For more information, see *Navigation*.
- Methods that allow to locate a layout node, or to search for particular layout nodes. For more information, see *Search* .
- Methods for getting and setting attributes or flags. The following attribute types are supported by the Layout bean:
  - Plain attributes.
  - Global flags
- Methods to create or delete layout objects. For more information, see *Lifecycle*.
- Methods to move or transfer layout objects. You can modify the component hierarchy as described in *Sequence* and *Hierarchy*. For more information, see *Organization*.

  "Navigation" on page 1150
  The Layout bean is a tree bean. The layout bean provides the index command to obtain and resolve so-called index paths. For more information, see Index paths. When started with an ID as argument, the command returns the absolute index path for that component. When started without an argument, it returns the absolute index path for the currently selected component.

  "Search" on page 1151
  The generic search syntax is documented in Search. The Layout bean supports the following keywords for node types in searches. Alternative, shorter keywords are documented in the bean help.

  "Plain attributes" on page 1151
  In addition to the default attributes, components have the following attributes. Alternatively, shorter names are documented in the bean help.

  "Global flags" on page 1152
  A composition hierarchy has several global flags. The global flags are similar to attributes of the content page node, except that they are associated with the

composition hierarchy rather than the content node. Therefore, they are accessed through the Layout bean after the page in the Content bean is selected.

"Lifecycle" on page 1153
The `create` command creates a new component. You must select the parent container for the new component. The first argument is a keyword that indicates whether a `container` or `control` is created. Alternative, shorter keywords are documented in the bean help.

Related concepts:

"Content hierarchy accessed through Content bean" on page 1140
The content hierarchy is a tree of content nodes. Content nodes can be labels, compositions, and links. Compositions are also called pages. Links can be internal or external. In the GUI, links represent the nodes in the Favorites list. Internal links point to a portal page, external links can point to any URL.

"Organization" on page 1138
For some beans, in particular the Content and Layout beans, the order of nodes is significant. In tree beans, the parent relationship of the nodes defines the hierarchy. Only nodes with the same parent node are in a particular order in trees.

Navigation:

The Layout bean is a tree bean. The layout bean provides the index command to obtain and resolve so-called index paths. For more information, see Index paths. When started with an ID as argument, the command returns the absolute index path for that component. When started without an argument, it returns the absolute index path for the currently selected component.

When started with an index path and a keyword, the `index` command resolves the argument path. If the keyword is `find`, it returns the ID of the addressed component. If the keyword is `select`, it also selects it. The keyword is not case-sensitive. If the index path is not absolute, it is resolved relative to the selected component.

Jython example:

```
Layout.index()
Layout.index(ID)

Layout.index(ipath, "find")
Layout.index(ipath>, "select")

# example: resolve an absolute index path
Layout.index("/1/0/3", "find")

# example: select first child of current container
Layout.index("0", "select")
```

Jacl example:

```
$Layout index
$Layout index ID

$Layout index ipath find
$Layout index ipath select

# example: resolve an absolute index path
$Layout index /1/0/3 find

# example: select first child of current container
$Layout index 0 select
```

Related concepts:

"Index paths" on page 1175
Index paths are used to refer to components in the component hierarchy. They are based on the index or position of a component in the surrounding container. An index path is a multi-dimensional index of a component, where the number of dimensions is equal to the depth of the component in the tree. Index paths are absolute or relative, depending on whether there is a leading slash. Absolute paths start with a leading slash and are resolved from the root component. Relative paths start with a number and are resolved from the selected component. Trailing slashes are irrelevant.

Search:

The generic search syntax is documented in Search. The Layout bean supports the following keywords for node types in searches. Alternative, shorter keywords are documented in the bean help.

- `container`
- `control`
- `all or any`

Plain attributes:

In addition to the default attributes, components have the following attributes. Alternatively, shorter names are documented in the bean help.

Table 153. A description of the plain attributes

Attribute	Description
`position`	The numeric position among the siblings, zero-based.
`skinid`	The identifier of the skin for the component.
`skinname`	The name of the skin for the component.
`modifiable`	A flag indicates whether the component can be modified.
`deletable`	A flag indicates whether the component can be deleted.
`width`	The width of the component, in pixel or percent.

The `skinid`, `modifiable`, `deletable`, and `width` attributes are writable. The Boolean value of the flag attributes can be given as *true/false*, *t/f*, *1/0*, or *on/off*. It is returned as *true/false* by the `get` command. The flag values are local values of the component. On derived pages, a component is modifiable or deletable only if the flag is also set on all base pages where that component is defined. The value of the width attribute can be given as a number of pixels, or as a numeric percentage followed by the percent sign.

The `skinname` attribute is not writable, but the value depends on the `skinid` attribute. The `position` attribute is not writable either, but the value depends on the organization of the component tree. For more information, see *Organization*.

Jython example:

```
# examples for setting attributes of the selected node
Layout.set("modifiable", "true")
Layout.set("deletable", "0")
Layout.set("width", "350")
```

Jacl example:

```
# examples for setting attributes of the selected node
$Layout set modifiable true
$Layout set deletable 0
$Layout set width "350"
```

Containers have the following extra attributes:

Table 154. Description of the additional containers attribute

Attribute	Description
`orientation`	The orientation of the container.

The `orientation` attribute is writable. The value is returned as *horizontal* or *vertical*. It can be set as *horizontal/vertical* or as *row/column* or as *row/col*.

Controls have the following extra attributes:

Table 155. Description of the additional controls attributes

Attribute	Description
`portletdefinition`	The ID of the portlet that is shown in the control.
`portletentity`	The ID of the portlet entity that is shown in the control.

These attributes are not writable.

Related concepts:

"Organization" on page 1138
For some beans, in particular the Content and Layout beans, the order of nodes is significant. In tree beans, the parent relationship of the nodes defines the hierarchy. Only nodes with the same parent node are in a particular order in trees.

Global flags:

A composition hierarchy has several global flags. The global flags are similar to attributes of the content page node, except that they are associated with the composition hierarchy rather than the content node. Therefore, they are accessed through the Layout bean after the page in the Content bean is selected.

You can query the global flags with the `getflag` and modify them with the `setflag` command. The `getflag` command expects the name of the flag as the first argument. If the keyword `numeric` is added, it returns the value of the flag as a Boolean *1* or *0*. Without the keyword, it returns a string *true* or *false*. The `setflag` command expects the name of the flag as the first argument, the new value as the second. You can specify the new value as any Boolean value recognized by the BSF, like *1/0*, *true/false*, *t/f*, *on/off*.

Jython example:

```
Layout.getflag(flag)
Layout.getflag(flag, "numeric")

Layout.setflag(flag, value)

# query a flag in a condition
if Layout.getflag("active", "numeric"):
  ...
else:
  ...
```

Jacl example:

```
$Layout getflag flag
$Layout getflag flag numeric

$Layout setflag flag value

# query a flag in a condition
if [$Layout getflag active numeric] then {...} else {...}
```

The following global flags are supported. The bean help documents alternative and shorter names.

Table 156. Description of the global flags

Attribute	Description
`active, a`	Whether the page is active.
`bookmarkable, bookmark, b`	Whether an internal link to the page can be created.
`shareable, share, s`	Whether the page can be used as a base for derived pages.

A page is not shown in the navigation if the active flag is false. You can deactivate the pages while they are modified to prevent user from seeing an inconsistent view of the page. The `bookmarkable` flag is not writable.

Lifecycle:

The `create` command creates a new component. You must select the parent container for the new component. The first argument is a keyword that indicates whether a `container` or `control` is created. Alternative, shorter keywords are documented in the bean help.

The interpretation of the second argument depends on the type of object to create. When you create a container, it is the initial value for the `orientation` attribute. When you create a control, it is the value for the `portletdefinition` attribute. See *Plain attributes* for both attributes. If the keyword `select` is appended, the newly created node becomes the current selection.

A page that is created, or from which all components are deleted, does not have a root container. Only in that case, it is possible to create a container without selecting a parent. The new container becomes the root container for the page. It is not possible to create a page with a root control.

Jython example:

```
Layout.create("container", orientation)
Layout.create("container", orientation, "select")
```

```
Layout.create("control", portlet_ID)
Layout.create("control", portlet_ID, "select")

# example: append a column to the second row
Layout.index("/1", "select")
Layout.create("container", "column")

# example: create a new control in the first row or column
Layout.index("/0", "select")
Layout.create("control", portlet_ID)

# example: create a page with a control in a row
Content.create("page", "New Page", "html", "select")

# no deselect required, since nothing can be selected
Layout.create("container", "row", "select")
Layout.create("control", portlet_ID)
```

Jacl example:

```
$Layout create container orientation
$Layout create container orientation select

$Layout create control portlet_ID
$Layout create control portlet_ID select

# example: append a column to the second row
$Layout index /1 select
$Layout create container column

# example: create a new control in the first row or column
$Layout index /0 select
$Layout create control portlet_ID

# example: create a page with a control in a row
$Content create page "New Page" html select
# no deselect required, since nothing can be selected
$Layout create container row select
$Layout create control portlet_ID
```

The only argument that is required for the `delete` command is the ID of the node to delete. To prevent accidental deletion, you must explicitly specify the ID even if the node is selected. You can delete a container only if it is empty. If the selected node is deleted, the bean automatically clears.

Jython example:

```
Layout.delete ID
```

Jacl example:

```
$Layout delete ID
```

Related concepts:

"Plain attributes" on page 1151
In addition to the default attributes, components have the following attributes. Alternatively, shorter names are documented in the bean help.

Portlet repository:

The portlet repository provides access to portlets, portlet applications, and web modules. To provide easy access to the relations between the repository objects, the repository is modeled as a tree. Unlike with the content and component hierarchies, the repository tree is not arbitrarily nested.

At the root of the repository is a dummy node. Children of the root are the web modules. web modules are the deployment units of the portal. They correspond to WAR files. The children of a web module are portlet applications.

The portlet repository is accessed by using the Portlet bean, referenced as `$Portlet` in Jacl. Modification of the portlet repository from a script is not supported.

The Portlet bean provides the following functions:

- Methods to browse in the portlet repository tree. For more information, see *Navigation*.
- Methods to locate a portlet, application, or web module, or to search for particular repository objects. For more information, see *Search* .
- Methods for getting attributes, metadata, or portlet preferences. The following attribute types are supported by the Layout bean:
  - Plain attributes
  - List valued attributes
  - Locale-specific attributes
  - Portlet metadata
  - Portlet preferences

"Navigation" on page 1156
The Portlet bean is a tree bean. The navigation is documented in Tree navigation. For more information, see Tree navigation.

"Search" on page 1156
The generic search syntax is documented in Search. The Portlet bean supports the following keywords for node types in searches. Alternative, shorter keywords are documented in the bean help.

"Plain attributes" on page 1157
In addition to the default attributes, repository objects have some of the following attributes. Alternative, shorter names are documented in the bean help.

"List valued attributes" on page 1158
The portlet nodes have two list valued attributes, `locales` and `parameters`. See the bean help for alternative, shorter names for these lists. Only the `list` command is supported, you cannot modify the lists. The locales list is available for portlets and portlet applications. It holds the locales for which locale-specific attributes are defined in locale-specific attributes. The `parameters` list holds the setting names for the *init* parameter that are associated with the specified portlet.

"Locale-specific attributes in portlet bean" on page 1158
Portlets and portlet applications have the following locale-specific attributes. All attributes are optional. You can query the list of locales for which attributes are defined by using the `list` command (List valued attributes).

"Portlet metadata" on page 1158
Portlet nodes can own metadata, which are name-value pairs of data that is associated with the content node. Metadata are used by the portal, for example, to set display attributes, or by the user. The Portal bean allows read-only access on portlet metadata. Metadata can be assigned only with portlet-type or application-type repository objects.

"Portlet preferences in portlet bean" on page 1159
It is possible to get and set preferences to portlet instances. Because Portlet beans specify a static portlet only, and not a portlet instance that is found on a

page, portlet instances are identified by two ID values: The ID of the portlet that specifies the portlet, and the ID of the portlet entity `piid` that anchors an instance of the portlet in a page.

Navigation:

The Portlet bean is a tree bean. The navigation is documented in Tree navigation. For more information, see Tree navigation.

There is one dedicated `select` command to select the web module of the currently selected portlet application or portlet.

Jython example:

```
Portlet.select("the", "root")
Portlet.select("the", "parent")
Portlet.select("the", "webmodule")
```

Jacl example:

```
$Portlet select the root
$Portlet select the parent
$Portlet select the webmodule
```

Related concepts:

"Tree navigation" on page 1131
The Content, Layout, and Portlet beans each represent a tree hierarchy. The basic navigation methods are the same for all three. A tree bean provides methods to access the root node to look up the parent and children of a node, and to maintain a cursor that points to a selected node in the tree.

Search:

The generic search syntax is documented in Search. The Portlet bean supports the following keywords for node types in searches. Alternative, shorter keywords are documented in the bean help.

- `webmodule`, `webmodules`, `module`, `modules`, `wm`, `w`, `m`
- `application`, `applications`, `app`, `apps`, `a`
- `portlet`, `portlets`, `p`
- `concrete` (portlet or application)
- `abstract` (web module)
- `all or any`

In addition to the default search criteria, the Portlet bean supports two criteria that match on the name of the objects. The name is similar to a common name, except that there are no restrictions on the character set. The name is a real attribute of the repository objects, not a synthesized one like the common name.

Table 157. Descriptions of the 2 search criteria that match on the name of the objects

Attribute	Description
`namehas`	The value is a string. The search is for objects with the string as a substring in their name. Comparison is not case-sensitive.
`nameis`	The value is a string. The search is for objects with the string as their name. Comparison is case-sensitive.

Jython example:

```
# example: search all applications related to news
Portlet.deselect()
Portlet.search("application", "namehas", "News")

# example: find and select unique news portlet
#          will fail if none or several are found
Portlet.find("portlet", "nameis", "NewsPortlet", "select")
```

Jacl example:

```
# example: search all applications related to news
$Portlet deselect
$Portlet search application namehas "News"

# example: find and select unique news portlet
#          will fail if none or several are found
$Portlet find portlet nameis "NewsPortlet" select
```

Plain attributes:

In addition to the default attributes, repository objects have some of the following attributes. Alternative, shorter names are documented in the bean help.

Table 158. Description of the plain attributes for the repository objects

Attribute	Description
`name`	The name, for all object types.
`version`	The version number, for web modules only.
`contextroot`	The absolute path URI, for web modules only.
`resourceroot`	The relative directory, for web modules only.
`defaultlocale`	The default locale, for applications and portlets.

These attributes are also available:

Table 159. Description of the additional plain attributes for the repository objects

Attribute	Description
`id, oid, guid`	The identifier of the node. Suitable input for *select*.
`uniquename, uname, un`	The global unique name of the node.
`type`	The type of the node. Options: repository, web module, application, portlet
`name`	The name of the node.
`commonname, cname, cn`	The common name that is generated for the node.
`version`	The version number. Only for web modules.
`contextroot, context, uri`	The relative URI for addressing the web module. Only for web modules.
`resourceroot, resdir, dir, war`	The subdirectory where the web module resources can be found.
`defaultlocale, dlocale, defloc`	The default locale. Only for portlet applications and portlets.

Table 159. Description of the additional plain attributes for the repository objects (continued)

Attribute	Description
`servlet`	The ID of the associated servlet. Only for portlets.
`cachescope, cs`	The remote cache scope of this repository node. Only portlet nodes can have this read-only flag. The attribute value can be empty. Valid attribute values are *shared* or *s* for a shared remote cache scope, *nonshared*, or *ns* for a non-shared remote cache scope.
`cacheexpiration, cacheexp, cexp`	The expiration value of the remote cache, in seconds. Only portlet nodes can have this read-only flag. The attribute value must be a numeric expression, where *-1*means that the cache never expires, *0* means that the cache expires at each request.
`remotecachedynamic, rcd`	Indicates whether the remote cache is dynamic or static. Only portlet nodes can have this read-only flag. A value of true indicates that the remote cache is dynamic. false indicates that the remote cache is a static cache.

All attributes are read-only, there is no `set` command. The `resourceroot` attribute typically is the name of the WAR file of the web module.

List valued attributes:

The portlet nodes have two list valued attributes, `locales` and `parameters`. See the bean help for alternative, shorter names for these lists. Only the `list` command is supported, you cannot modify the lists. The locales list is available for portlets and portlet applications. It holds the locales for which locale-specific attributes are defined in locale-specific attributes. The `parameters` list holds the setting names for the *init* parameter that are associated with the specified portlet.

Locale-specific attributes in portlet bean:

Portlets and portlet applications have the following locale-specific attributes. All attributes are optional. You can query the list of locales for which attributes are defined by using the `list` command (List valued attributes).

- `title`
- `shorttitle`
- `keywords`
- `description`

Only the `nlsget` command is available for locale-specific attributes.

Portlet metadata:

Portlet nodes can own metadata, which are name-value pairs of data that is associated with the content node. Metadata are used by the portal, for example, to set display attributes, or by the user. The Portal bean allows read-only access on portlet metadata. Metadata can be assigned only with portlet-type or application-type repository objects.

Jython example:

```
Portlet.parmget(ID, name)
Portlet.list(ID, "parm")

# only for beans with a current selection
Portlet.parmget(name)
Portlet.list("parm")

# example: get the metadata for an instance property named #
"MyPortletConfig"
print Portlet.parmget("MyPortletConfig")

# example: list all metadata names for the selected portlet
for pname in Portlet.list("parm").split():
    print pname
```

Jacl example:

```
$Portlet parmget ID name
$Portlet list ID parm

# only for beans with a current selection
$Portlet parmget name
$Portlet list parm

# example: get the metadata for an instance property named # "MyPortletConfig"
puts "[$Portlet parmget MyPortletConfig]"

# example: list all metadata names for the selected portlet
foreach pname [$Portlet list parm] {
puts "$pname"
}
```

Portlet preferences in portlet bean:

It is possible to get and set preferences to portlet instances. Because Portlet beans specify a static portlet only, and not a portlet instance that is found on a page, portlet instances are identified by two ID values: The ID of the portlet that specifies the portlet, and the ID of the portlet entity `piid` that anchors an instance of the portlet in a page.

The command `prefnames` is used to obtain a list of available preference names. Portlet preferences might have multiple values. The command `getpref` therefore accepts a numeric `index` attribute. The numeric `index` attribute denotes the number of the value to be obtained. For example, *0* means to return the first preference value, *1* means to return the second value. The total number of available preference values for a specific preference name is returned by the `prefcount` command.

The `addpref` command is used to add a portlet preference value to the portlet preference list. You can set portlet preferences to read-only. You can control this setting with the commands `spprof` (Set Portlet Preference Read-only Flag) and `gpprof` (Get Portlet Preference Read-only Flag).

Jython example:

```
Portlet.prefnames(ID, piid)
Portlet.getpref(ID, piid, name, "at", index)
Portlet.addpref(ID, piid, name, value)
Portlet.gpprof(ID, piid, name, ["numeric"])
Portlet.spprof(ID, piid, name, readonly_flag)
```

Jacl example:

```
$Portlet prefnames ID piid
$Portlet getpref ID piid name at index
$Portlet addpref ID piid name value
$Portlet gpprof ID piid name [numeric]
$Portlet spprof ID piid name readonly_flag
```

The portlet is identified by two ID values: The ID of the portlet and the ID of the portlet entity. You can obtain the ID of the portlet entity by the command `$Layout get piid` or `Layout.get("piid")`.

Jython example:

```
# example: set a portlet preference
# 1. locate the control where the portlet resides
# 2. obtain its portlet instance ID
# 3. set the portlet preference accordingly
ctl = Layout.find("all", "pid", Portlet.csn(), "select")
piid = Layout.get(ctl, "piid")
Portlet.addpref(piid, "MYKEY", "Value")

# example: list portlet preference values for key "MYKEY"
# the control ID is stored in the variable ctrl
for ix in range($Portlet.prefcount(piid, "MYKEY")):
  print Portlet.getpref(piid, "MYKEY", "at", ix)

# example: drop all portlet preferences on a certain key
Portlet.droppref(piid, "MYKEY")
```

Jacl example:

```
# example: set a portlet preference
# 1. locate the control where the portlet resides
# 2. obtain its portlet instance ID
# 3. set the portlet preference accordingly
set ctl [$Layout find all pid [$Portlet csn] select]
set piid [$Layout get $ctl piid]
$Portlet addpref $piid MYKEY "SomeValue"

# example: list portlet preference values for key "MYKEY"
# the control ID is stored in the variable ctrl
for {set ix 0} {$ix < [$Portlet prefcount $piid MYKEY]} {incr ix} {
  puts "[$Portlet getpref $piid MYKEY at $ix]"
}

# example: drop all portlet preferences on a certain key
$Portlet droppref $piid  MYKEY
```

Dependent on the standard that the portlet complies with, there are differences in the handling of portlet preferences. You can set multiple preference values on a preference key with standard portlets, while IBM portlets support only single values on each portlet preference key.

Themes and Skins:

Themes and skins are two distinct sets of objects with a matrix relation, where each skin can be tied to any number of themes. The sets of themes and skins are accessible through the Look bean, which is referenced as `$Look` in Jacl.

The Look bean provides access to the portal theme and skin objects:

- Methods that allow to search on themes. For more information, see *Search* .
- Methods for getting attributes. Themes and Skins support the default attributes. Only the `get` command is available. Since there is no current selection, the ID of the node for which to obtain an attribute value must be given explicitly.

- Methods to retrieve theme and skin lists. For more information, see *Global lists*.

  "Navigation"
  The Look bean is a simple lookup mechanism to access the name and ID of the available themes and skins. It does not support a current selection, nor navigation of the matrix relation between both sets. Since that might change in the future, the `deselect` command is implemented. It can be used before searches to ensure that the search is global, even if a current selection is introduced in the future.

  "Search"
  The syntax for searching is similar to the generic tree search described in Search. Since the look bean does not support a current selection, the search is not scoped and the keyword `select` is not supported for the `find` command. The command `deselect` (Navigation) can be used to ensure that a search is global, even if a search scope is introduced in the future.

  "Global lists" on page 1162
  For ease of use, two global lists hold the common name of all themes and skins. You can access the global lists with the `listall` command, which expects the list name as an argument. The following list names are supported. See the bean help for alternative list names.

Navigation:

The Look bean is a simple lookup mechanism to access the name and ID of the available themes and skins. It does not support a current selection, nor navigation of the matrix relation between both sets. Since that might change in the future, the `deselect` command is implemented. It can be used before searches to ensure that the search is global, even if a current selection is introduced in the future.

Jython example:

```
Look.deselect()
```

Jacl example:

```
$Look deselect
```

Search:

The syntax for searching is similar to the generic tree search described in Search. Since the look bean does not support a current selection, the search is not scoped and the keyword `select` is not supported for the `find` command. The command `deselect` (Navigation) can be used to ensure that a search is global, even if a search scope is introduced in the future.

The look bean supports the default search criteria, and the following keywords for node types in searches. Alternative keywords are documented in the bean help.

- `theme`
- `skin`

Since the sets of themes and skins are distinct, there is no option to search for both type of nodes at the same time.

Global lists:

For ease of use, two global lists hold the common name of all themes and skins. You can access the global lists with the `listall` command, which expects the list name as an argument. The following list names are supported. See the bean help for alternative list names.

- `themes`
- `skins`

Jython example:

```
# example: list the names of all themes
Look.listall("themes")

# example output:
Admin WebSphere Engineering Science Finance Corporate
```

Jacl example:

```
# example: list the names of all themes
$Look listall themes

# example output:
Admin WebSphere Engineering Science Finance Corporate
```

Portal Access Control:

The scripting operations for access control differ fundamentally from content, layout, or the portlet repository. The reason is that access control data is not transparently cached on the client. To avoid many requests and slow response times for every simple lookup operation, a different programming model is adopted for access control data.

The collected access control data for a resource is represented on the client by an object. These objects are explicitly created, locally modified, and written back to the portal. All local modifications become effective when the object is written back.

Two script beans are provided for access control data. The Access bean, referenced as `$Access` in a script, is for reading and writing the access control objects. The PacList bean, referenced as `$PacList`, provides operations to view and edit access control objects.

The access bean and PacList bean provide the following functions:

- Methods to request or release access list objects, and to load the permission set for view, edit, or manage permissions. For more information, see *Lifecycle* .
- Methods to get a list of available action sets. For more information, see *Global lists*.
- Methods to grant or revoke access for principals on the portal resource, or to view the permissions that are defined for the portal resource. For more information, see *Principals*.
- Methods to set or clear permission blocks. For more information, see *Permission blocks*.

"Access control objects" on page 1163
The complete access control data for a resource is represented on the client by a PacList object. PAC stands for Portal Access Control. PacList objects are opaque, they cannot be manipulated directly. Instead, they are loaded into the PacList

bean, which provides the operations to view and edit the objects. PacList objects are obtained from and written back by using the Access bean.

"Lifecycle" on page 1164
PacList objects are read from and written back to the portal by using `getacl` and `setacl` in the Access bean. The command `getacl` returns the PacList object for a resource. It expects the category and identifier of the resource as arguments. The category identifies the Script bean that is responsible for the resource. It is given as a keyword, which can be the bean name or the type of the resource. The supported keywords are documented in the help for the Access bean. PacList objects are obtained only for resources that are handled by the Content and Portlet beans. For the root node of the portlet repository, there are no PacList objects.

"Global lists" on page 1166
A global list holds the names of the predefined action sets. The global list is accessed with the `listall` command in the Access bean. The command expects the list name as an argument. The only supported list name is `actionsets`. See the bean help for alternative list names.

"Principals" on page 1166
Use the `list` command to access the list of principals for an action set.

"Permission blocks" on page 1168
Use the `show` command to access the flags that control distribution of permissions. The first argument is the name of the action set, and the second is the name of the flag to obtain. The optional keyword numeric indicates whether the flag value must be returned as a human readable string, or as a numeric value suitable for programmatic evaluation. Since the string value is subject to translation into different locales, only the numeric value can reliably be used in conditional statements. The numeric value *0* indicates a block, while *1* stands for allowed distribution. The following names are supported for the two flags. Alternative names are documented in the help for the PacList bean.

Access control objects:

The complete access control data for a resource is represented on the client by a PacList object. PAC stands for Portal Access Control. PacList objects are opaque, they cannot be manipulated directly. Instead, they are loaded into the PacList bean, which provides the operations to view and edit the objects. PacList objects are obtained from and written back by using the Access bean.

The access control data for a resource is split in similar data groups for several action sets. It is also referred to as role types. The term action set is used here, as it has the smaller potential for misinterpretation. The portal uses predefined action sets, which combine the actions for the following types of portal users. One name for each action set is given in parentheses. Alternative names are documented in the help of the PacList bean.

- User (`User`)
- Privileged User (`PrivilegedUser`)
- Editor (`Editor`)
- Manager (`Manager`)
- Delegator (`Delegator`)
- Security Administrator (`SecurityAdmin`)
- Administrator (`Admin`)

For each action set, a list of principals are explicitly allowed to complete the corresponding actions. A principal is a group or user, which are specified by a name or distinguished name (DN). There are three special principals, which represent an anonymous user, all authenticated users, and all user groups.

In addition to the list of principals, two flags control the implicit distribution of permissions through resource hierarchies. The inheritance flag controls distribution of permissions from parent to child nodes. If inheritance is not blocked, a principal that has permission on the parent has permission on the child as well. This kind of permission is useful for administrative actions. The propagation flag controls the other direction. If propagation is not blocked, a principal that has permission on at least one child has permission on the parent as well. This kind of permission is useful for simple actions, such as viewing a page, which requires view access to all parents.

Although the flags can be manipulated for each action set, they are ignored for security administrators and administrators. For these two action sets, inheritance and propagation are never blocked.

Lifecycle:

PacList objects are read from and written back to the portal by using `getacl` and `setacl` in the Access bean. The command `getacl` returns the PacList object for a resource. It expects the category and identifier of the resource as arguments. The category identifies the Script bean that is responsible for the resource. It is given as a keyword, which can be the bean name or the type of the resource. The supported keywords are documented in the help for the Access bean. PacList objects are obtained only for resources that are handled by the Content and Portlet beans. For the root node of the portlet repository, there are no PacList objects.

The command `setacl` writes a PacList object back to the portal. That command is necessary only if the PacList object is modified. The PacList object is the only argument to the command. Each PacList object is tied to a particular resource and cannot be written for any other resource.

Jython example:

```
Access.getacl(category, ID)
Access.setacl(paclist)

# example: get PacList object for a particular page
#          the object is stored in variable "acl"
Content.find("label", "uniquename",
"ibm.portal.Administration", "select")
acl = Access.getacl("Content", Content.current())

# example: write back PacList object in variable "acl"
Access.setacl(acl)
```

Jacl example:

```
$Access getacl category ID
$Access setacl paclist

# example: get PacList object for a particular page
#          the object is stored in variable "acl"
$Content find label uniquename ibm.portal.Administration select
set acl [$Access getacl Content [$Content current]]

# example: write back PacList object in variable "acl"
$Access setacl $acl
```

To access or manipulate a PacList object, it must be loaded in the PacList bean. The operations of the PacList bean always refer to the PacList object currently loaded. An object can be loaded only for viewing by using `view`, or for manipulation by using `edit`. Both commands expect the PacList object as an argument. The PacList bean is loaded only if it is unloaded, or if it is loaded with an object that is not modified.

Jython example:

```
PacList.view(paclist)
PacList.edit(paclist)

# example: load PacList object for content root, view only
PacList.view(Access.getacl("Content", Content.root()))

# example: load PacList object for a particular page
Content.find("label", "uniquename",
"ibm.portal.Administration", "select")
acl = Access.getacl("Content", Content.current())
PacList.edit(acl)
```

Jacl example:

```
$PacList view paclist
$PacList edit paclist

# example: load PacList object for content root, view only
$PacList view [$Access getacl Content [$Content root]]

# example: load PacList object for a particular page
$Content find label uniquename ibm.portal.Administration select
set acl [$Access getacl Content [$Content current]]
$PacList edit $acl
```

The command `current` returns the name of the loaded object, which includes the identifier of the resource. The command `done` unloads the PacList bean and returns the object that was loaded. If the PacList bean is already unloaded, `done` still returns the PacList object that was loaded before, which is useful for dealing with errors.

If the currently loaded PacList object is modified since it was loaded, the command `modified` returns. If the keyword `numeric` is added, it returns the value of the flag as a Boolean *1* or *0*. Without the keyword, it returns a string *true* or *false*.

Jython example:

```
PacList.current()
PacList.done()

PacList.modified()
PacList.modified("numeric")

# example: write back and unload PacList object
#          only if it was modified
if PacList.modified("numeric"):
  Access.setacl(PacList.done()

# example: recover the last loaded PacList object
PacList.view(PacList.done())
```

Jacl example:

```
$PacList current
$PacList done
```

```
$PacList modified
$PacList modified numeric

# example: write back and unload PacList object
#          only if it was modified
if [$PacList modified numeric] {
  $Access setacl [$PacList done]
}

# example: recover the last loaded PacList object
$PacList view [$PacList done]
```

Global lists:

A global list holds the names of the predefined action sets. The global list is accessed with the `listall` command in the Access bean. The command expects the list name as an argument. The only supported list name is `actionsets`. See the bean help for alternative list names.

Jython example:

```
# example: list the names of all predefined action sets
Access.listall("actionsets")

# example output:
Admin SecurityAdmin Delegator Manager Editor PrivilegedUser User
```

Jacl example:

```
# example: list the names of all predefined action sets
$Access listall actionsets

# example output:
Admin SecurityAdmin Delegator Manager Editor PrivilegedUser User
```

Principals:

Use the `list` command to access the list of principals for an action set.

The first argument is the name of an action set. If the second argument is the keyword `all`, the command returns all principals in the list, which is separated by newlines. A regular principal is listed with its name. A special principal is listed with a dedicated name embraced by special characters, which makes the special principals easy to spot. Since principal names can include white space, the complete list cannot be parsed into items.

You can access the principals in the list individually. The keyword `at` as the second argument indicates that the third argument is an index. Alternative keywords are documented in the bean help. The command returns the principal at the given position in the list, where *0* is the index of the first position. If the index is out of range, the command returns empty. You can obtain the number of principals by using `count`, which expects the name of the action that is set as the only argument.

Jython example:

```
PacList.list(actionset, "all")
PacList.list(actionset, "at", index)
PacList.count(actionset)

# example: get first principal in list of administrators
PacList.list("Administrator", "at", 0)
```

```
# example: list all principals in all lists
for as in Access.listall("actionsets").split():
  print "Principals for " + as:" + PacList.list(as, "all")
```

Jacl example:

```
$PacList list actionset all
$PacList list actionset at index
$PacList count actionset

# example: get first principal in list of administrators
$PacList list Administrator at 0

# example: list all principals in all lists
foreach as [$Access listall actionsets] {
  puts "\nPrincipals for $as:"
  puts [$PacList list $as all]
}
```

The command `grant` adds a principal to a list of principals. The first argument is the name of the action that is set for which the principal is added. The second argument is a keyword that indicates whether a regular or special principal is to be added, as described later. The third argument specifies the principal to add. The PacList bean help documents alternative keywords for the second argument.

If the `keyword` name is given as the second argument, a regular principal is added. The third argument is the name of the principal, either as a fully qualified distinguished name, or as a short name. The name is resolved to a standard form when the PacList object writes back to the portal. Therefore, it is possible to have duplicates in the list, which is removed by the portal.

If the keyword `special` is given as the second argument, a special principal is added to the list. The third argument is a keyword that indicates which special principal can be added. The following special principals are supported. Alternative keywords are documented in the help for the PacList bean.

- `anonymous` - a user that is not logged in
- `authenticated` - any user that is logged in
- `allgroups` - any group of a user that is logged in. This virtual user group contains all non-virtual user groups.

The `grant` command returns a success message if a principal was added to the list. It returns empty if the principal to be added was spotted as a duplicate. As mentioned earlier, duplicates cannot be spotted by the PacList bean if names in non-canonical form are used.

Jython example:

```
PacList.grant(actionset, "name", name)
PacList.grant(actionset, "special", keyword)

# example: grant user access to anyone authenticated
PacList.grant("User", "special", "Authenticated")

# example: grant editor access to John Doe - twice
PacList.grant("Editor", "name", "johndoe")
PacList.grant("Editor", "name", "uid=johndoe,dc=example_1,dc=com")
```

Jacl example:

```
$PacList grant actionset name name
$PacList grant actionset special keyword
```

```
# example: grant user access to anyone authenticated
$PacList grant User special Authenticated

# example: grant editor access to John Doe - twice
$PacList grant Editor name "johndoe"
$PacList grant Editor name "uid=johndoe,dc=example_1,dc=com"
```

The command `revoke` deletes principals from the list for an action set. The first argument is the name of the action set. The second argument is a keyword that indicates how the principal to delete is specified, as described later. Alternative keywords are documented in the help for the PacList bean. In most cases, a third argument specifies the principal to be removed. The command returns a success message if a principal was removed from the list, or empty if the principal to be removed was not found.

If the second argument is the keyword `all`, the command deletes all principals from the list. This deletion of principals is the only case where more than one principal is removed. It is also the only case without a third argument.

If the second argument is the keyword `at`, the third argument is the index of the principal to be deleted. This addressing style is similar to the `list` command. It can be used to delete regular and special principals. The indexes of the succeeding principals are changed by this operation.

If the second argument is the keyword `name`, the third argument is the name of the regular principal to delete. This addressing style is similar to the `grant` command. Since the PacList bean cannot resolve names into canonical form, the name must be given in the exact same form in which it is present in the list. If the principal was read from the portal, that is the standard form. If the principal was added by a previous `grant` command, the name in the list is unchanged from the argument that is given to that command.

If the second argument is the keyword `special`, the third argument is a keyword for the special principal to delete. This addressing style is similar to the grant command and uses the same keywords for the special principals.

Jython example:

```
PacList.revoke(actionset, "all")
PacList.revoke(actionset, "at", index)
PacList.revoke(actionset, "name", name)
PacList.revoke8actionset, "special", keyword)
```

Jacl example:

```
$PacList revoke actionset all
$PacList revoke actionset at index
$PacList revoke actionset name name
$PacList revoke actionset special keyword
```

Permission blocks:

Use the `show` command to access the flags that control distribution of permissions. The first argument is the name of the action set, and the second is the name of the flag to obtain. The optional keyword numeric indicates whether the flag value must be returned as a human readable string, or as a numeric value suitable for programmatic evaluation. Since the string value is subject to translation into different locales, only the numeric value can reliably be used in conditional statements. The numeric value *0* indicates a block, while *1* stands for allowed

distribution. The following names are supported for the two flags. Alternative names are documented in the help for the PacList bean.

- `inheritance` - from the parent to this node
- `propagation` - from the children to this node

Jython example:

```
PacList.show(actionset, flag)
PacList.show(actionset, flag, "numeric")

# example: evaluate manager inheritance flag
if PacList.show("Manager", "inheritance", "numeric"):
  print "inheritance is permitted"
else:
  print "inheritance is blocked"
```

Jacl example:

```
$PacList show actionset flag
$PacList show actionset flag numeric

# example: evaluate manager inheritance flag
if [$PacList show Manager inheritance numeric] then {
  puts "inheritance is permitted"
} else {
  puts "inheritance is blocked"
}
```

The commands `block` and `unblock` are used to change the flags that control distribution of permissions. They expect the name of the action set and the name of the flag as the first and second argument. The `block` command prevents the corresponding distribution of permissions, and the `unblock` command allows it. The commands return a success message if the flag value changed, or empty if the flag already had the required value.

Jython example:

```
PacList.block(actionset, flag)
PacList.unblock(actionset, flag)

# example: prevent propagation of delegator permissions
PacList.block("Delegator", "propagation");

# example: allow inheritance of user permissions
PacList.unblock("User", "inheritance")
```

Jacl example:

```
$PacList block actionset flag
$PacList unblock actionset flag

# example: prevent propagation of delegator permissions
$PacList block Delegator propagation

# example: allow inheritance of user permissions
$PacList unblock User inheritance
```

Portal authentication:

The Portal bean handles functions that are outside of the responsibility of the other beans. This responsibility includes global data and technical aspects of scripting, such as the scripting session with the portal. The Portal bean is referenced as `$Portal` in Jacl.

The Portal bean provides the following functions:

- Methods to do a login and logout. For more information, see *Authentication*.
- Methods to set a virtual portal. For more information, see *Virtual Portal selection*.

"Authentication"
Before you can access any data from a script, you must establish the user identity for script execution.

"Virtual portal selection" on page 1172
You can set a virtual portal for a `login` or `Portal.login()` command with the `$Portal setvp` or `Portal.setvp()` command. It is possible to set the virtual portal during a session, but the session becomes effective only during the `$Portal login` command. The virtual portal is specified by the URL context. The URL context is the part of the home URL that identifies the virtual portal. If `$Portal setvp` or `Portal.setvp()` is started without a URL context, the default virtual portal is set.

Authentication:

Before you can access any data from a script, you must establish the user identity for script execution.

The `login` command authenticates the scripting engine against the portal and creates a user session for the subsequent commands. There are several different options for logging in, which can be distinguished by the number of arguments.

The `login` command with two arguments expects a user name and a password as the first and second argument. The user name identifies the portal user. It can be given as a fully qualified distinguished name, or as a login name. The portal looks up the user name and verify the password.

Jython example:

```
Portal.login(user_ID, password)

# example: login as Portal user
Portal.login(user_ID, password)
```

Jacl example:

```
$Portal login user_ID password

# example: login as Portal user
$Portal login user_ID password
```

The `login` command without arguments is used only if security is enabled in the portal. In that case, a user name and password had to be given to the application server to connect to the portal. The portal accepts the user identity from the connection without a second password check. This action requires that the application server user is also a portal user.

Jython example:

```
Portal.login()
```

Jacl example:

```
$Portal login
```

The `login` command with a single argument is used only if security is enabled in the portal. Like the version without argument, the user identity from the

connection is accepted by the portal. In addition, that user requires manage permission on the portal itself. That permission is the same permission that is required to use the XMLAccess tool.

If these preconditions are satisfied, the user name that is given as argument, is looked up and taken as the portal user identity. The user name that is given as argument is different from the user name of the connection. There is no second password check. This action is similar to the su command found in many operating systems, which allows the super user to act as any other user, without knowing the user password.

Jython example:

```
Portal.login(user_ID)

# example: login and change to Portal user
Portal.login("johndoe")
```

Jacl example:

```
$Portal login user_ID

# example: login and change to Portal user
$Portal login johndoe
```

The logout command ends the scripting session and allows the portal to free the resources that are allocated for that session. It must always be used before the scripting client exits. The logout command also invalidates the portal data that is cached in the client. It is therefore possible to reauthenticate as a different user without starting a new scripting client.

You must always logout and reauthenticate as the same user if the connection from the scripting client to the portal is re-established by using the AdminControl bean of the wsadmin tool. The portal might lose the user session in that case, which causes subsequent operations fail.

Jython example:

```
Portal.logout()
```

Jacl example:

```
$Portal logout
```

In a deployment scenario, the authentication must not be handled by the deployment script itself. Use a wrapper script for the authentication instead, so that you can run the deployment script under different user identities without changes. Here is an example for a wrapper script, which assumes that the deployment script defines a run_deployment procedure.

Jython example:

```
# keep the user identity of the connection
Portal.login()
# execute the deployment procedure
run_deployment()
# clean up before exit
Portal.logout()
```

Jacl example:

```
# keep the user identity of the connection
$Portal login
# execute the deployment procedure
run_deployment
# clean up before exit
$Portal logout
```

Alternatively, the deployment script can use procedures that are defined in a profile script for login and logout. Profile scripts are run on start of the scripting client.

Jython example:

```
# login function
def portal_login(user_ID, password):
  Portal.login(user_ID, password)
  # other startup operations, like logging the access
# logout function
def portal_logout():
  Portal.logout()
  # other shutdown operations
```

Jacl example:

```
# procedure for login
proc portal_login { } {
  $Portal login user_ID password
 # other startup operations, like logging the access
}

# procedure for logout
proc portal_logout { } {
  $Portal logout
  # other shutdown operations
}
```

Virtual portal selection:

You can set a virtual portal for a `login` or `Portal.login()` command with the `$Portal setvp` or `Portal.setvp()` command. It is possible to set the virtual portal during a session, but the session becomes effective only during the `$Portal login` command. The virtual portal is specified by the URL context. The URL context is the part of the home URL that identifies the virtual portal. If `$Portal setvp` or `Portal.setvp()` is started without a URL context, the default virtual portal is set.

Jython example:

```
# example: set the virtual portal and execute a portal login
# in the example, the virtual portal URL context is employees,
# which would correspond with a base URL of
'/wps/myportal/employees'

Portal.setvp("employees")

Portal.login()
```

Jacl example:

```
# example: set the virtual portal and execute a portal login
# in the example, the virtual portal URL context is employees,
# which would correspond with a base URL of '/wps/myportal/employees'

$Portal setvp employees

$Portal login
```

Examples:

The following are examples for deleting portlets and adding portlets.

Delete portlets

Jython example:

```
# delete all welcome-portlets from all pages of a user
# see the Authentication section for portal_login and
portal_logout
portal_login(user_ID, password)
for page in Content.search("pages").split():
  Content.select(page)
  for c in Layout.search("control", "commonnamehas",
"Welcome").split():
    Layout.delete(c)

portal_logout()
```

Jacl example:

```
# delete all welcome-portlets from all pages of a user
# see the Authentication section for portal_login and portal_logout
portal_login
foreach page [$Content search pages] {
  $Content select $page
  foreach c [$Layout search control commonnamehas Welcome] {
    $Layout delete $c
   }
 }
portal_logout
```

Add portlets

Jython example:

```
# add a FunPortlet next to each WeatherPortlet

# procedure: add a portlet next to a control
# this changes the current selection of the Layout bean
def add_portlet(control, portlet):
  Layout.select(control)
  pos = Layout.get("position")
  pos = pos + 1
  Layout.select("the", parent)
  Layout.create("control", portlet, "select")
  Layout.move("to", pos)

# main program
# see the Authentication section for portal_login and
portal_logout

portal_login(user_ID, password)
fun = Portlet.find("portlet", "nameis", "FunPortlet")

for page in Content.search("pages").split():
  Content.select(page)
  for c in Layout.search("control", "commonnamehas",
"Weather").split():
    add_portlet(c, fun)

portal_logout()
```

Jacl example:

```
# add a FunPortlet next to each WeatherPortlet

# procedure: add a portlet next to a control
# this changes the current selection of the Layout bean
proc add_portlet { control portlet } {
  global Layout
  $Layout select $control
  set pos [$Layout get position]
  set pos [expr $pos + 1]
  $Layout select the parent
  $Layout create control $portlet select
  $Layout move to $pos
}

# main program
# see the Authentication section for portal_login and portal_logout

portal_login
set fun [$Portlet find portlet nameis FunPortlet]

foreach page [$Content search pages] {
  $Content select $page
  foreach c [$Layout search control commonnamehas Weather] {
    add_portlet $c $fun
  }
}
portal_logout
```

Troubleshooting:

The following solutions help solve the troubleshooting issues.

- Connectivity problem / MBean not found - connected to wrong application server
- Bean is not enabled - not logged in
- Layout bean is not enabled - no page that is selected in Content bean
- PacList bean is not enabled - no PacList object loaded
- No method that is named "xxx" is found - typographical error or capitalization in method name, check bean help
- No method that is named "xxx" with y arguments is found - wrong syntax, check bean help

Property file format:

You can provide locale-specific attributes for a set of locales in a Java property file. The generic format of Java property files is described in the Java API documentation for method load in class `java.util.Properties`. The description here covers the particular properties that are interpreted when locale-specific attributes are loaded from a portal script.

The property files are loaded by the script interpreter. They must be on the client workstation where the script is run.

Note: A property file must be encoded in the ISO 8859-1 character encoding format.

Here is an example of a property file with no prefix.

Jython example:

```
locales = en de

en.title = English Title
en.description = A description, in English.

de.title = Deutscher Titel
de.description = Eine Beschreibung auf Deutsch.
```

At the core of the file, is the property `locales`, which holds a white space separated list of the locales for which attributes are defined in the file. Only the locales that are listed here are interpreted when the property file is loaded.

The actual attribute values for each locale are defined as separate properties. The key for these properties is composed of the locale and the attribute name, which is separated by a dot. The attribute name is `title` or `description`, the locale must match the string that is listed in the `locales` property. Case is significant for property keys.

There is no fallback algorithm when locale-specific attributes are loaded. If a property `title.en` is defined, it is not considered when the title for locale en_US is required.

To combine different sets of local data into a single property file, prefixes can be used. All properties of a set must use the same prefix, which is separated from the property name by a dot. Here is an example of a property file with two sets, by using the prefixes `leisure` and `finance`.

Jython example:

```
leisure.locales = en fr
leisure.en.title = Leisure Page
leisure.fr.title = Page de Loisir

finance.locales = en de
finance.en.title = Finance Page
finance.en.description = Holds financial info portlets.
finance.de.title = Finanzseite
```

You can also load property files in the scripting format by using the XML configuration interface. To complete this step, specify the locale as part of the prefix in a `localedata` tag.

Index paths:

Index paths are used to refer to components in the component hierarchy. They are based on the index or position of a component in the surrounding container. An index path is a multi-dimensional index of a component, where the number of dimensions is equal to the depth of the component in the tree. Index paths are absolute or relative, depending on whether there is a leading slash. Absolute paths start with a leading slash and are resolved from the root component. Relative paths start with a number and are resolved from the selected component. Trailing slashes are irrelevant.

/ The root component.

/0 The first child of the root component.

/1 The second child of the root component.

...

/0/0 The first child of the first child of the root component.

/0/1 The second child of the first child of the root component.

...

/5/1/3 The fourth child of the second child of the sixth child of the root component.

...

0 The first child of the current component.

1 The second child of the current component.

...

1/0 The first child of the second child of the current component.

...

Web content administration tools

IBM Web Content Manager includes tasks and tools to help maintain your content management system. For example, use the member fixer task to resolve renamed or deleted users and user groups. Use the workflow checker and updater tools to modify workflow security settings, reschedule pending actions, and add workflow to items. Web Content Manager also includes tools to assist with library and item management, and item and version history management.

"The web content member fixer task" on page 1177
Use the member fixer task to check whether any users or groups that are referenced in IBM Web Content Manager items have been renamed or deleted and fix these references.

"Update security task" on page 1187
Use the update security task to apply inherited access permissions and remove existing item access permissions for all items or all items of a specific type. This task is useful as a post-migration step, or if you are applying major changes to your inheritance settings.

"Managing workflows by using the workflow checker tool" on page 1189
Use the workflow checker tool to update workflow security settings, or reschedule pending workflow actions.

"Updating workflows by using the workflow update tool" on page 1191
Use the workflow update tool to add a workflow to existing items that aren't already workflow enabled.

"Clearing item history" on page 1192
You use the clear history tool to clear the history of an item.

"Clearing version history" on page 1194
You use the clear versions tool to clear the version history of an item.

"Resetting the web content event log" on page 1195
From time to time, you might need to reset the web content event log. The event log can be reset only on a syndicator server. Any changes that are made by resetting the event log are then syndicated to its corresponding subscribers. In most cases, you reset the event log on the server you imported or migrated data onto, or on a syndicator to troubleshoot syndication problems in a syndication relationship.

"The export cache settings task" on page 1197
Use the export cache settings task to display a summary of the current cache settings of your system.

CF03 CF03 "How to manage plug-in tag usage" on page 1197
Plug-in tags are used to reference rendering plug-ins in Web content. Use the `run-wcm-admin-task-tag-usage` task to find or update plug-in tags in your Web content.

"Exporting and importing web content libraries" on page 1200
IBM Web Content Manager provides two methods for exporting and importing web content libraries: an export or import that operates on one library, and an export or import that creates a separate copy of a library. With either method, you can export the contents of a web content library to disk and import this data into another web content server. If you're working with a copy of a library, you can also import that library into the same web content server multiple times, resulting in a new library after each import without affecting previous copies. Exporting and importing libraries can be used to make a backup copy of a web content library and can also be used to move data between servers. However, this function cannot be used to send updates, deletes, and moves. It is only suitable for populating new items.

"Deleting libraries by using the delete libraries tool" on page 1213
Use the delete libraries tool to delete multiple libraries, even where there are references between the selected libraries.

"How to clone a web content repository" on page 1214
Syndicating items from one server to another, either after migration or to roll out a new server, can take a long time. Your database backup and restore features can be used to clone data from one repository to another, making your system ready for syndication to be used from then on for incremental updates.

The web content member fixer task

Use the member fixer task to check whether any users or groups that are referenced in IBM Web Content Manager items have been renamed or deleted and fix these references.

Member fixer is used to:

- Fix references to users in library and item level access settings that refer to users and groups from a user repository where the structure of the user repository has been altered. For example, an LDAP transfer might have been run, or the LDAP schema might have changed, or users and groups might have been moved in the LDAP.
- Fix references to users in item level access settings that refer to users and groups who have been deleted from the user repository.

The member fixer task's function is to check all of the items in a specified library for references to users and groups that no longer exist in the current user repository. In report mode, it reports all the references to members. In fix mode, these references can be fixed, either by replacing them with references to members that exist, or by removing the references. The **`fix`** parameter determines whether the member fixer task runs in report or fix mode.

References to members in library items contain the distinguished name of the member or a unique ID for the member. This unique ID is an internal ID that is unique over time, and is different to the distinguished name. This means if a member is deleted and another member is created with the same distinguished name, the two members will have different unique IDs. The **`mismatchedId`** parameter can be used to update or remove references from web content items to users with these unique IDs.

When a member that has been given permissions on a library is deleted, the member permissions are entirely removed from the library, so that any inherited permissions for items in the library will also be removed. Therefore, the member fixer task cannot be used to update these permissions to a different member. However, when an LDAP transfer is carried out, the member permissions on the library are maintained. So, the member fixer task can be run after an LDAP transfer to update or remove these permissions

"How to use the member fixer task"
Enable the member fixer task, create custom mappings, and then run the task.

"Member fixer with syndication" on page 1183
You can configure your system to automatically run the member fixer tool when syndicating. The member fixer is run on the subscriber during syndication. It is run against items that have just been syndicated. Details of the member fixer operations are included in the syndication report.

"Member fixer task frequently asked questions" on page 1184
Some frequently asked question about how to use the web content member fixer task.

How to use the member fixer task

Enable the member fixer task, create custom mappings, and then run the task.

Enabling the member fixer tool

You must first enable the member fixer by adding the following parameters to the **WCM WCMConfigService** service using the WebSphere Integrated Solutions Console:

- **connect.businesslogic.module.memberfixer.class=**com.aptrix.pluto.security.MemberFixerModul
- **connect.businesslogic.module.memberfixer.remoteaccess=**true
- **connect.businesslogic.module.memberfixer.autoload=**false

Custom Mapping

To update a reference to a member that does not exist with a member that does exist, member mappings can be defined in a custom mapping file. Where the member fixer task does not find a mapping in this file for a member, it will search the user repository for members with the same ID as the member that no longer exists. If such a member is found, it will update the reference with this user or group, or remove the reference, as specified by the altDn parameter. If no such member is found, this member is classified as 'invalid' and will be updated or removed as specified by the `invalidDn` parameter.

If custom mapping is required you must perform the following steps to map the user and group domain names before running the member fixer task:

1. Update the following properties in the *wp_profile_root*/PortalServer/wcm/shared/app/config/wcmservices/MemberFixerModule.properties file:

   cn=contentAuthors,dc=lotus,o=ibm->cn=contentEditors,dc=rational,o=ibm
   This format is used to completely replace one distinguished name with another.

   cn=[ID],dc=websphere,o=ibm->cn=[ID],dc=tivoli,o=ibm
   This format is used to replace part of a distinguished name. This example will change all of the distinguished name except the common name.

Further examples are listed in the `MemberFixerModule.properties` file.

2. You then run the member fixer task using the **-DaltDn** option as details in the following section.

Running the Member Fixer task:

1. Open a command prompt.

   Library parameters in steps 2 and 3:

   - The library specified in the command is the library to be scanned by the member fixer task. If the query parameter "library" is omitted, the default library that has been configured with the `defaultLibrary` property in the `WCM WCMConfigService` service is used.

2. To create a report of users or groups referenced in Web Content Manager items that need fixing, run the following command from the *wp_profile_root*`/ConfigEngine` directory:

   Windows

   ```
   ConfigEngine.bat run-wcm-admin-task-member-fixer
   -DPortalAdminId=username -DPortalAdminPwd=password
   -DWasUserId=username -DWasPassword=password -Dlibrary="MyLibrary"
   ```

   Linux

   ```
   ./ConfigEngine.sh run-wcm-admin-task-member-fixer
   -DPortalAdminId=username -DPortalAdminPwd=password
   -DWasUserId=username -DWasPassword=password -Dlibrary="MyLibrary"
   ```

   IBM i

   ```
   ConfigEngine.sh run-wcm-admin-task-member-fixer
   -DPortalAdminId=username -DPortalAdminPwd=password
   -DWasUserId=username -DWasPassword=password -Dlibrary="MyLibrary"
   ```

   Note: An administrator user name and password is not required if you have already specified the portal administrator username and password using the **PortalAdminId** and **PortalAdminPwd** settings in the `wkplc.properties` file.

   Note: Before you progress to the next step and run the member fixer task in fix mode, ensure that the report mode indicates that the updates will happen as you require. A summary of the updates will be shown by the command.

   A detailed report containing the updates that will be made for each item will be shown in the SystemOut.log file located in *wp_profile_root*`\logs\WebSphere_Portal`. If the report indicates that the update will not happen as required, change the member fixer task parameters and run the report mode again. Repeat this process until you are satisfied that the fixes will be applied correctly. This is important because the fixes made by the member fixer task when run in fix mode may not be easy to undo if incorrect fixes are applied.

3. If there have been changes to users and groups, update the items that reference them by running the run-wcm-admin-task-member-fixer task. If the member fixer task indicates that certain mismatched member conditions exist, append the specified parameters to the command.

Table 160. Member fixer conditions

Condition description	Command to correct condition
Nonexistent users or groups have alternate distinguished names (DNs) available.	• To update references to nonexistent users or groups with the DN values that are mapped in the `MemberFixerModule.properties` file, append **`-DaltDn=update`** to the command. • To remove references to nonexistent users or groups append `-DaltDn=remove` to the command.
If users or groups have invalid distinguished names (DNs) the report lists these as "invalid". This means the distinguished name doesn't exist and there is no alternate distinguished name available.	• To remove references to users and groups that have invalid distinguished names append **`-DinvalidDn=remove`** to the command. • To update references to users and groups that have invalid distinguished names with the portal administrator user distinguished name, append **`-DinvalidDn=update`** to the command.
Users or groups have been found with mismatched unique IDs.	• To fix the mismatched unique IDs append **`-DmismatchedId=update`** to the command. • To remove references to users and groups with mismatched unique IDs append **`-DmismatchedId=remove`** to the command.

Windows

```
ConfigEngine.bat run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password -Dlibrary="MyLibrary"
-Dfix=true
```

Linux

```
./ConfigEngine.sh run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password -Dlibrary="MyLibrary"
-Dfix=true
```

IBM i

```
ConfigEngine.sh run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password -Dlibrary="MyLibrary"
-Dfix=true
```

Note: An administrator user name and password is not required if you have already specified the portal administrator username and password using the **`PortalAdminId`** and **`PortalAdminPwd`** settings in the `wkplc.properties` file.

4. After the member fixer task runs, review the log output to verify that the task ran correctly. The member fixer task may not be able to save items that fail validation, such as items that contain invalid fields. You must edit these items to make them valid and then run the member fixer task again.

Running the Member Fixer in a federated security environment

In a federated security environment with multiple realms, you can specify the realm to run the member fixer task on by adding `-Drealm=realmName` to the command. If this parameter is omitted the default realm will be used.

The member fixer task will check whether there are any members and groups referenced in items that contain any of the base distinguished names defined for the specified realm and fix these references. References to members can only be updated with references to members in the specified realm.

Additionally, the member fixer task can be used to check whether there are any members and groups referenced in items that are not under any of the base distinguished names defined for any of the realms in the environment and fix these references. To do this, follow the same steps described for a single realm environment and add `-DnoRealmDn=true` to the command.

In a federated security environment with multiple realms, the member fixer task should be run for each realm in turn to make sure that all of the references are fixed.

Preserving dates

You can preserve the last modified date of items updated by the member fixer task by adding **`-DpreserveDates=true`** to the command. Otherwise, the last modified date is updated when the member fixer task is run.

Restricting which items types to fix

You can restrict which objects types are processed by appending **`-DrestrictOn=ItemType`** to the command.

For example:

- *content*
- *folder*
- *project*
- *style* for presentation templates
- *template* for authoring templates
- *taxonomy*
- *category*
- *SiteArea*
- *Workflow*
- *WorkflowStage*
- *WorkflowAction*
- *Cmpnt* for components

You can restrict multiple object types by separating the types with a comma (**,**). For example, to restrict workflows and workflow stages, you can specify **`-DrestrictOn=Workflow,WorkflowStage`**.

If not specified, all object types will be updated.

Running the task for all libraries

You can run this task for all libraries by replacing the option `-Dlibrary=`*`libraryName`* with the option **`-DallLibraries=true`** in the command. If neither option is specified, this task will process the default library that has been configured in the **WCM WCMConfigService** service using the WebSphere Integrated Solutions Console.

Running the task on a virtual portal

When running this task on a virtual portal you must add either `-DVirtualPortalHostName=`*name* or `-DVirtualPortalContext=`*context* to the command.

Parameters to set for large repositories

To prevent your session timing out before the task has finished, you can append the option `-DsessionTimeOut=`*timeOut* to the command. This sets the number of seconds in which the task must complete before its session will timeout. The default session timeout is 14,440 seconds, which is 4 hours. For large repositories you should increase this setting. For example: `-DsessionTimeOut=36000`, which is 10 hours.

Examples

These options can be combined when the conditions occur at the same time. For example, if alternate DNs are available for nonexistent users and groups and there are mismatched unique IDs, you would use the following command:

Windows

```
ConfigEngine.bat run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password -Dlibrary="MyLibrary"
-Dfix=true -DaltDn=update -DmismatchedId=update
```

Linux

```
./ConfigEngine.sh run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password-Dlibrary="MyLibrary"
-Dfix=true -DaltDn=update -DmismatchedId=update
```

IBM i

```
ConfigEngine.sh run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password -Dlibrary="MyLibrary"
-Dfix=true -DaltDn=update -DmismatchedId=update
```

If there have been changes to users and groups that are within the specified realm or that are not within any realm, update the items that reference them by entering the following command:

Windows

```
ConfigEngine.bat run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password -Drealm=MyRealm
-Dlibrary="MyLibrary" -Dfix=true -DnoRealmDn=true
```

Linux

```
./ConfigEngine.sh run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password -Drealm=MyRealm
-Dlibrary="MyLibrary" -Dfix=true -DnoRealmDn=true
```

IBM i

```
ConfigEngine.sh run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password -Drealm=MyRealm
-Dlibrary="MyLibrary" -Dfix=true -DnoRealmDn=true
```

Member fixer with syndication

You can configure your system to automatically run the member fixer tool when syndicating. The member fixer is run on the subscriber during syndication. It is run against items that have just been syndicated. Details of the member fixer operations are included in the syndication report.

To run the member fixer during syndication add or update the following properties in the `WCM WCMConfigService` service using the WebSphere Integrated Solutions Console:

Table 161. Member fixer syndication parameters

Parameter	Details
deployment.fixMembers	To enable member fixer to be run during syndication, set this parameter to *true*.
syndication.memberfixer.altDn	To update references to nonexistent users or groups with the portal administrator user distinguished name, set this parameter to *update*. To remove references to nonexistent users or groups, set this parameter to *remove*.
syndication.memberfixer.invalidDn	To update references to users or groups that have invalid distinguished names with the portal administrator user distinguished name, set this parameter to *update*. To remove references to users or groups that have invalid distinguished names, set this parameter to *remove*.
syndication.memberfixer.mismatchid	To fix references to users and groups with mismatched unique IDs, set this parameter to *update*. To remove references to users and groups with mismatched unique IDs, set this parameter to *remove*.
syndication.memberfixer.fixCase	This parameter is used to define how to treat case differences when updating or fixing distinguished names. To leave the case unchanged, set this parameter to *update*. To convert the case to lowercase, set this parameter to *lower*.
syndication.memberfixer.realm	In a federated security environment with multiple realms, you must specify the name of the realm to run the member fixer against using this parameter.
syndication.memberfixer.norealmdn	In a federated security environment with multiple realms, the member fixer task can be used to check whether there are any users and groups that are referenced in items that are not under any of the base distinguished names that are defined for the realm and fix these references. To enable this, set this parameter to *true*.

Note: The Member fixer, when run automatically via Syndication, preserves the dates of the updated items.

Member fixer task frequently asked questions

Some frequently asked question about how to use the web content member fixer task.

How can users or groups in web content items be replaced if they still exist in the user repository or LDAP?

The member fixer task cannot replace any valid users that exist in the user repository or LDAP.

How can last modified dates be preserved when web content items are updated by the member fixer task?

Use the **-DpreserveDates=true** option. See "How to use the member fixer task" on page 1178 for details.

How do I avoid session time-outs before the member fixer task is finished?

The session timeout needs to be increased for long running member fixer tasks. The default is 14,440 seconds which is 4 hours. For example, to set the session timeout to 10 hours add **-DsessionTimeout=36000** to the task request. See "How to use the member fixer task" on page 1178 for details.

How do I restrict the member fixer task to only operate on certain web content item types?

Use the **-DrestrictOn=ItemType** option. See "How to use the member fixer task" on page 1178 for details.

How can users or groups be deleted from web content items when they have been removed from the user repository or LDAP?

For example:

1. A user has left the organization and has been removed from the user repository. This user needs to be removed from web content items they previously had access to.
2. A functional group that is no longer required has been deleted from the user repository. This group needs to be removed from web content items they previously had access to.

Use the **-DinvalidDn=remove** and **-Dfix=true** options.

For example:

Windows

```
ConfigEngine.bat run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password
-Dlibrary="MyLibrary" -DinvalidDn=remove -Dfix=true
```

Linux

```
./ConfigEngine.sh run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password
-Dlibrary="MyLibrary" -DinvalidDn=remove -Dfix=true
```

IBM i

```
ConfigEngine.sh run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password
-Dlibrary="MyLibrary" -DinvalidDn=remove -Dfix=true
```

How can users or groups be replaced in web content items by a specific user or group when they have been removed from the user repository or LDAP?

For example, a user or group has been removed from the user repository. Another user or group is continuing the duties of the removed member and needs to have access to their documents.

To fix this, use the **-DaltDn=update** and **-Dfix=true** options.

For this option, it is necessary to add member mappings in the custom mapping file: *wp_profile_root*/PortalServer/wcm/shared/app/config/wcmservices/MemberFixerModule.properties

The member mappings for one user or group to be replaced with another user or group use this format: `Old DN -> New DN`

For example, to replace any instance of `cn=group1,dc=lotus,o=ibm` with `cn=group2,dc=rational,o=ibm`, use:

```
cn=group1,dc=lotus,o=ibm -> cn=group2,dc=rational,o=ibm
```

Each member mapping corresponds to one user or group to be replaced. Once all the member mappings have been made, restart the portal server and execute the member fixer task:

Windows

```
ConfigEngine.bat run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password
-Dlibrary="MyLibrary" -DaltDn=update -Dfix=true
```

Linux

```
./ConfigEngine.sh run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password
-Dlibrary="MyLibrary" -DaltDn=update -Dfix=true
```

IBM i

```
ConfigEngine.sh run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password
-Dlibrary="MyLibrary" -DaltDn=update -Dfix=true
```

How can users or groups be replaced in web content items where user common names are unchanged but their distinguished names have been changed in the user repository or LDAP?

For example, an organization unit or organization name changes in the user repository, but the common name of users and groups remain the same. There could potentially be hundreds or thousands of users affected by this organization unit change and listing each member explicitly in a custom mapping entry in the mapping file is not feasible. Only one member mapping is required when common names remain unchanged.

To fix this, use the **-DaltDn=update** and **-Dfix=true** options.

For this option, it is necessary to add member mappings in the custom mapping file: *wp_profile_root*/PortalServer/wcm/shared/app/config/wcmservices/MemberFixerModule.properties

Create member mappings for changing multiple users or groups DN with the exception of the common name portion. Once all the member mappings have been made, restart the portal server and execute the member fixer task.

For example, To replace all DNs like `cn=[ID],dc=websphere,o=ibm` with `cn=[ID],dc=tivoli,o=ibm`, use:

```
cn=[ID],dc=websphere,o=ibm -> cn=[ID],dc=tivoli,o=ibm
```

Each member mapping corresponds to one user or group to be replaced. Once all the member mappings have been made, restart the portal server and execute the member fixer task:

Windows

```
ConfigEngine.bat run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password
-Dlibrary="MyLibrary" -DaltDn=update -Dfix=true
```

Linux

```
./ConfigEngine.sh run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password
-Dlibrary="MyLibrary" -DaltDn=update -Dfix=true
```

IBM i

```
ConfigEngine.sh run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password
-Dlibrary="MyLibrary" -DaltDn=update -Dfix=true
```

How can users or groups be replaced in web content items by the administrator user when they have been removed from the user repository or LDAP?

For example:

1. A user has left the organization and has been removed from the user repository. The administrator user needs to replace the removed user in all the web content items that references that user.
2. A group has been deleted from the user repository. The administrator user needs to replace the removed group in all the web content items that references that group.

To fix this, use the **-DinvalidDn=update** and **-Dfix=true** options.

For example:

Windows

```
ConfigEngine.bat run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password
-Dlibrary="MyLibrary" -DinvalidDn=update -Dfix=true
```

Linux

```
./ConfigEngine.sh run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password
-Dlibrary="MyLibrary" -DinvalidDn=update -Dfix=true
```

IBM i

```
ConfigEngine.sh run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password
-Dlibrary="MyLibrary" -DinvalidDn=update -Dfix=true
```

Distinguished names for users and groups remain unchanged in the user repository, but users or groups cannot access web content items.

For example, web content items store a member's DN as well as a unique identifier of the LDAP entry of the user. When a member in the LDAP is

deleted and then recreated with the same details, such as the same distinguished name, the identifier in the user repository is different to the one stored in the web content Item.

To fix this, use the **-DmismatchedId=update** and **-Dfix=true** options.

For example:

Windows

```
ConfigEngine.bat run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password
-Dlibrary="MyLibrary" -DmismatchedId=update -Dfix=true
```

Linux

```
./ConfigEngine.sh run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password
-Dlibrary="MyLibrary" -DmismatchedId=update -Dfix=true
```

IBM i

```
ConfigEngine.sh run-wcm-admin-task-member-fixer
-DPortalAdminId=username -DPortalAdminPwd=password
-DWasUserId=username -DWasPassword=password
-Dlibrary="MyLibrary" -DmismatchedId=update -Dfix=true
```

Update security task

Use the update security task to apply inherited access permissions and remove existing item access permissions for all items or all items of a specific type. This task is useful as a post-migration step, or if you are applying major changes to your inheritance settings.

Running the Update Security task

1. Open a command line.
2. To apply inherited access permissions to all items in a library named "MyLibrary" for all roles, run the following command from the *wp_profile_root*/ConfigEngine directory:
   - IBM i: `ConfigEngine.sh run-wcm-admin-task-update-security -DPortalAdminId=username -DPortalAdminPwd=password -DWasPassword=password -Dlibrary=MyLibrary -DinheritPerms=apply -DlibSecurity=true`
   - Linux: `./ConfigEngine.sh run-wcm-admin-task-update-security -DPortalAdminId=username -DPortalAdminPwd=password -DWasPassword=password -Dlibrary=MyLibrary -DinheritPerms=apply -DlibSecurity=true`
   - Windows: `ConfigEngine.bat run-wcm-admin-task-update-security -DPortalAdminId=username -DPortalAdminPwd=password -DWasPassword=password -Dlibrary=MyLibrary -DinheritPerms=apply -DlibSecurity=true`

   Note: An administrator user name and password is not required if you already specified the portal administrator user name and password with the **PortalAdminId** and **PortalAdminPwd** settings in the wkplc.properties file.
3. To remove inherited access permissions to all items in a library named "MyLibrary" for all roles, run the following command:
   - IBM i: `ConfigEngine.sh run-wcm-admin-task-update-security -DPortalAdminId=username -DPortalAdminPwd=password -DWasPassword=password -Dlibrary=MyLibrary -DinheritPerms=remove`

- Linux: `./ConfigEngine.sh run-wcm-admin-task-update-security -DPortalAdminId=username -DPortalAdminPwd=password -DWasPassword=password -Dlibrary=MyLibrary -DinheritPerms=remove`
- Windows: `ConfigEngine.bat run-wcm-admin-task-update-security -DPortalAdminId=username -DPortalAdminPwd=password -DWasPassword=password -Dlibrary=MyLibrary -DinheritPerms=remove`

4. To remove existing item access permissions for all items in a library named "MyLibrary" for all roles, run the following command:
   - IBM i: `ConfigEngine.sh run-wcm-admin-task-update-security -DPortalAdminId=username -DPortalAdminPwd=password -DWasPassword=password -Dlibrary=MyLibrary -DremoveExistingPerms=true`
   - Linux: `./ConfigEngine.sh run-wcm-admin-task-update-security -DPortalAdminId=username -DPortalAdminPwd=password -DWasPassword=password -Dlibrary=MyLibrary -DremoveExistingPerms=true`
   - Windows: `ConfigEngine.bat run-wcm-admin-task-update-security -DPortalAdminId=username -DPortalAdminPwd=password -DWasPassword=password -Dlibrary=MyLibrary -DremoveExistingPerms=true`

Running the Update Security task for all libraries

You can run the Update Security task for all libraries by replacing the option **-Dlibrary**=*libraryName* with the option **-DallLibraries=true** in the command. If neither option is specified, the Update Security task processes the default library.

Restricting the task to update only specified items types

You can restrict which objects types are processed by appending **-DrestrictOn=ItemType** to the command. For example:

- *content*
- *folder*
- *project*
- *style* for presentation templates
- *template* for authoring templates
- *taxonomy*
- *category*
- *SiteArea*
- *Workflow*
- *WorkflowStage*
- *WorkflowAction*
- *Cmpnt* for components

If not specified, the security of all object types is updated.

Running the task on a virtual portal

When you run this task on a virtual portal, you must add either **-DVirtualPortalHostName**=*name* or**-DVirtualPortalContext**=*virtual_portal_context* to the command.

Preserving dates

You can preserve the last modified date of items that are updated by the update security task by adding **-DpreserveDates=true** to the command. Otherwise, the last modified date is updated when the update security task is run.

Defining the session timeout

To prevent your session from timing out before the task finishes, you can append the option **-DsessionTimeOut**=*timeOut* to the command. This parameter sets the number of seconds in which the task must complete before its session will timeout. The default session timeout is 14,440 seconds, which is 4 hours. For large repositories, increase this setting. For example: **-DsessionTimeOut=36000**, which is 10 hours.

Examples

All of the options can be combined. For instance, run the following task to accomplish the following tasks:

- Remove existing item access permissions
- Apply inherited access permissions to content in the library called 'MyLibrary'
- Preserve the last modified dates of the items
- IBM i: `ConfigEngine.sh run-wcm-admin-task-update-security -DPortalAdminPwd=`*`password`* `-Dlibrary=`*`MyLibrary`* `-DremoveExistingPerms=true -DinheritPerms=apply -DrestrictOn=Content -DpreserveDates=true`
- Linux: `./ConfigEngine.sh run-wcm-admin-task-update-security -DPortalAdminPwd=`*`password`* `-Dlibrary=`*`MyLibrary`* `-DremoveExistingPerms=true -DinheritPerms=apply -DrestrictOn=Content -DpreserveDates=true`
- Windows: `ConfigEngine.bat run-wcm-admin-task-update-security -DPortalAdminPwd=`*`password`* `-Dlibrary=`*`MyLibrary`* `-DremoveExistingPerms=true -DinheritPerms=apply -DrestrictOn=Content -DpreserveDates=true`

Managing workflows by using the workflow checker tool

Use the workflow checker tool to update workflow security settings, or reschedule pending workflow actions.

Before you begin

You must first enable the workflow update tool by adding the following parameters to the **WCM WCMConfigService** service by using the WebSphere Integrated Solutions Console:

```
connect.businesslogic.module.workflowcontrolchecker.class=com.aptrix.pluto.workflow.WorkflowContro
connect.businesslogic.module.workflowcontrolchecker.remoteaccess=true
connect.businesslogic.module.workflowcontrolchecker.autoload=false
```

Procedure

1. Log in to the portal as an administrator.
2. Open one of the following URLs in the browser:

   ```
   http://[HOST]:[PORT]/wps/wcm/myconnect/?MOD=WorkflowControlChecker
   &library=libraryname
   &updateDocSecurity=true&fix=true
   ```

   ```
   http://[HOST]:[PORT]/wps/wcm/myconnect/?MOD=WorkflowControlChecker
   &library=libraryname
   &rescheduleActions=true&fix=true
   ```

Note: If the "`library`" parameter is omitted, the default library that is configured in the **WCM WCMConfigService** service is used.

Note: If the "`&fix=true`" parameter is omitted, the tool runs in read-only mode and generates a report.

Results

&library
Enter a library name. If the `library` parameter is omitted, the default library that is configured in the **WCM WCMConfigService** service by using the WebSphere Integrated Solutions Console.

To run this tool against all libraries, you instead use `&alllibraries=true`. If you have many libraries, this process can take a long time to run, so it might be better to run this tool against individual libraries instead of all libraries.

&updateDocSecurity
If set to true, the workflowed item is saved and the workflow stage security is reapplied. This parameter is set to false by default.

&rescheduleActions
When set to true, workflow actions are rescheduled to run again. This parameter is set to false by default.

&restrictOn
Enter the items types to run the workflow checker tool against. Valid parameters are:

- `Content`
- `Cmpnt`
- `Template`
- `Site`
- `SiteArea`
- `Style`
- `Taxonomy`
- `Category`
- `Workflow`
- `WorkflowStage`
- `WorkflowAction`

&numberOfJcrOperationsPerLogin
This parameter is used to define the number of operations for 1 JCR login. Used to help with memory consumption.

&preserve_dates
When set to true, the current last modified date of the item is preserved. This parameter is set to false by default.

Running the tool on a virtual portal

There are two methods available when the tool is run on a virtual portal:

Using the URL context of a virtual portal:

```
http://[HOST]:[PORT]/wps/wcm/myconnect/[url_context]?MOD=WorkflowControlChecker
&library=libraryname
&updateDocSecurity=true&fix=true
```

Using the host name of a virtual portal:

```
http://[Virtual_HOST]:[PORT]/wps/wcm/myconnect?MOD=WorkflowControlChecker
&library=libraryname
&updateDocSecurity=true&fix=true
```

Updating workflows by using the workflow update tool

Use the workflow update tool to add a workflow to existing items that aren't already workflow enabled.

Before you begin

You must first enable the workflow update tool by adding the following parameters to the WCM WCMConfigService service by using the WebSphere Integrated Solutions Console:

- connect.businesslogic.module.workflowenablement.class=com.aptrix.pluto.workflow.Workflo
- connect.businesslogic.module.workflowenablement.remoteaccess=true
- connect.businesslogic.module.workflowenablement.autoload=false

Procedure

1. Log in to the portal as an administrator.
2. Open the following URL in the browser and specify which workflow you want to apply and the library that contains the items you want to apply the workflow to:

   ```
   http://[HOST]:[PORT]/wps/wcm/myconnect/?MOD=workflowenablement
   &library=libraryname
   &workflow=workflowname&fix=true
   ```

   Note: If the "library" parameter is omitted, the default library that is configured in the **WCM WCMConfigService** service is used.

   Note: If the "&fix=true" parameter is omitted, the tool runs in read-only mode and generates a report.

Results

Specifying the workflow:

If the workflow is in a different library to the workflow content, you must also specify the library name. For example: workflow=libraryName/WorkflowName

Specifying a workflow stage:

You can specify the workflow stage to move the updated items to by adding &workflowstage=*workflowstagename* to the URL. The stage that is specified here must have a status of published. You cannot assign items to stages with a status of draft. If not specified, items are assigned to the first stage with a status of published.

If the workflow stage is in a different library to the workflow content, you must also specify the library name. For example: workflowstage=libraryName/WorkflowStageName

Preserving dates:

You can preserve the last modified date of items that are updated by the Workflow update tool by adding &preserve_dates=true to the URL used to run the Workflow update tool.

Restricting which items types to fix:
You can restrict which objects types are processed by adding `&restrictOn=itemtype` to the URL used to run the Workflow update tool. For example:

- *content*
- *style* for presentation templates
- *template* for authoring templates
- *taxonomy*
- *category*
- *SiteArea*
- *Cmpnt* for components

If not specified, all object types are fixed.

library
Enter a library name. If the `library` parameter is omitted, the default library that is configured in the **WCM WCMConfigService** service by using the WebSphere Integrated Solutions Console.

To run this tool against all libraries, you instead use `&alllibraries=true`. If you have many libraries, this process can take a long time to run, so it might be better to run this tool against individual libraries instead of all libraries.

Unlocking items:
To force locked items to be unlocked while the tool is running, add `&forceUnlock=true` to the query. This setting defaults to true.

Specify a timeout in seconds:
To prevent your server from timing out before the workflow update tool finishes, you can specify `&sessionTimeOut=` to the URL. This setting is defined as the number of seconds before a session will timeout. For example: `&sessionTimeOut=36000`. The default session timeout is 14440 seconds.

Running the tool on a virtual portal

There are two methods available when the tool is run on a virtual portal:

Using the URL context of a virtual portal:
```
http://[HOST]:[PORT]/wps/wcm/myconnect/[url_context]?MOD=workflowenablement
&fix=true
```

Using the host name of a virtual portal:
```
http://[Virtual_HOST]:[PORT]/wps/wcm/myconnect?MOD=workflowenablement&fix=true
```

Clearing item history

You use the clear history tool to clear the history of an item.

Before you begin

You must first enable the clear history tool by adding the following parameters to the `WCM WCMConfigService` service by using the WebSphere Integrated Solutions Console:

- `connect.businesslogic.module.clearhistory.class=com.aptrix.history.ClearHistoryModule`
- `connect.businesslogic.module.clearhistory.remoteaccess=true`

- `connect.businesslogic.module.clearhistory.autoload=false`

Procedure

1. Log in to the portal as an administrator.
2. Open the following URL in the browser and specify details of what history details to clear:

   `http://[HOST]:[PORT]/wps/wcm/myconnect?MOD=ClearHistory&day=`*`date`*`&month=`*`month`*
   `&year=`*`year`*`&keep=`*`number_of_entries`*`&restrictOn=`*`item_type`*
   `&library=`*`library_name`*`&fix=true`

   day, month, and year
   The history details are cleared before the date specified in the day, month, and year parameters. If no date is specified, then the date defaults to one year before the current date.

   keep Specify the minimum number of history entries to keep. For example, if an item has not been updated for over a year, and you specify to clear all history entries more than a year old, but choose to keep the last five entries, all the history will be cleared except for the last five entries even though they are over a year old. If a number is not specified, then the minimum number of history entries to keep defaults to 10.

   restrictOn
   Select the item types to run the clear history tool against. If no item types are specified, all item types are processed. Use the following parameters for each item-type:
   - *content*
   - *folder*
   - *project*
   - *style* for presentation templates
   - *template* for authoring templates
   - *taxonomy*
   - *category*
   - *SiteArea*
   - *Workflow*
   - *WorkflowStage*
   - *WorkflowAction*
   - *Cmpnt* for components

   library
   Enter a library name. If the "library" parameter is omitted, the default library that is configured in the `WCM WCMConfigService` service is used.

   To run this tool against all libraries, you instead use `&alllibraries=true`. If you have many libraries, this process can take a long time to run, so it might be better to run this tool against individual libraries instead of all libraries.

   fix If omitted or set to false, a report listing which history items were cleared is displayed. If set to true, history items are cleared as specified.

Results

Note: You cannot completely clear item history. One history item always remains in an item no matter what parameters you select when the item history is cleared.

Running the tool on a virtual portal

There are two methods available when the tool is run on a virtual portal:

Using the URL context of a virtual portal:

```
http://[HOST]:[PORT]/wps/wcm/myconnect/[url_context]?MOD=ClearHistory
&fix=true
```

Using the host name of a virtual portal:

```
http://[Virtual_HOST]:[PORT]/wps/wcm/myconnect?MOD=ClearHistory&fix=true
```

Clearing version history

You use the clear versions tool to clear the version history of an item.

Before you begin

You must first enable the clear versions tool by adding the following parameters to the `WCM` `WCMConfigService` service by using the WebSphere Integrated Solutions Console:

- `connect.businesslogic.module.clearversions.class=com.aptrix.versioncontrol.ClearVersionsM`
- `connect.businesslogic.module.clearversions.remoteaccess=true`
- `connect.businesslogic.module.clearversions.autoload=false`

Procedure

1. Log in to the portal as an administrator.
2. Open the following URL in the browser and specify details of what history details to clear:

   ```
   http://[HOST]:[PORT]/wps/wcm/myconnect?MOD=ClearVersions&day=date
   &month=month&year=year&keep=number_of_entries&restrictOn=item_type
   &library=library_name&fix=true&preserve_dates=true
   ```

   day, month, and year
   The version history is cleared before the date specified in the day, month, and year parameters. If no date is specified, then the date defaults to one year before the current date.

   keep Specify the minimum number of history versions to keep. For example, if a version has not been created for over a year, and you specify to clear all versions more than a year old, but choose to keep the last five versions, all versions are cleared except for the last five versions even though they are over a year old. If a number is not specified, then the minimum number of versions to keep defaults to 10.

   restrictOn
   Select the item types to run the clear versions tool against. If no item types are specified, all item types are processed. Use the following parameters for each item-type:

   - *content*
   - *style* for presentation templates
   - *template* for authoring templates
   - *taxonomy*
   - *category*
   - *SiteArea*
   - *Workflow*

- *WorkflowStage*
- *WorkflowAction*
- *Cmpnt* for components

library
Enter a library name. If the `library` parameter is omitted, the default library that is configured in the `WCM WCMConfigService` service by using the WebSphere Integrated Solutions Console.

To run this tool against all libraries, you instead use `&alllibraries=true`. If you have many libraries, this process can take a long time to run, so it might be better to run this tool against individual libraries instead of all libraries.

fix If omitted or set to false, a report listing which versions were cleared is displayed. If set to true, versions are cleared as specified.

preserve_dates
If set to true, the last modified date of items that are updated by the module is preserved. If omitted or set to false, the last modified date is not preserved.

Results

Note: You cannot completely clear all versions. One version of an item always remains no matter what parameters you select when the version history is cleared.

Running the tool on a virtual portal

There are two methods available when the tool is run on a virtual portal:

Using the URL context of a virtual portal:
`http://[HOST]:[PORT]/wps/wcm/myconnect/[url_context]?MOD=ClearVersions`

Using the host name of a virtual portal:
`http://[Virtual_HOST]:[PORT]/wps/wcm/myconnect?MOD=ClearVersions`

Resetting the web content event log

From time to time, you might need to reset the web content event log. The event log can be reset only on a syndicator server. Any changes that are made by resetting the event log are then syndicated to its corresponding subscribers. In most cases, you reset the event log on the server you imported or migrated data onto, or on a syndicator to troubleshoot syndication problems in a syndication relationship.

Before you begin

You must first enable the reset event log module by adding the following parameters to the **WCM WCMConfigService** service by using the WebSphere Integrated Solutions Console:

- `connect.businesslogic.module.reseteventlog.class=com.ibm.workplace.wcm.services.eventlo`
- `connect.businesslogic.module.reseteventlog.remoteaccess=true`
- `connect.businesslogic.module.reseteventlog.autoload=false`

About this task

You must reset the web content event log under these circumstances:

- The contents of the repository is modified through an external mechanism such as a JCR import or some other custom application.
- As a post migration step during migration before syndication.
- To troubleshoot syndication problems such as items on the syndicator not being sent.

Note:

- Before you reset the web content event log, you must edit the `wkplc_dbtype.properties` file and ensure the DbSafeMode property is set to false. This file is located in *wp_profile_root*`/ConfigEngine/properties`.
- In clustered environments, you must reset only the event log on the primary node.
- Any objects that were purged on the syndicator since the last syndication is not purged on the subscriber. Purged objects are lost since the event log does not maintain records of objects that were deleted. To clean up purged items on a subscriber, you need to go the subscriber and manually delete them.

To reset

Run the `run-wcm-admin-task-reset-event-log` task from the *wp_profile_root*`/ConfigEngine` directory.

IBM i
```
ConfigEngine.sh run-wcm-admin-task-reset-event-log
-Dlibrary="library_name" -Dfix=true
```

Linux
```
./ConfigEngine.sh run-wcm-admin-task-reset-event-log
-Dlibrary="library_name" -Dfix=true
```

Windows
```
ConfigEngine.bat run-wcm-admin-task-reset-event-log
-Dlibrary="library_name" -Dfix=true
```

Note: If **`-Dfix=true`** is omitted, then the task runs in report-mode only.

Note: The library that is specified in the command is the library to be scanned by the reset event log task. If the query parameter "library" is omitted, the default library that is configured with the defaultLibrary property in the **WCM WCMConfigService** service is used.

Note: When you run this task on a virtual portal, you must add either `-DVirtualPortalHostName=`*name* or `-DVirtualPortalContext=`*virtual_portal_context* to the command.

Related tasks:

"Exporting and importing web content libraries" on page 1200
IBM Web Content Manager provides two methods for exporting and importing web content libraries: an export or import that operates on one library, and an export or import that creates a separate copy of a library. With either method, you can export the contents of a web content library to disk and import this data into another web content server. If you're working with a copy of a library, you can also import that library into the same web content server multiple times, resulting in a new library after each import without affecting previous copies. Exporting and importing libraries can be used to make a backup copy of a web content library and can also be used to move data between servers. However, this function cannot be used to send updates, deletes, and moves. It is only suitable for populating new

items.

The export cache settings task

Use the export cache settings task to display a summary of the current cache settings of your system.

When you run the export cache settings task, a summary of your cache settings is generated and set to the `SystemOut.log`. This summary includes the type of cache that is being used and how it is being applied. For example, basic caching per session or data caching per site.

Running the export cache settings task

1. Open a command prompt.
2. Run the following command from the *wp_profile_root*`/ConfigEngine` directory:

   Windows

   ```
   ConfigEngine.bat run-wcm-admin-task-export-cache-settings
   -DWasPassword=password -DPortalAdminId=username
   -DPortalAdminPwd=password -Dlibrary=MyLibrary
   -DinheritPerms=apply -DlibSecurity=true
   ```

   Linux

   ```
   ./ConfigEngine.sh run-wcm-admin-task-export-cache-settings
   -DWasPassword=password -DPortalAdminId=username
   -DPortalAdminPwd=password -Dlibrary=MyLibrary
   -DinheritPerms=apply -DlibSecurity=true
   ```

   IBM i

   ```
   ConfigEngine.sh run-wcm-admin-task-export-cache-settings
   -DWasPassword=password -DPortalAdminId=username
   -DPortalAdminPwd=password -Dlibrary=MyLibrary
   -DinheritPerms=apply -DlibSecurity=true
   ```

   Note: An administrator user name and password is not required if the portal administrator user name and password are specified in the **PortalAdminId** and **PortalAdminPwd** settings in the `wkplc.properties` file.

Displaying cache settings in a browser

You can also display your cache settings in a browser by using the following URL:

```
http://hostname:port/wps/wcm/connect?MOD=ExportCacheSettings&processLibraries=false
```

Related concepts:

"Cache expire parameters" on page 1842
You use the "expires" parameter in IBM Web Content Manager tags and URLs to specify how long to maintain data in the cache before it is expired. When data expires from a cache, the next request for the data will be retrieved from the original server. The **expires** parameter is not mandatory.

How to manage plug-in tag usage

Plug-in tags are used to reference rendering plug-ins in Web content. Use the `run-wcm-admin-task-tag-usage` task to find or update plug-in tags in your Web content.

Finding plug-in tags

To create a report of plug-in tags that are referenced in Web Content Manager items, run the following command from the *wp_profile_root*/ConfigEngine directory:

Windows

```
ConfigEngine.bat run-wcm-admin-task-tag-usage
-DPortalAdminId=username -DPortalAdminPwd=password
-Dfind=Plugin:PluginName -Dlibrary="MyLibrary"
```

Linux

```
./ConfigEngine.sh run-wcm-admin-task-tag-usage
-DPortalAdminId=username -DPortalAdminPwd=password
-Dfind=Plugin:PluginName -Dlibrary="MyLibrary"
```

IBM i

```
ConfigEngine.sh run-wcm-admin-task-tag-usage
-DPortalAdminId=username -DPortalAdminPwd=password
-Dfind=Plugin:PluginName -Dlibrary="MyLibrary"
```

Note: The library that is specified in the command is the library to be scanned by the task. If the query parameter "library" is omitted, the default library that is configured with the `defaultLibrary` property in the `WCM WCMConfigService` service is used.

Note: An administrator user name and password is not required if you specify the portal administrator user name and password by using the **PortalAdminId** and **PortalAdminPwd** settings in the `wkplc.properties` file.

Note: Before you progress to the next step and run the task in fix mode, ensure that the report mode indicates that the updates will happen as you require. A summary of the updates are shown by the command.

A detailed report of the updates that are made for each item is shown in the SystemOut.log file in *wp_profile_root*\logs\WebSphere_Portal. If the report indicates that the update will not happen as required, change the task parameters and run the report mode again. Repeat this process until you are satisfied that the fixes are applied correctly. This is important because the fixes made by the task when run in fix mode might not be easy to undo if incorrect fixes are applied.

Replacing plug-in tags

To replace one plug-in tag type with another when referenced in web content items, run the following command from the *wp_profile_root*/ConfigEngine directory:

Windows

```
ConfigEngine.bat run-wcm-admin-task-tag-usage
-DPortalAdminId=username -DPortalAdminPwd=password
-Dfind=Plugin:OldPluginName -Dfix=true
-Dreplace=Plugin:NewPluginName -Dlibrary="MyLibrary"
```

Linux

```
./ConfigEngine.sh run-wcm-admin-task-tag-usage
-DPortalAdminId=username -DPortalAdminPwd=password
-Dfind=Plugin:OldPluginName -Dfix=true
-Dreplace=Plugin:NewPluginName -Dlibrary="MyLibrary"
```

IBM i

```
ConfigEngine.sh run-wcm-admin-task-tag-usage
-DPortalAdminId=username -DPortalAdminPwd=password
```

```
-Dfind=Plugin:OldPluginName -Dfix=true
-Dreplace=Plugin:NewPluginName -Dlibrary="MyLibrary"
```

Note: The library that is specified in the command is the library to be scanned by the task. If the query parameter "library" is omitted, the default library that has been configured with the `defaultLibrary` property in the `WCM WCMConfigService` service is used.

Note: An administrator user name and password is not required if you specify the portal administrator user name and password by using the **`PortalAdminId`** and **`PortalAdminPwd`** settings in the `wkplc.properties` file.

Extra parameter values

Use these additional task parameters to update the parameters in a plug-in tag:

Table 162. Extra parameter values

Parameter	Description
`-DaddParam=`*ParameterName*	The name of a tag parameter to add to the 'find' tag when in fix mode.
`-DaddParamValue=`*ParameterValue*	The value of the tag parameter to add to the 'find' tag when in fix mode.
`-DupdateParam=`*ParameterName*	The name of the tag parameter to update on the 'find' tag when in fix mode.
`-DupdateParamNewName=`*NewParameterName*	The new name of the tag parameter to update on the 'find' tag when in fix mode.
`-DupdateParamValue=`*OldParameterValue*	The old value of the 'updateParam' parameter to update on the 'find' tag when in fix mode.
`-DupdateParamNewValue=`*NewParameterValue*	The new value that replaces the old value in the 'updateParam' parameter on the 'find' tag when in fix mode.
`-DremoveParam=`*ParameterName*	The name of a tag parameter to remove from the 'find' tag when in fix mode.
`-DremoveParamValue=`*ParameterValue*	The values of the updateParam to remove from the 'find' tag when in fix mode.
`-DfixBehavior=`*Behavior Parameter*	Used to determine how a fix is implemented. Valid behavior parameters are: **`allowMultipleValues`** If specified, parameters are allowed to have multiple values when you add parameter values in fix mode. **`ifParamValue:`*ParameterValue*** The fix is applied only to tags that have a parameter that has this value **`ifParamNotExist:`*ParameterName*** The fix is applied only to tags where this parameter does not exist.

Preserving dates

You can preserve the last modified date of items that are updated by the task by adding **-DpreserveDates=true** to the command. Otherwise, the last modified date is updated when the task is run.

Running the task for all libraries

You can run this task for all libraries by replacing the option `-Dlibrary=`*`libraryName`* with the option **-DallLibraries=true** in the command. If neither option is specified, this task processes the default library that is configured in the **WCM WCMConfigService** service by using the WebSphere Integrated Solutions Console.

Running the task on a virtual portal

When this task is run on a virtual portal, you must add either `-DVirtualPortalHostName=`*name* or`-DVirtualPortalContext=`*`context`* to the command.

Parameters to set for large repositories

To prevent your session timing out before the task has finished, you can append the option `-DsessionTimeOut=`*`timeOut`* to the command. This sets the number of seconds in which the task must complete before its session will timeout. The default session timeout is 14,440 seconds, which is 4 hours. For large repositories, you should increase this setting. For example: `-DsessionTimeOut=36000`, which is 10 hours.

Examples

In this example, the plug-in tag `ifEqual` has been deprecated and replaced with the plug-in tag `Equals`. To up date all instances of the `ifEqual` tag in your site, run the following commands:

```
./ConfigEngine.sh/bat run-wcm-admin-task-tag-usage -Dfind=Plugin:ifEqual -DallLibraries=true -Dfix=t
./ConfigEngine.sh/bat run-wcm-admin-task-tag-usage -Dfind=Plugin:ifEqual -DallLibraries=true -Dfix=t
./ConfigEngine.sh/bat run-wcm-admin-task-tag-usage -Dfind=Plugin:ifEqual -DallLibraries=true -Dfix=t
```

These commands replace the 'value1' and 'value2' parameters with 'text1' and 'text2' parameters, and then change the tag name to `Equals`.

Exporting and importing web content libraries

IBM Web Content Manager provides two methods for exporting and importing web content libraries: an export or import that operates on one library, and an export or import that creates a separate copy of a library. With either method, you can export the contents of a web content library to disk and import this data into another web content server. If you're working with a copy of a library, you can also import that library into the same web content server multiple times, resulting in a new library after each import without affecting previous copies. Exporting and importing libraries can be used to make a backup copy of a web content library and can also be used to move data between servers. However, this function cannot be used to send updates, deletes, and moves. It is only suitable for populating new items.

About this task

Before you begin, create an empty shared directory to hold the exported web content library. If you move data between servers, both systems must have write access to this directory. In addition, review the following considerations before you export or import web content libraries:

Importing libraries into different versions

You can import libraries from a different version of Web Content Manager so long as the version you are importing the library into is later than the version you exported the library from. For example:

- you can import a library that is exported from version 6.1.0.1 into version 7.0
- you cannot import a library that is exported from version 7.0 into version 6.1.0.1

Upgrade to the most recent version of each release before you attempt to import libraries between versions. It is not possible to export libraries from releases before 6.0.

Exporting and importing a web content library versus syndication.

This feature does not replace the syndication feature. Although this feature can be used to transfer data between servers, it is a manual process and is not meant to be used for regular updates between servers. Syndication is instead used to automatically keep two or more servers synchronized. Also, whereas syndication can be used to send updates, deletes and moves, the import feature is only suitable for populating new items.

Limitations of exporting and importing a web content library.

- Saved versions of items are not exported. Only the current version of each item is exported.
- Children are only exported and imported when the parent is successfully exported and imported.
- If an item exists on the target server with the same path, name and ID, then the item is overwritten.
- Library and item level access controls remain unchanged when a library is exported and imported. You need to run the member fixer tool on the imported library to fix references to missing users and groups.
- You cannot import an item if an item on the target server has the same ID but a different parent than the item that is being imported.
- Projects are not exported.

Disabling JCR text search.

Disable JCR text search indexing on your WebSphere Portal Express server before you export or import large libraries to reduce the load on the database during export and import.

1. Log in to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers** > **JCR ConfigService PortalContent** > **Custom properties**.

   Cluster note: If you are using this web content server as part of a cluster, ensure that you use the WebSphere Integrated Solutions Console for the deployment manager when you edit configuration properties.
3. Change **jcr.textsearch.enabled** to false.

Note: You must restart your server any time you change these settings. After you export and import your library, you must then enable JCR text search again. It can take some time to rebuild the indexes when you re-enable JCR text search indexing.

Exporting and importing large libraries

- When web content libraries are imported, a temporary directory is used to store the library files during the upload process. If the size of the uploaded files exceeds the available disk space for the temporary directory, the import operation fails. When large libraries are exported, ensure that there is sufficient disk space to accommodate the import.
  1. Log in to the WebSphere Integrated Solutions Console.
  2. Click **Resources** > **Resource Environment** > **Resource Environment Providers** > **JCR ConfigService PortalContent** > **Custom properties**.
  3. Make sure the location that is specified under **jcr.binaryValueFileDir** has sufficient disk space to accommodate the import.
- When you export or import large libraries, you might need to adjust the following settings:
  1. Log in to the WebSphere Integrated Solutions Console.
  2. Click **Servers** > **Server Types** > **WebSphere application servers** > *portal_server* > **Container Services** > **Transaction Service**.
  3. Change the **total transaction lifetime timeout** and the **maximum transaction timeout** settings to 360 seconds.

Personalization components.

Personalization rules that are created within a Personalization component are exported and imported along with your web content library.

If you are using Personalization rules that are created directly in the Personalization portlet, you need to export and import your rules to and from Personalization on the same servers as your web content by using the same process as moving WebSphere Portal content from a staging system to a production system. Personalization export and import must be run before you export and import web content.

JSP components

If you are using JSP components that you must manually copy any related JSP files to and from the same servers that you are exporting and importing to.

"Exporting and importing a web content library" on page 1203
You can export the contents of a web content library to disk and import this data into another web content server. Use this feature to make a backup copy of a web content library, and to move data between servers. This function cannot be used to send updates, deletes, and moves. It is only suitable for populating new items.

"Exporting and importing a web content library copy" on page 1207
You can export the contents of a web content library to disk by creating a copy of the web content library. By working with an exported copy, you can import the copied library into the same web content server multiple times, resulting in a new library after each import without affecting previous copies. This is a quick way of creating new libraries that are fully populated with web content that you can easily adapt for other purposes.

Related tasks:

"Resetting the web content event log" on page 1195
From time to time, you might need to reset the web content event log. The event log can be reset only on a syndicator server. Any changes that are made by resetting the event log are then syndicated to its corresponding subscribers. In most cases, you reset the event log on the server you imported or migrated data onto, or on a syndicator to troubleshoot syndication problems in a syndication relationship.

"Exporting and importing a web content library"
You can export the contents of a web content library to disk and import this data into another web content server. Use this feature to make a backup copy of a web content library, and to move data between servers. This function cannot be used to send updates, deletes, and moves. It is only suitable for populating new items.

"Exporting and importing a web content library copy" on page 1207
You can export the contents of a web content library to disk by creating a copy of the web content library. By working with an exported copy, you can import the copied library into the same web content server multiple times, resulting in a new library after each import without affecting previous copies. This is a quick way of creating new libraries that are fully populated with web content that you can easily adapt for other purposes.

Exporting and importing a web content library

You can export the contents of a web content library to disk and import this data into another web content server. Use this feature to make a backup copy of a web content library, and to move data between servers. This function cannot be used to send updates, deletes, and moves. It is only suitable for populating new items.

About this task

Follow these steps to export and import a web content library. The server that the data is being exported from is called the source server, and the server that the data is being imported into is called the target server.

Procedure

1. **Exporting:**
   a. Log in to the WebSphere Integrated Solutions Console on the source server.
   b. Click **Resources** > **Resource Environment** > **Resource Environment Providers** > **WCM WCMConfigService** > **Custom properties**.

      Cluster note: If you are using this web content server as part of a cluster, ensure that you use the WebSphere Integrated Solutions Console for the deployment manager when you are manipulating configuration properties.
   c. Create or update the export properties.

      export.directory
      The directory on the source server where the exported data is written. The export task creates a subdirectory with the name corresponding to library name within this directory for each exported library. The default value is `${USER_INSTALL_ROOT}/PortalServer/wcm/ilwwcm/system/export`.

      export.libraryname
      The name of the web content library to transfer. If you are exporting multiple libraries enter each library name, separated by a semi-colon. For example: `Lib_1;Lib_2;Lib_3`

Note: The library names that are specified in this parameter must be the original name of the library, not the localized name. Click the **Administration menu** icon. Then, click **Portal Content** > **Web Content Libraries**. You can view the original name and view the edit settings of the library.

export.singledirectory
If set to `true`, multiple libraries are written into a single directory that is specified by the `export.directory` property. If set to `false`, the export task created subdirectories with the name corresponding to each exported library names. For example, if `export.directory` is specified as `C:\export` and the library name is `Web Library`, the export task saves the exported library under `C:\export\Web Library`. Set this property to `true` when you are exporting multiple libraries that contain references between each library.

Note: You must restart your server any time you change these settings.

d. Export the web content library from the source server:
   - Open a command prompt on the source server.
   - Run the `export-wcm-data` task from the *wp_profile_root*`/ConfigEngine` directory.

     IBM i
     ```
     ConfigEngine.sh export-wcm-data -DWasPassword=password
     -DPortalAdminPwd=password
     ```

     Linux
     ```
     ./ConfigEngine.sh export-wcm-data -DWasPassword=password
     -DPortalAdminPwd=password
     ```

     Windows
     ```
     ConfigEngine.bat export-wcm-data -DWasPassword=password
     -DPortalAdminPwd=password
     ```

     By default, this task is done on the base portal. To run this task on a different virtual portal, identify the virtual portal by adding one of the following parameters to the command line. Each parameter requires the prefix `-D` on the command line.

     `VirtualPortalHostName`
     Specify the host name of the virtual portal. For example, `vp.example.com`.

     `VirtualPortalContext`
     Specify the virtual portal context that identifies the virtual portal. For example, `vp1`.

     Note:
     - You can override the `export.directory` property that is defined in the `WCM WCMConfigService` service by using the `-Dexport.directory` parameter. For example: `export-wcm-data -Dexport.directory=c:\export`
     - You can override the `export.singledirectory` property that is defined in the `WCM WCMConfigService` service by using the `-Dexport.singledirectory` parameter. For example: `export-wcm-data -Dexport.singledirectory=false` saves the exported libraries under different directories.
     - You can override the `export.libraryname` property that is defined in the `WCM WCMConfigService` service by using the `-Dexport.libraryname` parameter. For example: `export-wcm-data -Dexport.libraryname=`*`libraryname`*

- You can override the `export.libraryname` property that is defined in the `WCM WCMConfigService` service by adding the option `-Dexport.allLibraries` parameter to export all libraries. If this option is used, the export might take a long time to finish.

Important: To ensure that your exported libraries can be successfully imported, do not change the names of any of the folders or files within the exported data.

e. Verify that this transfer step completed without errors. If any errors occurred, check the portal logs on the target server for extended diagnostic information.

f. Verify that the export directories were populated correctly, including any subdirectories for each exported library.

2. **Importing:**

a. Log in to the WebSphere Integrated Solutions Console on the target server.

b. Click **Resources** > **Resource Environment** > **Resource Environment Providers** > **WCM WCMConfigService** > **Custom properties**.

Cluster note: If you are using this web content server as part of a cluster, ensure that you use the WebSphere Integrated Solutions Console for the deployment manager when you are manipulating configuration properties.

c. Create or update the `import.directory` property. This directory is from where the exported data is read when you are importing the data to the target server. If you are exporting and importing across a network, this property can be the same directory as the one specified in `export.directory` property. Otherwise, you must copy the exported data from the location that is specified in the `export.directory` property to the location specified in the `import.directory` property before you run the import task in step 2.

- If you specified `true` for the `export.singledirectory` property when you exported your libraries, specify the parent directory where all the exported libraries are located.
- If you specified `false` for the `export.singledirectory` property when you exported your libraries, or if you want to import only specific libraries, then you must list the directory of each library, separated by semicolons. For example: `c:\import\Lib1;c:\import\Lib2;c:\import\Lib3`. If you are using Linux use `/;` to separate each library, such as `/opt/importdata/Lib1/;/opt/importdata/Lib2/;/opt/importdata/Lib3`.

Note: You must restart your server any time you change this setting.

d. Import the web content library to the target server.

- Open a command prompt on the target server.
- Run the `import-wcm-data` task from the *wp_profile_root*`/ConfigEngine` directory.

IBM i
```
ConfigEngine.sh import-wcm-data -DWasPassword=password
-DPortalAdminPwd=password
```

Linux
```
./ConfigEngine.sh import-wcm-data -DWasPassword=password
-DPortalAdminPwd=password
```

Windows
```
ConfigEngine.bat import-wcm-data -DWasPassword=password
-DPortalAdminPwd=password
```

By default, this task is done on the base portal. To run this task on a different virtual portal, identify the virtual portal by adding one of the following parameters to the command line. Each parameter requires the prefix `-D` on the command line.

`VirtualPortalHostName`
Specify the host name of the virtual portal. For example, `vp.example.com`.

`VirtualPortalContext`
Specify the virtual portal context that identifies the virtual portal. For example, `vp1`.

Note: You can override the `import.directory` property that is defined in the `WCM WCMConfigService` service by using the `-Dimport.directory` parameter. For example: `import-wcm-data -Dimport.directory=c:\import\Lib1;c:\import\Lib2;c:\import\Lib3`.

Differences in paths between versions:

When you are exporting from a version 6.1 system, you can specify a folder to export the library to:

```
/opt/61/folder/jcr_root
```

When you are importing into version 8.0 from version 6.1, the jcr_root is not required to be specified in the import path:

```
/opt/61/folder/
```

When you are exporting from versions 7.0 or higher, the following structure is used:

```
/opt/70/folder1/folder2
```

When exported, `folder2` is automatically generated.

When you are importing into version 8.0 from version 7.0 or higher, `folder2` is not required to be specified in the import path:

```
/opt/70/folder1/
```

e. Verify that the imported libraries are imported by reviewing the list of libraries that are listed in the web content libraries section of the administration portlet on the target server. If any errors occurred, check the portal logs on the target server for extended diagnostic information.
f. Reset the web content event log.
g. Restart the server.

What to do next

Troubleshooting:

- If items are exported and imported twice between the same servers, and items are moved or deleted between the first and second export and import, then you must manually delete these items from the target server before you transfer the items again. If this step is not done, an error like this example is generated:

  ```
  javax.jcr.ItemExistsException: A node already exists with uuid:
  376dba00408608aea231b2c714d0bda6 at path: /contentRoot/icm:libraries[10]/
  F1/F3/test1.ort
  ```

- If you receive 500 errors on ext2 and ext3 versions of Linux, you exceeded the number of children that a parent folder can hold. You cannot store more than 32768 children under one folder on ext2 and ext3 versions of Linux. Move some content items out of the affected site area to another site area so that none of your site areas contain more than 32768 children under one folder and then try exporting again. You can move the content items back to the correct site areas when you import the library.

Related tasks:

"Exporting and importing web content libraries" on page 1200
IBM Web Content Manager provides two methods for exporting and importing web content libraries: an export or import that operates on one library, and an export or import that creates a separate copy of a library. With either method, you can export the contents of a web content library to disk and import this data into another web content server. If you're working with a copy of a library, you can also import that library into the same web content server multiple times, resulting in a new library after each import without affecting previous copies. Exporting and importing libraries can be used to make a backup copy of a web content library and can also be used to move data between servers. However, this function cannot be used to send updates, deletes, and moves. It is only suitable for populating new items.

Exporting and importing a web content library copy

You can export the contents of a web content library to disk by creating a copy of the web content library. By working with an exported copy, you can import the copied library into the same web content server multiple times, resulting in a new library after each import without affecting previous copies. This is a quick way of creating new libraries that are fully populated with web content that you can easily adapt for other purposes.

About this task

Although many aspects are the same for the standard export and import and the copy export and import, there are some important differences:

- When you export a library as a copy, new IDs are generated for all items in the library. This ensures that there are no conflicts with existing libraries or items when you import the copy into a web content server that already contains the original library. In this way, you can run multiple imports to the same web content server, resulting in a new library for each import.
- The configuration tasks (`export-library-copy` and `import-library-copy`) that work on library copies, use properties that can either be added to the `wkplc.properties` file or manually appended to the command line, for easier scripting.

Follow these steps to export or import a copy of a web content library. The server that the data is being exported from is called the source server, and the server that the data is being imported into is called the target server.

Procedure

- **Exporting:**
  1. Open a command prompt on the source server.
  2. Run the `export-library-copy` task from the *`wp_profile_root`*`/ConfigEngine` directory.

     Windows

     ```
     ConfigEngine.bat export-library-copy
     ```

```
-DLibraryPath=path_to_export_file
-DLibraryName=library_name_to_export -DWasPassword=password
-DPortalAdminPwd=password
```

Linux
```
./ConfigEngine.sh export-library-copy
-DLibraryPath=path_to_export_file
-DLibraryName=library_name_to_export -DWasPassword=password
-DPortalAdminPwd=password
```

IBM i
```
ConfigEngine.sh export-library-copy
-DLibraryPath=path_to_export_file
-DLibraryName=library_name_to_export -DWasPassword=password
-DPortalAdminPwd=password
```

The following properties must be specified either on the command line or in the `wkplc.properties` file.

Note: If you are specifying properties in the `wkplc.properties` file, it is not necessary to put quotation marks (") around values that contain spaces. These quotation marks are only required when specifying property values on the command line.

LibraryPath
The directory path and file name that is used to store the exported library. The export process creates a compressed archive file, so it is recommended that you specify a file extension such as `.zip`. If you are exporting and importing across a network, this location can be a network drive accessible by both the source and target servers.

LibraryName
The name of the web content library to copy. If you are exporting multiple libraries, separate each library name by a semi-colon (`;`). For example, `LibraryName="Web Content;Samples"`.

WasUserid
The administrator ID for WebSphere Application Server.

WasPassword
The administrator password for WebSphere Application Server.

PortalAdminId
The administrator ID for WebSphere Portal Express.

PortalAdminPwd
The administrator password for WebSphere Portal Express.

By default, this task is performed on the default virtual portal. To run this task on a different virtual portal, identify the virtual portal by adding one of the following parameters to the command line. Each parameter requires the prefix `-D` on the command line.

VirtualPortalHost
Specify the host name of the virtual portal. For example, `vp.example.com`.

Important: If the host name of the virtual portal is the same as the host name of the default virtual portal, you must also specify the **`VirtualPortalContext`** property. You can specify the **`VirtualPortalHost`** property by itself only if the host name is unique.

VirtualPortalContext
Specify the virtual portal context that identifies the virtual portal. For example, `vp1`.

Example commands:

– Windows: `ConfigEngine.bat export-library-copy -DLibraryPath=C:\wcm_export\webcontent.zip -DLibraryName="Web Content" -DWasPassword=mypassword -DPortalAdminPwd=mypassword -DVirtualPortalHost=vp.example.com`

– Linux: `./ConfigEngine.sh export-library-copy -DLibraryPath=/opt/wcm_export/webcontent.zip -DLibraryName="Web Content" -DWasPassword=mypassword -DPortalAdminPwd=mypassword -DVirtualPortalHost=vp.example.com`

– IBM i: `ConfigEngine.sh export-library-copy -DLibraryPath=/opt/wcm_export/webcontent.zip -DLibraryName="Web Content" -DWasPassword=mypassword -DPortalAdminPwd=mypassword -DVirtualPortalHost=vp.example.com`

3. Verify that this transfer step completed without errors. If any errors occurred, check the portal logs on the source server for extended diagnostic information.

- **Importing:**
  1. Open a command prompt on the target server.
  2. Run the `import-library-copy` task from the *wp_profile_root*`/ConfigEngine` directory.

     Windows
     `ConfigEngine.bat import-library-copy -DLibraryPath=`*path_to_export_file* `-DLibraryName=`*library_name_to_import* `-DWasPassword=`*password* `-DPortalAdminPwd=`*password*

     Linux `./ConfigEngine.sh import-library-copy -DLibraryPath=`*path_to_export_file* `-DLibraryName=`*library_name_to_import* `-DWasPassword=`*password* `-DPortalAdminPwd=`*password*

     IBM i `ConfigEngine.sh import-library-copy -DLibraryPath=`*path_to_export_file* `-DLibraryName=`*library_name_to_import* `-DWasPassword=`*password* `-DPortalAdminPwd=`*password*

     The following properties must be specified either on the command line or in the `wkplc.properties` file.

     Note: If you are specifying properties in the `wkplc.properties` file, it is not necessary to put quotation marks (") around values that contain spaces. These quotation marks are only required when specifying property values on the command line.

     Differences in paths between versions:

     When exporting from a version 6.1 system, you can specify a folder to export the library to:

     `/opt/61/folder/jcr_root`

     When importing into version 8.0 from version 6.1, the jcr_root is not required to be specified in the import path:

`/opt/61/folder/`

When exporting from versions 7.0 or higher, the following structure is used:
`/opt/70/folder1/folder2`

When exported, `folder2` is automatically generated.

When importing into version 8.0 from version 7.0 or higher, `folder2` is not required to be specified in the import path:
`/opt/70/folder1/`

LibraryPath
The directory path and file name containing the library to be imported. If you are exporting and importing across a network, the value for this property can be the same file path that was used for the `LibraryPath` property during the export process. Otherwise, you must copy the exported data to a location accessible by the target server before attempting to import.

LibraryName
The name to use for the web content library copy that you are importing. If you are importing multiple libraries, separate each new library name by a semi-colon (`;`). For example, `LibraryName="Web Content Copy;Samples Copy"`.

LibraryExportName
The sequence of library names used during the original export, as defined by the `LibraryName` property specified for the export process. For example, `LibraryExportName="Web Content;Samples"`.

This property enables the import process to correctly set the new library names for the imported copies, in conjunction with the `LibraryName` property specified for the import process. The `LibraryExportName` property is only required if you are importing multiple libraries at one time.

WasUserid
The administrator ID for WebSphere Application Server.

WasPassword
The administrator password for WebSphere Application Server.

PortalAdminId
The administrator ID for WebSphere Portal Express.

PortalAdminPwd
The administrator password for WebSphere Portal Express.

By default, this task is performed on the default virtual portal. To run this task on a different virtual portal, identify the virtual portal by adding one of the following parameters to the command line. Each parameter requires the prefix `-D` on the command line.

VirtualPortalHost
Specify the host name of the virtual portal. For example, `vp.example.com`.

Important: If the host name of the virtual portal is the same as the host name of the default virtual portal, you must also specify the **`VirtualPortalContext`** property. You can specify the **`VirtualPortalHost`** property by itself only if the host name is unique.

VirtualPortalContext
Specify the virtual portal context that identifies the virtual portal. For example, `vp1`.

The following properties are optional and can be specified either on the command line or in the `wkplc.properties` file:

LibraryTitle
The title to use for the web content library copy that you are importing. If you are importing multiple libraries, separate each new library title by a semi-colon (`;`). For example, `LibraryTitle="Web Content Title;Samples Title"`.

LibraryDescription
The description to use for the web content library copy that you are importing. If you are importing multiple libraries, separate each new library description by a semi-colon (`;`). For example, `LibraryDescription="Copy of Web Content library;Copy of Samples library"`.

LibraryNameTextProvider
This property specifies the name of the text provider to use to locate the translated title of the library that you are importing. If you are importing multiple libraries with different text providers, separate each provider name by a semi-colon (`;`). For example, `LibraryNameTextProvider=provider1;provider2`.

LibraryNameTextProviderKey
This property specifies the key within the associated text provider that identifies the translated title of the library that you are importing. If you are importing multiple libraries with different translated titles, separate each key by a semi-colon (`;`). For example, `LibraryNameTextProviderKey=key1;key2`.

LibraryBaseLocale
The locale used when importing the web content library copy.

Example commands:

- Windows: `ConfigEngine.bat import-library-copy -DLibraryPath=C:\wcm_import\webcontent.zip -DLibraryName="Web Content Copy" -DLibraryTitle="Web Content Copy Title" -DLibraryDescription="Copy of Web Content library" -DLibraryNameTextProvider=provider -DLibraryNameTextProviderKey=key -DLibraryBaseLocale=en -DWasPassword=mypassword -DPortalAdminPwd=mypassword`
- Linux: `./ConfigEngine.sh import-library-copy -DLibraryPath=/opt/wcm_import/webcontent.zip -DLibraryName="Web Content Copy" -DLibraryTitle="Web Content Copy Title" -DLibraryDescription="Copy of Web Content library" -DLibraryNameTextProvider=provider -DLibraryNameTextProviderKey=key -DLibraryBaseLocale=en -DWasPassword=mypassword -DPortalAdminPwd=mypassword`
- IBM i: `ConfigEngine.sh import-library-copy -DLibraryPath=/opt/wcm_import/webcontent.zip -DLibraryName="Web Content Copy" -DLibraryTitle="Web Content Copy Title" -DLibraryDescription="Copy of Web Content library" -DLibraryNameTextProvider=provider -DLibraryNameTextProviderKey=key -DLibraryBaseLocale=en -DWasPassword=mypassword -DPortalAdminPwd=mypassword`

3. Verify that the imported libraries have been imported by reviewing the list of libraries that are listed in the web content libraries section of the

administration portlet on the target server. If any errors occurred, check the portal logs on the target server for extended diagnostic information.

4. Reset the web content event log.

Example commands for exporting and importing multiple libraries

When exporting and importing multiple web content libraries with a single command, the following considerations apply:

- For properties such as `LibraryName` that reference multiple libraries, separate the respective values for that property with a semi-colon (`;`).
- The value of the `LibraryExportName` property must match the value of the `LibraryName` property used during the export process to indicate the sequence of libraries.

Windows

- Export: `ConfigEngine.bat export-library-copy -DLibraryPath=C:\wcm_export\webcontent.zip -DLibraryName="Web Content;Samples" -DWasPassword=mypassword -DPortalAdminPwd=mypassword`
- Import: `ConfigEngine.bat import-library-copy -DLibraryPath=C:\wcm_import\webcontent.zip -DLibraryName="Web Content Copy;Samples Copy" -DLibraryExportName="Web Content;Samples" -DLibraryTitle="Web Content Copy Title;Samples Copy Title" -DLibraryBaseLocale=en -DWasPassword=mypassword -DPortalAdminPwd=mypassword`

Linux

- Export: `./ConfigEngine.sh export-library-copy -DLibraryPath=/opt/wcm_export/webcontent.zip -DLibraryName="Web Content;Samples" -DWasPassword=mypassword -DPortalAdminPwd=mypassword`
- Import: `./ConfigEngine.sh import-library-copy -DLibraryPath=/opt/wcm_import/webcontent.zip -DLibraryName="Web Content Copy;Samples Copy" -DLibraryExportName="Web Content;Samples" -DLibraryTitle="Web Content Copy Title;Samples Copy Title" -DLibraryBaseLocale=en -DWasPassword=mypassword -DPortalAdminPwd=mypassword`

IBM i

- Export: `ConfigEngine.sh export-library-copy -DLibraryPath=/opt/wcm_export/webcontent.zip -DLibraryName="Web Content;Samples" -DWasPassword=mypassword -DPortalAdminPwd=mypassword`
- Import: `ConfigEngine.sh import-library-copy -DLibraryPath=/opt/wcm_import/webcontent.zip -DLibraryName="Web Content Copy;Samples Copy" -DLibraryExportName="Web Content;Samples" -DLibraryTitle="Web Content Copy Title;Samples Copy Title" -DLibraryBaseLocale=en -DWasPassword=mypassword -DPortalAdminPwd=mypassword`

Related tasks:

"Exporting and importing web content libraries" on page 1200
IBM Web Content Manager provides two methods for exporting and importing web content libraries: an export or import that operates on one library, and an export or import that creates a separate copy of a library. With either method, you can export the contents of a web content library to disk and import this data into another web content server. If you're working with a copy of a library, you can also import that library into the same web content server multiple times, resulting in a new library after each import without affecting previous copies. Exporting and importing libraries can be used to make a backup copy of a web content library and can also be used to move data between servers. However, this function cannot be used to send updates, deletes, and moves. It is only suitable for populating new items.

Deleting libraries by using the delete libraries tool

Use the delete libraries tool to delete multiple libraries, even where there are references between the selected libraries.

Before you begin

- Back up your database before the module is run as a precaution.
- You must first enable the delete libraries tool by adding the following parameters to the **WCM WCMConfigService** service by using the WebSphere Integrated Solutions Console:
  –
  `connect.businesslogic.module.deletemultiplelibraries.class=com.aptrix.pluto.util.Dele`
  – `connect.businesslogic.module.deletemultiplelibraries.remoteaccess=true`
  – `connect.businesslogic.module.deletemultiplelibraries.autoload=false`

  You must also edit the **connect.businesslogic.module** property, and add *deletemultiplelibraries* to the end of the comma-separated list.

Procedure

1. Log in to the portal as an administrator.
2. Open the following URL in the browser and specify which libraries you want to delete:

   `http://[HOST]:[PORT]/wps/wcm/myconnect/?MOD=deletemultiplelibraries &libraries=`*`libraryname1`*`,`*`libraryname2`*

   There are two methods available when the tool is run on a virtual portal:

   Using the URL context of a virtual portal:

   `http://[HOST]:[PORT]/wps/wcm/myconnect/[url_context]?MOD=deletemultiplelibraries &libraries=`*`libraryname1`*`,`*`libraryname2`*

   Using the host name of a virtual portal:

   `http://[Virtual_HOST]:[PORT]/wps/wcm/myconnect?MOD=deletemultiplelibraries &libraries=`*`libraryname1`*`,`*`libraryname2`*

Running the tool using the configuration engine

Run the `run-wcm-admin-task-delete-libraries` task from the *`wp_profile_root`*`/ConfigEngine` directory.

IBM i
```
ConfigEngine.sh run-wcm-admin-task-delete-libraries
-Dlibraries="library1,library2" -DPortalAdminId=username
-DPortalAdminPwd=password
```

Linux
```
./ConfigEngine.sh run-wcm-admin-task-delete-libraries
-Dlibraries="library1,library2" -DPortalAdminId=username
-DPortalAdminPwd=password
```

Windows
```
ConfigEngine.bat run-wcm-admin-task-delete-libraries
-Dlibraries="library1,library2" -DPortalAdminId=username
-DPortalAdminPwd=password
```

Note: When you run this task on a virtual portal, you must add either `-DVirtualPortalHostName=`*name* or `-DVirtualPortalContext=`*context* to the command.

How to clone a web content repository

Syndicating items from one server to another, either after migration or to roll out a new server, can take a long time. Your database backup and restore features can be used to clone data from one repository to another, making your system ready for syndication to be used from then on for incremental updates.

There are two basic cloning scenarios:

- Cloning all items from one server to another. For example, cloning data from one authoring server to another authoring server.
- Cloning all items from one server to another, but then removing unrequired data from the cloned server. For example, cloning data from an authoring server to a delivery server where you would want to remove version history and draft items from the delivery server repository.

Note: These procedures describe how to clone a web content repository only. To clone a WebSphere Portal environment, XMLaccess export and import are used to transfer the WebSphere Portal data to the target environment

Note: Do not attempt to clone the JCR database if managed pages are enabled.

"Cloning preparation"
You must prepare your source and target systems before a web content repository is cloned.

"Cloning data" on page 1215
These procedures describe how to clone web content data from one system to another.

Cloning preparation

You must prepare your source and target systems before a web content repository is cloned.

- Managed pages must be disabled on both the source and target environments if virtual portals are in use before the cloning procedure is begun.
- The source and target environments must be on the same version and fix level
- Ideally the source and target environments use the same LDAP
- If the target server already contains data:
  - If you need to use this data later, ensure that you take a backup of the target environment before cloning.
  - Note down the syndicator and subscriber setups. The syndication setup on the target environment is lost during the cloning process, and needs to be re-created.

– Note down the library access control setup. Library access levels for target environment are lost during the cloning process, and need to be reapplied.

Oracle databases

When you set up an Oracle database for JCR, you must have a separate physical Oracle database for each JCR repository. This setup makes it easy to copy a JCR repository from one system to another. If you do choose to store all of your JCR repositories in a single database, then you must use different schema names for each system. WebSphere Portal does not support more than one instance of WebSphere Portal running against a single JCR database or schema.

Cloning data

These procedures describe how to clone web content data from one system to another.

Before you begin

Note: Managed pages must be disabled on both the source and target environments if virtual portals are in use before you begin the cloning procedure.

Procedure

1. On the source system:
   a. Disable all syndicators
   b. Stop the Portal server
   c. Back up the data of JCR database name that is specified by "`jcr.DbName`" parameter in the `wkplc_dbdomain.properties` file:

Table 163. `wkplc_dbdomain.properties`

System	Location
Windows	*wp_profile_root*`\ConfigEngine\properties` directory.
Linux	*wp_profile_root*`/ConfigEngine/properties` directory.
IBM i	*wp_profile_root*`/ConfigEngine/properties`

      Refer to the documentation for your database for specific backup instructions.
   d. Restart the Portal server.
   e. Re-enable any syndicators that do not syndicate to the target server.
2. On the target system:
   a. Stop the Portal server.
   b. Remove the existing JCR database by dropping the database. Refer to the documentation for your database for specific instructions.
   c. Create a new database, and restore the source system database backup. Refer to the documentation for your database for specific instructions.
   d. Restart the Portal server.
   e. Delete all syndicators and subscribers as they are not valid for the target system.
   f. If the target system is using a different LDAP run the member fixer tool to fix member information to match the new LDAP. First run the module in

report mode to see what member information requires fixing and then run the tool in fix mode to fix various potential member information mismatches.

g. If you are cloning from an authoring system to a delivery system, run the clear versions tool to remove any versions from the delivery system.

h. Set up syndication:
   - Create subscribers and syndicators for this system.
   - If the target system is a syndicator to the original source system, leave that syndicator disabled for now. All other Syndicators can be enabled

3. On source system:
   a. If the target system is a subscriber to the source system, update the syndicator with the new target subscriber ID and enable syndication.
   b. If the target system is a syndicator to the source system, update the subscriber with the new target syndicator ID
4. On target system:
   a. If the target system is a syndicator to the source system, enable syndication.

What to do next

When you finish cloning your web content data:

- Validate that the web content data on the target environment is rendering correctly.
- Validate access control settings for both rendering and authoring are set as expected, and working correctly for a selection of users.
- Validate updates are being syndication into and from the target environment as expected.

Related tasks:

"Clearing version history" on page 1194
You use the clear versions tool to clear the version history of an item.

Starting and stopping servers, deployment managers, and node agents

Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

Procedure

1. Open a command prompt and change to the following directory:

   Note: In a clustered environment, use the `dmgr_profile_root` directory and not the *wp_profile_root* directory.

   Windows: *wp_profile_root*`\bin`

   Linux : *wp_profile_root*`/bin`

   IBM i: *wp_profile_root*`/bin`
2. Complete the following steps to start the servers, deployment managers, and node agents

Table 164. Steps to start the servers, deployment managers, and node agents.

Server	Steps
Deployment manager	Enter the following command: • Linux : `./startManager.sh` • Windows: `startManager.bat` • IBM i: `startManager`
Node agent	Enter the following command: • Linux : `./startNode.sh` • Windows: `startNode.bat` • IBM i: `startNode`
WebSphere Portal Express server	Enter the following command: **Note:** If you have a clustered environment or you renamed your WebSphere Portal Express server, *WebSphere_Portal* is the name that you defined for your WebSphere Portal Express server. • Linux : `./startServer.sh` *`WebSphere_Portal`* • Windows: `startServer.bat` *`WebSphere_Portal`* • IBM i: `startServer` *`WebSphere_Portal`*

3. Complete the following steps to stop the servers, deployment managers, and node agents:

Table 165. Steps to stop the servers, deployment managers, and node agents

Server	Steps
WebSphere Portal Express server	Enter the following command: **Note:** If you have a clustered environment or you renamed your WebSphere Portal Express server, *WebSphere_Portal* is the name that you define for your WebSphere Portal Express server. • Linux : `./stopServer.sh` *`WebSphere_Portal`* • Windows: `stopServer.bat` *`WebSphere_Portal`* • IBM i: `stopServer` *`WebSphere_Portal`*
Node agent	Enter the following command: • Linux : `./stopNode.sh` • Windows: `stopNode.bat` • IBM i: `stopNode`
Deployment manager	Enter the following command: • Linux : `./stopManager.sh` • Windows: `stopManager.bat` • IBM i: `stopManager`

4. In a clustered environment, you can use the deployment manager WebSphere Integrated Solutions Console to stop and start the application servers that are managed by the deployment manager:

Table 166. Stopping and starting the application servers that are managed by the deployment managerWebSphere Integrated Solutions Console.

Option	Steps
Start a specific application server in a cell	Complete the following steps to start a specific application server in a cell: 1. Open the deployment manager WebSphere Integrated Solutions Console. 2. Click **Servers** > **Application Servers**. 3. Select the server and click **Start**.
Start the entire cluster	Complete the following steps to start the entire cluster: 1. Open the deployment manager WebSphere Integrated Solutions Console. 2. Click **Servers** > **Clusters**. 3. Select the cluster and click **Start** or **Ripple Start**.
Stop a specific server in a cell	Complete the following steps to stop a specific application server in a cell: 1. Open the deployment manager WebSphere Integrated Solutions Console. 2. Click **Servers** > **Application Servers**. 3. Select the server and click **Stop** or **Immediate Stop**.
Stop the entire cluster	Complete the following steps to stop the entire cluster: 1. Open the deployment manager WebSphere Integrated Solutions Console. 2. Click **Servers** > **Clusters**. 3. Select the cluster and click **Stop** or **Immediate Stop**.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

Users and groups

IBM WebSphere Portal Express offers you centralized administration of users and user groups, allowing you to better define users and manage user access rights. Users can register and manage their own account information, or an administrator can provision and manage users. Group memberships can be used to give the required permissions to access an object or perform a request.

"Choose the type of group to use" on page 1220
IBM WebSphere Portal Express offers many types of groups and locations to store the groups. Choosing the type of group and the location the groups are stored is an important step in planning for using groups with WebSphere Portal Express.

"Deregistering users and groups" on page 1237
Depending on circumstances, you might want to remove users or groups from your WebSphere Portal Express that are no longer used or required. You can use the XML configuration interface (XML Access) to list such users and groups. You can also remove only some selected users and groups, and keep others for further use.

Related concepts:

"Working with the Portal Scripting Interface" on page 1120
Learn more about the different modes that you can use with the Portal Scripting Interface.

Choose the type of group to use

IBM WebSphere Portal Express offers many types of groups and locations to store the groups. Choosing the type of group and the location the groups are stored is an important step in planning for using groups with WebSphere Portal Express.

The following list provides a summary of available options for using groups in WebSphere Portal Express.

File Repository

Type of group: Static only

Advantage: Configured with your initial installation

Disadvantage: Not supported in production systems

LDAP Type of group: Static, Nested (optional), Dynamic (optional)

Advantage: LDAP is an established industry protocol, which can integrate with multiple applications. In most use cases, WebSphere Portal Express connects to and uses existing groups in an LDAP server.

Disadvantage: LDAPs are only required to support static groups. Individual LDAP vendors might optionally choose to support both nested and dynamic groups or to support either nested group or dynamic group. Check with your LDAP administrator to determine which type of groups your LDAP supports. If you have more requirements for using groups, consider to use an LDAP location and another location for the groups that are listed next.

Virtual Member Manager Federated database

Type of group: Static only

Advantage: Useful in situations where LDAP is read-only and needs more groups specific to Portal. Allows for cross-repository groups, that is, users from LDAP#1 and users from LDAP#2 may exist in the same database group.

Disadvantage: Groups are not accessible outside of the WebSphere Portal Express server.

Rule Based User Groups

Type of group: Dynamic only

Advantage: Provides dynamic group functions to WebSphere Portal Express in cases where an LDAP server does not support dynamic groups.

Disadvantage: Only supported by WebSphere Portal Express. Other WebSphere Application Server based products cannot use these groups. Groups are accessible outside of the WebSphere Portal Express server.

Related concepts:

"User registry considerations" on page 128
A user registry or repository authenticates a user and retrieves information about users and groups to do security-related functions, including authentication and authorization.

"Rule-based user groups"
Rule-based user groups for IBM WebSphere Portal Express allow you to define dynamic portal user groups.

Related information:

 Configuring dynamic member attributes in a federated repository configuration

Managing users and groups

The Manage Users and Groups portlet allows you to view, create, and delete users and user groups. You can also change group memberships.

Procedure

1. Log in to IBM WebSphere Portal Express as an administrator.
2. Click the **Administration menu** icon. Then, click **Access** > **Users and Groups**.

Rule-based user groups

Rule-based user groups for IBM WebSphere Portal Express allow you to define dynamic portal user groups.

Rule-based user groups are implemented as a custom repository adapter for Virtual Member Manager (VMM). Rule based user groups are represented by a unique group name, the Lightweight Directory Access Protocol (LDAP) search filter rule expression, and an optional description. The portal handles them as normal portal user groups. They are in a special base distinguished name in the user realm hierarchy. Administrators can create, define, update, or delete them by using the VMM API in WebSphere Application Server or the Portal User Management Architecture (PUMA) in WebSphere Portal Express like other groups. You can use these soft groups to assign security role mappings, portal access permissions, or visibility rules the same way as other portal user groups. The rule-based user groups feature handles the correct membership determination for the users during run time. Advantages:

- Rule-based user groups allow you to define and assemble dynamic portal user groups. They are based on LDAP search filter expressions applied to user attributes.
- These groups are persisted in the portal database, not in the main portal user repository. You do not need to enter them into the LDAP.

What you can do with rule-based user groups

- Define a rule-based user group including the syntax validation of the rule.
- Modify the rule or description of an existing rule-based user group including the syntax validation of the rule.
- Search for rule-based user groups based on the group name.
- Resolve the rule-based user group membership for particular users during run time.
- Display the members of a particular rule-based user group.

Note: This operation can have an impact on the performance of your portal. Perform it only to verify the resulting set of members after defining a rule base group.

- Delete an existing rule-based user group.

Notes:

1. Rule based user groups can contain only individual users, but not groups.
2. After defining a rule-based user group, you cannot change the unique group name.

"Configuring the rule-based user groups adapter"
To install rule-based user groups on your WebSphere Portal Express, you must set up a database, and configure VMM rule based groups.

Configuring the rule-based user groups adapter

To install rule-based user groups on your WebSphere Portal Express, you must set up a database, and configure VMM rule based groups.

"Database setup"
You must create the database table manually before you can use rule based user groups.

"Database source configuration" on page 1223
The rule-based user groups adapter uses a Java data source to communicate with the database that holds the table for the rule-based groups.

"Configuring the VMM rule-based groups repository" on page 1224
To enable the VMM rule-based groups repository adapter, modify several VMM configuration files.

Database setup:

You must create the database table manually before you can use rule based user groups.

The rule-based user groups feature stores the definitions of the rule-based user groups in a database table. This includes the name, rule, and description of the group. Use one of the following SQL statements to create the table, using a database and schema of your choice. Replace *schema_name* in the scripts with the schema name of your choice.

Syntax for DB2 and Derby databases:

```
CREATE TABLE schema_name.SOFTGROUPS
(ID INT NOT NULL GENERATED ALWAYS AS IDENTITY,
 GROUPNAME VARCHAR(128) NOT NULL,
 RULE VARCHAR(300) NOT NULL,
 DESCRIPTION VARCHAR(512),
 LASTMODIFIED TIMESTAMP,
 PRIMARY KEY (ID),
 UNIQUE (GROUPNAME));

CREATE INDEX schema_name.SOFTGROUPSIX1 ON
schema_name.SOFTGROUPS (LASTMODIFIED DESC);
```

Syntax for SQL databases:

```
CREATE TABLE schema_name.SOFTGROUPS
(ID INT NOT NULL IDENTITY PRIMARY KEY,
GROUPNAME VARCHAR(128) NOT NULL UNIQUE,
"RULE" VARCHAR(300) NOT NULL,
DESCRIPTION VARCHAR(512),
LASTMODIFIED DATETIME);
```

```
CREATE INDEX SOFTGROUPSIX1 ON
schema_name.SOFTGROUPS(LASTMODIFIED DESC);
sp_indexoption 'schema_name.SOFTGROUPS',
'disallowpagelocks', TRUE;
```

Syntax for Oracle databases:

```
CREATE TABLE schema_name.SOFTGROUPS
  (
     ID           INT,
     GROUPNAME    VARCHAR(128) NOT NULL,
     RULE         VARCHAR(300) NOT NULL,
     DESCRIPTION  VARCHAR(512),
     LASTMODIFIED TIMESTAMP,
     PRIMARY KEY  (ID),
     UNIQUE (GROUPNAME)
  );

CREATE INDEX schema_name.SOFTGROUPSIX1 ON
schema_name.SOFTGROUPS  (LASTMODIFIED DESC);

CREATE SEQUENCE softgroups_seq;

CREATE TRIGGER softgroups_seq_trigger
  before INSERT ON schema_name.SOFTGROUPS
  FOR each ROW
BEGIN
    IF ( :new.id IS NULL ) THEN
      SELECT softgroups_seq.nextval
      INTO   :new.id
      FROM   dual;
    END IF;
END;
/
```

Oracle does not support auto-increment or identity feature directly as part of the ID column definition. You must create a sequence and a trigger. For easy submission of the statement, make sure to add the final slash character (/). You can submit the statement by pressing the `Enter` key.

Database source configuration:

The rule-based user groups adapter uses a Java data source to communicate with the database that holds the table for the rule-based groups.

You must define a data source that references the required JDBC driver and points to the database that contains the groups table. For information about how to configure a data source, see the related links section.

Create the data source and, if you have a portal cluster environment, map it to the cluster scope.

Before you continue with the next configuration step, run the **Test connection** operation in the WebSphere Integrated Solutions Console and make sure that it runs successfully. Configure the rule-based user groups adapter later with the JNDI name of the data source. See the related links section for information.

Related information:

Configuring a data source using the administrative console

Configuring the VMM rule-based groups repository:

To enable the VMM rule-based groups repository adapter, modify several VMM configuration files.

About this task

In a cluster, you can change the configuration files directly on the deployment manager and then synchronize the changes to all cluster nodes.

"Configuring the VMM repository and realm"
Run the **wp-create-cur**, **wp-create-cur-custom-property**, and **wp-update-group-repository-relationship** commands to configure the VMM repository and realm.

"Configuring the rule attribute for the Group" on page 1225
In addition to the repository configuration, you must define the rule attribute as a new attribute for the entity type `Group`.

Related information:

"Database source configuration" on page 1223
The rule-based user groups adapter uses a Java data source to communicate with the database that holds the table for the rule-based groups.

Configuring the VMM repository and realm:

Run the **wp-create-cur**, **wp-create-cur-custom-property**, and **wp-update-group-repository-relationship** commands to configure the VMM repository and realm.

Procedure

1. Open a command prompt and change to the *wp_profile_root*`/ConfigEngine` directory.
2. Run the following task to add the repository configuration to VMM. See the related links section for information about this task.
   - Linux : `./ConfigEngine.sh wp-create-cur -DWasPassword=yourpassword -Dfederated.cur.id=SoftGroups -Dfederated.cur.adapterClassName=com.ibm.wps.vmm.adapter.softgroups.SoftgroupsAdapter -Dfederated.cur.baseDN=o=softgroups`
   - IBM i: `ConfigEngine.sh wp-create-cur -DWasPassword=yourpassword -Dfederated.cur.id=SoftGroups -Dfederated.cur.adapterClassName=com.ibm.wps.vmm.adapter.softgroups.SoftgroupsAdapter -Dfederated.cur.baseDN=o=softgroups`
   - Windows: `ConfigEngine.bat wp-create-cur -DWasPassword=yourpassword -Dfederated.cur.id=SoftGroups -Dfederated.cur.adapterClassName=com.ibm.wps.vmm.adapter.softgroups.SoftgroupsAdapter -Dfederated.cur.baseDN=o=softgroups`
3. Run the following command to update the repository configuration with custom properties: Specify the following attributes on the command line:

   dataSource
   The attribute must point to the correct JNDI name of the previously configured data source for the rule-based groups database.

   dbSchema
   The attribute must declare the database schema that holds the rule-based groups table.

   dbType If your database server type is SQLServer, you must declare the

attribute dbType by specifying SQLServer as the value. For all other database server types, you can omit the value.

Base entry specification
You can set the base entry specification that defines the base distinguished name and suffix for rule based groups to a different value.

name The name and the nameInRepository must be the same.

- Linux :

```
./ConfigEngine.sh wp-create-cur-custom-property -DWasPassword=yourpassword -Dcur.id=SoftGrou
./ConfigEngine.sh wp-create-cur-custom-property -DWasPassword=yourpassword -Dcur.id=SoftGrou
./ConfigEngine.sh wp-create-cur-custom-property -DWasPassword=yourpassword -Dcur.id=SoftGrou
```

- IBM i:

```
ConfigEngine.sh wp-create-cur-custom-property -DWasPassword=yourpassword -Dcur.id=SoftGroups
ConfigEngine.sh wp-create-cur-custom-property -DWasPassword=yourpassword -Dcur.id=SoftGroups
ConfigEngine.sh wp-create-cur-custom-property -DWasPassword=yourpassword -Dcur.id=SoftGroups
```

- Windows:

```
ConfigEngine.bat wp-create-cur-custom-property -DWasPassword=yourpassword -Dcur.id=SoftGroup
ConfigEngine.bat wp-create-cur-custom-property -DWasPassword=yourpassword -Dcur.id=SoftGroup
ConfigEngine.bat wp-create-cur-custom-property -DWasPassword=yourpassword -Dcur.id=SoftGroup
```

4. You must enable the cross repository group lookup for the repositories you want to use. To find Groups Entities in the SoftGroups Repository, run the following task:
   - Linux : `./ConfigEngine.sh wp-update-group-repository-relationship -DWasPassword=password -Drepository.id=ldapid -Drepository.forgroups=SoftGroups`
   - IBM i: `ConfigEngine.sh wp-update-group-repository-relationship -DWasPassword=password -Drepository.id=ldapid -Drepository.forgroups=SoftGroups`
   - Windows: `ConfigEngine.bat wp-update-group-repository-relationship -DWasPassword=password -Drepository.id=ldapid -Drepository.forgroups=SoftGroups`

Related information:

"Database source configuration" on page 1223
The rule-based user groups adapter uses a Java data source to communicate with the database that holds the table for the rule-based groups.

Configuring a federated custom user registry

Configuring the rule attribute for the Group:

In addition to the repository configuration, you must define the rule attribute as a new attribute for the entity type Group.

Procedure

1. Edit the file wimxmlextension.xml in the directory *PortalServer_root*/config/*cell_name*/wim/model/. If the file does not exist yet, create it.
2. Add the following attribute definitions:

```
<?xml version="1.0" encoding="UTF-8"?>
<sdo:datagraph xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xmlns:sdo="commonj.sdo"
               xmlns:wim="http://www.ibm.com/websphere/wim">
  <wim:schema>
    ...
      <wim:propertySchema nsURI="http://www.ibm.com/websphere/wim" dataType="String"
```

```
            multiValued="false"
            propertyName="rule">
           <wim:applicableEntityTypeNames>Group</wim:applicableEntityTypeNames>
      </wim:propertySchema>
     ...
   </wim:schema>
</sdo:datagraph>
```

See the related links section for information about adding attributes.

3. Restart the portal server for the VMM configuration changes to become effective. In a portal cluster environment, synchronize the changes and restart the complete cluster, including deployment manager and node agents.

Related tasks:

"Adding attributes" on page 569
The VMM is configured with a default attribute schema that might not be compatible with your LDAP server. Add attributes to extend the VMM attribute schema and then map them between IBM WebSphere Portal Express and your user registry.

Creating new users and groups

Use this topic to create new users and groups by using the portlet and adding them to an existing group.

About this task

The objective of this task is to step you through the process of creating new users and groups.

To complete this task, you must have at least the Editor@USERS role to create users and the Editor@USER_GROUPS role to create user groups. USERS and USER_GROUPS are virtual resources.

The portlets that are used in this task are Users and Groups

Note: As you follow the instructions for this task, refer to the *user IDs and passwords* topic, for more information on the characters you can use in the user ID, password, first name, and last name fields.

Procedure

1. Log in to your portal as an administrator.
2. Click the **Administration menu** icon. Then, click **Access** > **Users and Groups**.
3. Select the user group for the user.
4. Click **New User** or **New Group**.

   Note: If the Editor role is not assigned to the following virtual resources: USER, USER GROUPS, and USER SELF ENROLLMENT, **New User** and **New Group** might not display in virtual portals .
5. If you are creating a new user group, enter a name for the user group.
6. If you are creating a new user, do the following tasks:
   a. Enter a user ID for the new user. The user ID must be 3 - 60 characters in length.
   b. Enter and confirm a password for the new user. The password must be unique and 5 - 60 characters in length.
   c. Enter a first name for the new user.

d. Enter a last name for the new user.
e. Optional: Enter an email address for the new user. This field is not needed for successful creation of a new user.

Note: If you use an LDAP server for your users and groups, your LDAP configuration might place additional restrictions on user and group names. For example, the LDAP configuration might require user names and passwords to be a minimum of 8 characters in length. For information about supported characters, see the related links.

7. Select **Preferred language** from the drop-down list. This field is not needed for successful creation of a new user. If you do not select a preferred language or if the language is not supported by the portal, the default language is the default WebSphere Portal Express language.
8. Click **OK**.

Related information:

"User IDs and passwords" on page 107
Understanding character limitations for user IDs and passwords is important because they are used throughout the system to provide access and secure content. The character limitations provided here apply to the IBM WebSphere Portal Express administrator, IBM WebSphere Application Server administrator, database administrator, LDAP server administrator, and user IDs. Database and LDAP servers can have more restrictive limitations than provided here. Therefore, check the database and LDAP server product documentation for restrictions. Failure to correctly define user IDs and passwords during the installation process can result in installation failure. In addition, your company might have more restrictive user ID and password requirements that you must also follow.

Viewing the members of a group

Display a list of all the group members for a particular user group.

About this task

Perform these steps to view the members of a user group:

Procedure

1. Search for the user group whose members you want to view.
2. Click the required user group in the search results. All members of the group are listed.

Editing user information

Edit user information such as password, User ID, first name, last name, email address, or preferred language.

Procedure

1. Search for the required user or click the **All Authenticated Portal Users** link to get a list of users.
2. Click the **Edit** icon for the required user.
3. Make the necessary changes to the user information.
4. Click **OK** to save your changes, or **Cancel** to exit without saving your changes.

Reusing group information

During the authentication of a user, IBM WebSphere Application Server stores information about which groups users belong to. This information is static for the authentication session of the user. In addition, it can be provided by an External Security Manager via a Trust Association Interceptor. In this case, IBM WebSphere Application Server does not load the information on its own. IBM WebSphere Portal Express can participate in this flow and reuse the information from the WebSphere Application Server security context instead of retrieving it from the LDAP server. This function is also referred to as group assertion or WebSphere Application Server group assertion.

About this task

To prevent modifying existing behavior of your environment or losing existing group information, WebSphere Portal Express does not reuse group information by default. For this reason, you must configure WebSphere Portal Express to reuse group information from the WebSphere Application Server security context. You can choose to reuse group information for user management or for access control.

- Reusing group information for user management results in all components of WebSphere Portal Express benefit from the faster group membership lookup. During the authentication session, the membership of the current user is based on the information that is provided by WebSphere Application Server. This reuse of information reduces load on your LDAP server, increases authentication performance, and results in the ability to define group membership at the authentication layer.
- Reusing group information for access control enables the system to react on possible per request changes of the WebSphere Security context. By default the Security context is not modifiable during an authentication session. However,WebSphere Application Server provides plug points, which allow the execution of a Trust Association Interceptor on every request, which could be used to establish a new security Subject on every request. In this case Portal Access Control would be able to work with the updated subject information and would build a dynamic environment. However, this option requires more system resources, custom extensions to WebSphere Application Server security and impacts performance.

Note: The recommended option is for user management, as this case provides the performance and functional enhancements. The second option for access control is used in specific scenarios, typically as directed by IBM Support or IBM technical documentation.

Note: Do not combine both options as this leads to high CPU load on your system.

Complete the following steps to reuse group information:

Procedure

1. Log on to the WebSphere Integrated Solutions Console (or deployment manager WebSphere Integrated Solutions Console in a cluster).
2. Go to **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Choose the appropriate options to reuse group information:
   - To reuse group information for user management, the first option that is used typically as an enhancement, complete the following steps:
     a. Select the **WP_PumaStoreService** resource environment provider.

b. Select **Custom properties**.
c. Click **New**.
d. Enter `store.puma_default.filter.assertionFilter.classname` in the **Name** field.
e. Enter `com.ibm.wps.um.AssertionFilter` in the **Value** field.
f. Click **Apply**.
g. Click **Save** to save the changes to the master configuration.

- To reuse group information for access control, complete the following steps:
  a. Select the **WP PACGroupManagementService** resource environment provider.
  b. Select **Custom properties**.
  c. Click **New**.
  d. Enter `accessControlGroupManagement.useWSSubject` in the **Name** field.
  e. Enter `true` in the **Value** field.
  f. Click **Apply**.
  g. Click **Save** to save the changes to the master configuration.

4. Log out of the WebSphere Integrated Solutions Console.
5. Restart the WebSphere_Portal server.

Related tasks:

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

Deleting users and groups

You can delete a user or user group from WebSphere Portal Express.

Procedure

1. From the **Users and Groups** page, choose the link for deleting either a user or a user group:
   - To delete a user, click **All Authenticated Portal Users**.
   - To delete a user group, click **All Portal User Groups**.
2. Select the user or user group that you want to delete.
3. Click the **Delete** icon. A dialog box warns that the selected user or group information and all private data will be removed from the database.
4. Click **OK** to save your changes, or **Cancel** to exit without saving your changes.

   Note: Deleting a user group from the portal does not delete the members of the group.

Virtual Users and Groups

There are two predefined virtual user groups (All Authenticated Portal Users and All Portal User Groups) and one virtual user (Anonymous Portal User). These predefined virtual groups and user allow for access control configuration that applies to abstract sets of users. They are not stored in the user registry. They only exist within the access control context. You cannot change group membership or other attributes of these virtual usergroups and user.

This section describes the following virtual user and user groups:

- Anonymous Portal User
- All Authenticated Portal Users
- All Portal User Groups

Anonymous Portal User

This virtual user models a user who has not yet logged in. Assigning roles to this user on a resource allows access to this resource prior to authentication. This is useful for creating public welcome pages. The **Anonymous Portal User** is not considered to be a member of any group. On pages and their virtual resource parents **CONTENT_NODES** and **PORTAL**, you can only assign the **Anonymous Portal User** to the **User** role type. Delegated Administration on the **Anonymous Portal User** are derived from privileges on the virtual resource **Users**. In other words, the delegating user must have at least Delegator role on the virtual resource **Users** before being allowed to delegate role assignments to **Anonymous Portal Users**.

All Authenticated Portal Users

This virtual user group models the set of all known users. After successfully logging in, users lose the **Anonymous Portal User** identity and become authenticated members of the **All Authenticated Portal Users** virtual user group. Roles assigned to this user group allow establishment of permissions that will apply to all authenticated users and thus support setting up the default privileges for authenticated portal access.

All Portal User Groups

This virtual user group contains all non-virtual user groups.

Administering user impersonation

Use the impersonation feature to access another user's system as though you are that user. Use caution with this configuration. The user who is doing the impersonation might get more permissions than initially granted to them.

Client-side aggregation does not support user impersonation. Do not activate client-side aggregation on any portal pages where the impersonation portlet is deployed.

Use the impersonation feature to view new pages and portlets. Users such as support specialists can use the impersonation feature to find issues and errors. For example, a portal administrator encounters a problem that cannot be resolve. A support specialist can use the impersonation feature to access the system to determine a solution to the problem.

You can use the default Impersonation portlet to impersonate specific users. Alternatively, you can create a resource environment provider to enable impersonation and develop a custom portlet for impersonating users. Use Portal Access Control to assign impersonation roles to users. Assign the **Can Run As User** role to the user you plan to impersonate after you enable the impersonation feature.

User impersonation and people awareness: When a user who is enabled for impersonation impersonates other users, the people awareness feature is disabled for the entire session for which that user is authenticated.

Restriction: A user cannot impersonate himself or herself.

"Enabling and disabling impersonation"
By default, user impersonation is enabled, but you can manually disable or enable the impersonation feature as needed.

"Developing a custom portlet" on page 1232
You can use the default Impersonation portlet to impersonate specific users. Alternatively, you can create a resource environment provider to enable impersonation and develop a custom portlet for impersonating users.

"Assigning the **Can Run As User** role" on page 1233
Users with administrator access in WebSphere Portal Express can assign the **Can Run As User** role to designated users. Use Portal Access Control to assign the role. You can use the Virtual Resources option to grant permission for all users or groups in the system.

"Impersonating an unauthenticated user" on page 1234
Users with administrator access in WebSphere Portal Express can impersonate an unauthenticated user.

Related concepts:

"Controlling access" on page 1524
After creating users and groups, you can assign them different levels of access to specific resources, roles, and policies. This access controls what actions they can perform on various pages, portlets, and applications.

"Auditing" on page 1731
IBM WebSphere Portal Express includes an auditing function that allows users to log certain events and their originators in to a separate log file. This file can then be used to track administrative activities.

"User and group management" on page 2902
The Portal User Management Architecture (PUMA) System programming interface (SPI) provides interfaces for accessing the profiles of a portal User or Group.

Related tasks:

"Installing a portlet" on page 1240
Installing a portlet makes it available to portal users. Adding a portlet to a page makes the portlet accessible to users with the appropriate rights.

Related information:

"Users and groups" on page 1218
IBM WebSphere Portal Express offers you centralized administration of users and user groups, allowing you to better define users and manage user access rights. Users can register and manage their own account information, or an administrator can provision and manage users. Group memberships can be used to give the required permissions to access an object or perform a request.

Enabling and disabling impersonation

By default, user impersonation is enabled, but you can manually disable or enable the impersonation feature as needed.

Procedure

1. Complete the following steps to disable or enable the impersonation feature:
   - To disable the impersonation feature, run the `disable-impersonation` task from the *wp_profile_root*`/ConfigEngine` directory.

Windows

```
ConfigEngine.bat disable-impersonation -DWasPassword=password
-DPortalAdminPwd=password -DCategoriesList=wp.auth.base
```

Linux

```
./ConfigEngine.sh disable-impersonation -DWasPassword=password
-DPortalAdminPwd=password -DCategoriesList=wp.auth.base
```

IBM i

```
ConfigEngine.sh disable-impersonation -DWasPassword=password
-DPortalAdminPwd=password -DCategoriesList=wp.auth.base
```

- To enable the impersonation feature, run the `enable-impersonation` task from the *wp_profile_root*`/ConfigEngine` directory.

  Windows

  ```
  ConfigEngine.bat enable-impersonation -DWasPassword=password
  -DPortalAdminPwd=password -DCategoriesList=wp.auth.base
  ```

  Linux

  ```
  ./ConfigEngine.sh enable-impersonation -DWasPassword=password
  -DPortalAdminPwd=password -DCategoriesList=wp.auth.base
  ```

  IBM i

  ```
  ConfigEngine.sh enable-impersonation -DWasPassword=password
  -DPortalAdminPwd=password -DCategoriesList=wp.auth.base
  ```

2. Stop and restart the *WebSphere_Portal* server.

Related tasks:

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

Developing a custom portlet

You can use the default Impersonation portlet to impersonate specific users. Alternatively, you can create a resource environment provider to enable impersonation and develop a custom portlet for impersonating users.

Sample code

If you do not want to use the WebSphere Portal Express **Impersonation** portlet, use the following sample to develop a portlet to impersonate users:

```
import com.ibm.portal.portlet.service.impersonation.ImpersonationService;
import com.ibm.portal.portlet.service.PortletServiceHome;

public class MyImpersonationPortlet extends GenericPortlet
{
    private PortletServiceHome psh;

    @Override
    public void init() throws PortletException
    {
        try
        {
            javax.naming.Context ctx = new javax.naming.InitialContext();
            psh = (PortletServiceHome) ctx.lookup(ImpersonationService.
JNDI_NAME);
        } catch (Exception ex)
        {
            // error handling
        }
    }

    @Override
    public void processAction(ActionRequest request, ActionResponse
response) throws PortletException, IOException
    {
        // obtain the service object and use the service
        ImpersonationService impersonationService = (ImpersonationService)
```

```
psh.getPortletService(ImpersonationService.class);
        try
        {
            impersonationService.doImpersonate(request, response,
stringuserDN);
        } catch (Exception e)
        {
            // error handling
        }
    }
}
```

You can enter the information for the user you want to impersonate in the `stringuserDN`. Alternatively, you can use the PUMA SPI User object.

Note: The impersonation feature becomes active with the next request.

Assigning the Can Run As User role

Users with administrator access in WebSphere Portal Express can assign the **Can Run As User** role to designated users. Use Portal Access Control to assign the role. You can use the Virtual Resources option to grant permission for all users or groups in the system.

About this task

Important: If you use virtual resources to assign the **Can Run As User** role, the user can also impersonate the administrator.

Procedure

1. Log on to WebSphere Portal Express as an administrator.
2. Click the **Administration menu** icon in the toolbar.
3. Choose one of the following options to assign **Can Run As User** access:
   - Complete the following steps to assign **Can Run As User** access to a user:
     a. Click **Access** > **User and Group Permissions**.
     b. Click **Users**.
     c. Search for the user that you want to assign the **Can Run As User** role.
     d. Click the **Select Resource Type** icon for the appropriate user.
     e. Go to the page that contains the **Virtual Resources** option. Use the **Page Next** button and click the link.
     f. Go to the page that contains the **USERS** option and click the **Assign Access** icon.
     g. Select the **Explicitly Assign** check box for the **Can Run As User** role.
     h. Click **OK**.
   - Complete the following steps to assign **Can Run As User** access for a user group:
     a. Click **Access** > **Resource Permissions**.
     b. Click **User Groups**.
     c. Search for the user group that you want to assign the **Can Run As User** role.
     d. Click **Assign Access**.
     e. Select the **Explicitly Assign** check box for the **Can Run As User** role.
     f. Add the user that you want to impersonate the users of this user group.
     g. Click in the breadcrumb to return to the user group page.
     h. Click **Apply** and accept the changes.

i. Click **OK**.

4. Verify that the user now has **User** and **Can Run As User** access.

Impersonating an unauthenticated user

Users with administrator access in WebSphere Portal Express can impersonate an unauthenticated user.

Procedure

1. Log on to WebSphere Portal Express as the administrator.
2. Click the **Administration menu** icon. Then, click **Access** > **Resource Permissions**.
3. Click **Virtual Resources**.
4. Locate the **WCM REST SERVICES** resource type and then click the **Assign Access** icon.
5. Locate the **User** resource permission and then click the **Edit Role** icon.
6. Click **Add**.
7. Check the **Anonymous Portal User** check box.
8. Click **OK**.
9. Verify that you can now impersonate an unauthenticated user.

Results

The users with the **`Can Run As User`** role can now impersonate another user.

Customize common name generation

By default, IBM WebSphere Portal Express generates common names that consists of the user's first name followed by the last name. You can change the order that common names are generated.

To change the order of common names modify the following three configuration properties, located in the Puma Store Service. See the related links section for information:

- **`puma.commonname`**: This property defines how the common name is generated. You can define dynamic and static parts. Dynamic parts are added using *X*, where *X* stands as a reference number. Dynamic parts can only be available and valid user attributes. By default {0} {1} is used.
- **`puma.commonname.parts`**: This property defines the amount of dynamic parts in the common name.
- **`puma.commonname.X`**: This property defines the user attribute for dynamic part *X*. *X* must be between 0 and **`puma.commonname.parts-1`**.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Nested groups

Two groups are nested if one of the groups contains the other group as a member. The access control system treats this as though all members of the contained group are also members of the containing group. In other words, permissions for nested groups are treated as cumulative.

One group, GlobalMarketing, could for example contain another group, USMarketing, resulting in all members of USMarketing being treated as members of GlobalMarketing. This means that members of USMarketing inherit the access rights granted to GlobalMarketing members. So, if GlobalMarketing has view access to the **File Server** portlet, and USMarketing has view access to the **Reminder** portlet, USMarketing has view access to both the **File Server** and **Reminder** portlets. For example, Joe, as a member of the GlobalMarketing group, can only access the **File Server** portlet, but Susan, as a member of the USMarketing group, can access both portlets.

Note: If you do not plan to use nested groups for access control inheritance, set **accessControlDataManagement.enableNestedGroups** to `false` in the **Access Control Data Management** service, **nestedGroupLookup.disabled** to true in the **WCM WCMConfigService** service, and **rulesEngine.user.nestedGroupLookup** to false in **WCM PersonalizationService** to improve performance. This will limit the membership lookup that Portal Access Control performs to one group level in the hierarchy. This means that a user is granted access rights only by explicit role mappings or role mappings to the groups of which that user is a direct member. See Setting service configuration properties for information.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Registration/Edit My Profile and Login portlets

The Registration/Edit My Profile and Login portlet resides on special pages where the anonymous user has access rights based on the User role. The unique name **wps.Login** is assigned to the page holding the login portlet, and the unique name **wps.Selfcare** is assigned to the page holding the Registration/Edit My Profile portlet.

The Registration/Edit My Profile portlet is for subscriber management. Registration allows users to register for access and information entered during Registration can be updated in Edit My Profile. Registration is also referred to as "Sign up".

This topic is divided into the following subtopics:

Registration/Edit My Profile

During registration, the user enters mandatory data, such as the user ID and first and last names. The user has the option to select the preferred language from a list

of available languages. The portal uses this language in all interactions and makes this information available to all portlets so that they can adapt to the user preference. If a language is not selected, the portal determines which language to use from the users' browser settings.

Table 167. Registration/Edit My Profile portlet configuration parameters

Parameter Name	Registration/Edit My Profile portlet configuration parameters
HomePageUniqueName	Use this property to identify the page that is selected when users click **Cancel**. Update this parameter to match your environment. Default: `ibm.portal.Home`
GenerateCN	Use this property to specify whether WebSphere Portal Express should generate the value of the `cn` attribute from the supplied user attributes. Default: `true`
attr_XXX	This portlet applies special handling to a subset of user attributes. You can use the **attr_XXX** parameters to match the actual attribute name to the logical name used in the portlet. Attributes that receive special handling: **alternativeCalendar** **cn** **email** **firstDayOfWeek** **firstName** **firstWorkDayOfWeek** **lastName** **password** **preferredCalendar** **preferredLanguage** **timeZone,uid**

Creating new attributes for the Registration/Edit My Profile portlet

You can change information entered at registration. Administrators can determine what information appears in the profile. LDAP or member management fields determine what fields appear as potential fields for the user profile. Some fields are disabled because they are required fields in PUMA (user management), and cannot be removed. Optional attributes of data type **string** can be added using the configuration mode of the portlet. If you do not see the required attribute displayed in the configuration mode, review the information under Adapting the attribute configuration to ensure that the attribute was properly added, mapped, or both.

By default, the same portlet instance is used for Registration and for Edit My Profile. In this design, changes made with the configure link apply to both cases.

Since a separate instance of the Registration/Edit My Profile portlet exists for each virtual portal, it is possible to have different customized portlets for each virtual portal.

Login

To access the Login portlet, click **Log In** in the banner. The Login portlet can also be placed on any page.

Since a separate instance of the Login portlet exists for each virtual portal, it is possible to have different customized portlets for each virtual portal.

Related concepts:

"Authentication" on page 1520
Authentication requires users to identify themselves to gain access to a system or resources. The combination of a user ID and a password is the most common method of authentication. Users can identify themselves immediately upon entry to the system or the system can prompt users to identify themselves before accessing protected resources. After users successfully authenticate, the system identifies which resources-specific users have sufficient authorization to access.

Related tasks:

"Adding more attributes to VMM" on page 568
After you install IBM WebSphere Portal Express and configuring your LDAP user registries, you must adapt the attribute configuration to match the configured LDAP servers and your business needs. However, do not complete these steps if you configured only a database user registry or the default federated file-based repository for out-of-box installations.

Related information:

"Enabling step-up authentication and/or the Remember me cookie" on page 1587
Using step-up authentication and/or the Remember me cookie lets you fine-tune user authentication to pages and portlets.

Deregistering users and groups

Depending on circumstances, you might want to remove users or groups from your WebSphere Portal Express that are no longer used or required. You can use the XML configuration interface (XML Access) to list such users and groups. You can also remove only some selected users and groups, and keep others for further use.

About this task

IBM WebSphere Portal Express stores users and groups that exist in the user registry as entries in the database. When you use the XML configuration interface or the **Manage User and Groups** portlet to delete users and groups, they are deleted from both the user registry and from the database. Deleting a user or group directly from the configured user registry does not remove the database entry. Also, WebSphere Portal Express does not remove entries from its database when users or groups are muted in the user registry, for example, users with too many wrong password attempts. You can manually remove the users and groups from the database.

Examples for removing users or groups can be the following cases:

- Portal users or groups were removed from the user registry, but not from the portal database.

- User IDs were deactivated, for example after too many wrong password attempts.

Note: After you delete these entries by using the modified XML script, all customization is lost for the deleted users and groups.
To remove users and groups from your portal, proceed as follows:

Procedure

1. Make a backup copy of your portal database.
2. To identify and list these users and groups, run an XML export and use the `cleanup-users` attribute.

   Specify the `cleanup-users` attribute with the `request` tag of type `export`, and set its value to `true`. You also need to set the `export-users` attribute to `true`.

   The resulting output file lists the affected users and groups with their action set to `delete`.

   The XML sample file `CleanupUsers.xml` shows an example of how you can export such users and groups. For information about the sample XML configuration files and their location, read *Sample XML configuration files*.

   Note: CF03 If the number of invalid users is very high, the XML export step can fail with an out-of-memory exception. For such cases, APAR PI23109 introduces a new XML element `threshold`. In case of such out-of-memory exceptions, add `threshold="10000"` to the `<request ... >` element in the `CleanupUsers.xml` script. This option limits the number of exported users to 10,000. When you use this approach, repeat the export step and all following steps until the exported file contains no entries any more. You need to have APAR PI23109 or fix pack CF03 installed to use this XML element.
3. Check the output file from the previous step and **remove** all users and groups that you want to **keep** in the portal database. For example, you might want to keep the muted users and re-enable their passwords. All users and groups that remain in the file are removed from the database in the following import step.
4. Import the modified XML file into your portal. The portal removes all users and groups that you retained in the XML file during the previous step from the portal database.

Results

After you delete these entries by using the modified XML script, all customization is lost for the deleted users and groups.

Related reference:

"Sample XML configuration files" on page 1111
Sample files are provided for your reference to help illustrate how to use XML configuration for different portal configuration purposes. Before you use them, read the other topics about the XML configuration interface carefully. Many of the samples need to be modified with valid page or user name before they can be used.

Managing portlets, portlet applications, and iWidgets

You must do some preparatory tasks before you make your portlets, portlet applications, or iWidgets available to your users by putting them on portal pages. This preparatory task includes installing, deploying, and configuring portlets, applications, and iWidgets.

You can further work with portlets by using the administration portlets that are available under the portal **Administration menu** > **Portlet Management**. Refer to the portlet helps for details. After you complete preparing your portlets or iWidgets, you can make them available for your users.

start of the page that is displayed, even if the link that was clicked points to a portlet, which is placed further down on the page. To achieve this, you can disable the anchors.

"Managing iWidgets in your portal" on page 1247
You can add iWidgets to your IBM WebSphere Portal Express.

Web modules, portlet applications, and portlets

A web module represents a web application. It is used to assemble servlets and JSP files as well as static content, such as HTML pages, into a single deployable unit.

Web modules are stored in web archive (WAR) files, which are standard Java archive files. The standard file extension for WAR files is `.war`. A Web module can contain one or more portlet applications, servlets, JavaServer Pages (JSP) files, and other files. A deployment descriptor, stored in an Extensible Markup Language (XML) file, declares the contents of the modules, information about the structure and external dependencies, and a description of how the components are to be used at run time.

Portlet applications are created implicitly when a WAR file is deployed. The portlet application holds one or more related portlets that come packaged in the same installation file. These portlets can share resources and send messages among themselves to communicate events. A portlet application may consist of a single portlet or multiple portlets. An example of a single portlet application is the Reminder portlet application, which contains a single portlet named Reminder Portlet. An example of a portlet application with multiple portlets is Portlet Manager Application, which contains Manage Web Modules, Manage Applications, and Manage Portlets.

You can add portlets to a running system at any time. After installation, the new portlets are immediately available to administrative users. They can assign the appropriate user roles to the appropriate groups and users so that they can access and use the portlets. Once available, the portlets can be selected for display on the portal pages of users and can be edited as appropriate. Identification information of WAR files is stored in a database for easy deployment in complex server environments with multiple portal servers. By allowing all files associated with a portlet to be packed into a single file, distribution and deployment of new portlets is made easier. Portlets can be distributed in WAR file format through websites and other means.

Installing a portlet

Installing a portlet makes it available to portal users. Adding a portlet to a page makes the portlet accessible to users with the appropriate rights.

Before you begin

Before you install a portlet, you must be aware of the following requirements:

- An administrator must have the Manager role on the portal to install portlets. Therefore, you must log in with a user ID with the Manager role access rights.
- Windows limits the maximum path length to 260 characters, the name of the WAR file must be 25 characters or less. Installing a WAR file with a name that is more than 25 characters results in an error.
- You cannot install a portlet more than once. If you want two instances of a portlet application or portlet, use the copy command to create a second instance.

About this task

Each WAR file includes descriptive information about the portlet, which is placed in a database that can be queried by other portal components. During installation, Application Server unpacks the WAR file and places the portlet classes and resources in a file system.

During the installation, the portlet state is set to active, and a new rule is automatically added to Access Control that defines the user who installed the portlet as the owner, granting management access for that portlet. The user must assign the appropriate user roles to the appropriate users and groups so that they can access and use that portlet.

Procedure

1. Click the **Administration menu** icon. Then, click **Portlet Management** > **Web Modules**.
2. Click **Install**.
3. Enter the location of the WAR file or click **Browse** to find the location of the file to install.
4. Click **Next**.
5. Verify WAR file information and click **Finish** to install the WAR file.

Results

By default, portlet applications and portlets are set to an active state after installation. After you install a portlet, you can add the portlet to a page with the Portlet Palette. Add the installed portlet to a category in the Portlet Palette and then drag the portlet to the page. If you do not plan to frequently add this portlet to other pages, you can still add the portlet to a page without adding the portlet to a category. To add a portlet to the page without adding it a category, search for the installed portlet in the Portlet Palette and drag the portlet to the page.

What to do next

After the installation is complete, a message appears at the start of the page that indicates a successful installation. If there are any problems during the process, an error message appears in the Manage Web Modules page. Click the **View Details** link to examine the error log.

Deploying Java Platform, Enterprise Edition resources

You can manually pre-deploy portlet application WAR files by using the WebSphere Integrated Solutions Console and later register them into WebSphere Portal Express, together with other Java Platform, Enterprise Edition resources and artifacts.

About this task

Portlet applications are usually packaged into single WAR files. These WAR files can be directly deployed into the portal by using portal administration means. For example, the Manage Web Modules portlet or the XML configuration interface. The portal administration functions manage the correct deployment and configuration into both WebSphere Application Server and WebSphere Portal Express.

You might want to deploy portlet applications together with EJBs, or bundle several WAR files into the same EAR file, or work with similar scenarios. For this type of requirement WebSphere Portal Express provides the predeployed mode with the XML configuration interface. It allows you to configure portlet applications into the portal that you predeployed into the application server as part of a larger EAR file.

You deploy the EAR file into the application server by using the application server administration interfaces, such as the WebSphere Integrated Solutions Console or the `wsadmin` command-line tool. When you deploy the EAR file into the application server, you can use the portal XML configuration interface to run the portal specific configuration steps that are required to configure the available portlet applications.

Procedure

1. Bundle the portlet applications that you want to register together with other Java Platform, Enterprise Edition resources as an Enterprise Application Archive (EAR file). For details about how to do this procedure refer to the correct version of the WebSphere Application Server product documentation: http://www.ibm.com/software/webservers/appserv/was/library/.
2. Deploy this EAR file into WebSphere Application Server. To do this step, use the `wsadmin` command-line tool or the WebSphere Integrated Solutions Console. Consult the WebSphere Application Server product documentation for details about how to do this step. Take a note of the target directory to which you deploy the EAR file; this step is needed for creating the XML file in the next step. Also, make sure that the web modules in the EAR file are deployed with the same server and virtual host mappings as the portal.
3. Create a portal XML script file that deploys and configures the portlet applications that are deployed with the EAR file. For a sample XML file for deploying and configuring a predeployed portlet, refer to RegisterPreDeployedEAR.xml.
4. Run the XML script by using the portal XML configuration interface.

Results

After you complete these steps, the portlet is ready for use.

Note:

1. You can register predeployed applications into the portal only by using the XML configuration interface. When you register a predeployed application, the Manage Web Modules portlet shows this application.
2. You can later use the portal administration portlets to remove the portlet definitions from the portal database. However, this process does not remove the EAR file from WebSphere Application Server.
3. You can update a predeployed portlet application only by using the XML configuration interface.
4. You cannot update a deployed portlet application WAR file with a predeployed EAR file and vice versa. If you want to change between the two types of files, you must delete the existing portlet application and deploy the new one. However, deleting the existing portlet application also deletes all configuration data of that application. A predeployed application can be updated only by updating the EAR file in WebSphere Application Server and subsequent update

of the contained WAR file in portal by using the XML configuration interface. Cross updates of a predeployed EAR file with a real WAR and vice versa are not possible.

5. The WebSphere Application Server administrator must take care of correctly configuring the dispatch mechanisms for these applications.

For more information about the XML configuration interface and how to use it, refer to XML configuration interface. The sample XML file RegisterPreDeployedEAR.xml shows you an example for deploying and configuring a predeployed portlet.

The following table shows the differences between WAR and EAR deployment:

Table 168. Differences between WAR files and predeployed EAR files and the affected portal area

Affected portal area	WAR file	Predeployed EAR file
Portlet application	The portlet application is provided as WAR file. The portal configuration is read directly from the file stream. The WAR file is deployed into the application server by the portal.	The portlet application is already extracted and deployed into the application server as part of the EAR file. The portal server reads the available portal configuration information (`portlet.xml`, and so on) from the location where the contained WAR file was extracted to.
Context Root	The context root is assigned by the portal during WAR deployment.	The context root is assigned by the EAR developer and stored in the file `application.xml`. You must ensure that the context root that you specify when you register the portlet matches the one specified in the EAR `application.xml`. (refer to the sample XML file RegisterPreDeployedEAR.xml.
Display name	The display name is assigned by Portal during WAR deployment.	The display name is assigned by the EAR developer and stored in the file `application.xml`.
WebSphere Application Server policy for portlet applications	The policy is stored in the WAR file and promoted to the EAR file by the portal during WAR deployment.	The policy is stored in the EAR file.
Portlet administration	You administer WAR files by using the XML configuration interface and the administration portlets.	You can register EAR files only by using an XML script with `predeployed` mode. You can remove EAR files by using either the administration portlet or an XML script.

Related reference:

"Sample XML configuration files" on page 1111
Sample files are provided for your reference to help illustrate how to use XML configuration for different portal configuration purposes. Before you use them, read the other topics about the XML configuration interface carefully. Many of the samples need to be modified with valid page or user name before they can be used.

Activating and deactivating portlet applications or portlets

By default, portlet applications and portlets are set to an active state after installation. You can deactivate a resource to prevent users from accessing it without changing their user roles. When a portlet application or portlet is in the active state, portal users with appropriate access can include it on their personal pages and customize it. Users that have active references to those inactive portlets on a portal page will see a message stating that the portlet is temporarily disabled.

Before you begin

About this task

You can toggle portlet application or portlet states to active or inactive by using the portal XML configuration interface.

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related reference:

"Sample XML configuration files" on page 1111
Sample files are provided for your reference to help illustrate how to use XML configuration for different portal configuration purposes. Before you use them, read the other topics about the XML configuration interface carefully. Many of the samples need to be modified with valid page or user name before they can be used.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Modifying portlet applications and portlets

You must perform some preparatory tasks before you make your portlets and portlet applications available to your users by putting them on portal pages. This includes installing, deploying, and configuring portlets and applications.

A portlet is initially displayed in the **View** mode. A user can apply customizations to individual, selected, or all instances of the portlet by leaving the View mode and entering into the Personalize, Edit Shared Settings, or Configure mode as appropriate. To access these other modes, hover over the portlet title bar, click ▼ to display the portlet menu, and select the appropriate mode. Refer to the following descriptions for more information about the Personalize, Edit Shared Settings, and Configure modes.

Note: Once you leave the View mode by selecting another portlet mode, you must click **Back** from the portlet menu to return to the View mode. You must return to the View mode before changing to another mode.

- **Personalize:** Updates change the look of the portlet only for the user who makes the updates. In order for a user to have access to personalize settings, they must be granted at least Privileged User role on the page and Privileged User role on the corresponding Portlet Definition. This is available for standard portlets and IBM portlets.

- **Edit Shared Settings:** Updates change the default look of the portlet on a specific page. All users see the change when they access that page. In other words, all (the particular) portlet instances on a page are modified, but not all instances of that portlet on every page. If you want the changes to appear on every page that portlet appears, you must modify the settings on each page. In order for a user to have access to edit shared settings they must be granted at least Editor role on the page and Editor role on the corresponding Portlet definition. This is available for standard portlets and IBM portlets.
- **Configure:** Updates change the default look of a portlet for all portlet instances. All users see the portlet changes on all pages that portlet is available. A user needs to have at least Manager role on portlet definition to have access to configure a portlet.

It depends on the portlet which selection options are available in the portlet menu.

"Configuring portlet applications or portlet parameters"
Many portlet applications and portlets have associated configuration parameters that must be changed after deployment. Manage Applications and Manage Portlets allow you to modify configuration parameters.

Configuring portlet applications or portlet parameters

Many portlet applications and portlets have associated configuration parameters that must be changed after deployment. Manage Applications and Manage Portlets allow you to modify configuration parameters.

About this task

The original configuration is determined by settings in the `portlet.xml` deployment descriptor file. Configuration changes apply to each specific instance of the portlet. WebSphere Portal Express does not validate configuration modification to portlet applications. Some of the portlet parameters can be configured in the portlets themselves by using the portlet modes.

For instructions about the tasks and detailed steps for changing configuration values refer to the **Manage Applications** or **Manage Portlets** helps.

Copying portlet applications

Using the Manage Portlet Application, you can copy a portlet application, which includes some or all of the portlets from the original portlet application. If you want additional instances of a portlet application, use the Manage Portlet Application to make copies.

About this task

Copying can be useful when different portlet configuration parameters are required for different instances of a portlet. A copy of the portlet application can be used to configure the portlet instance for each portlet application. For example, the administrator might want to have multiple Welcome applications. Each copied application would have a different URL, accessing a different content channel, configured for its WelcomePortlet.

When you copy a portlet application, give the copy a new name. The portlet data and the portlet application data are copied, but the copied portlet application uses the same resources as the original portlet application. The new portlet application also uses the same resources as the original portlet application. An example of portlet data is the portlet configuration parameters needed for a particular instance

of a portlet. An example of a portlet resource is an image that the portlet displays in the user interface. While the portlet configuration parameter is associated with each instance of the portlet, all instances of the portlet use the same image.

Results

Only portlets in the copied portlet application are in the new portlet application. For instance, if the copied portlet application contains four portlets and the original contains five portlets, only the four portlets are copied to the new portlet application.

Copying portlets

Use Manage Portlets to copy a portlet. If a portlet is copied or a remote portlet is integrated, a new portlet application is created by the portal that holds the new portlet.

About this task

When you copy a portlet, give the copy a new name. When you copy the portlet, the portlet data is copied, but the copied portlet uses the same resources as the original portlet. When a copied portlet is copied, the new portlet also uses the same resources as the original portlet. An example of portlet data is the portlet configuration parameters needed for a particular instance of a portlet An example of a portlet resource is an image that the portlet displays in the user interface. While the portlet configuration parameter is associated with each individual instance of the portlet, all instances of the portlet use the same image.

Updating Web modules, portlet applications, and portlets

Update a portlet application by updating the WAR file in Manage Web Modules. The newer version of the resource in the WAR file updates the existing portlet application, without breaking the links between user data and the resource. User-specific settings for portlets within the updated resource remain unchanged.

About this task

Before a selected Web module is updated, WebSphere Portal Express checks for an identical resource name in the selected WAR file. If the name of the selected object and the object name in the deployment descriptor of the WAR file do not match, the update is not performed, and a message is displayed.

When a Web module is updated, WebSphere Application Server removes the existing files and installs the updated application in the same directory.

New parameters that are introduced by the updated Web module are always added to the Web module.

Deleting Web modules, portlet applications, or portlets

Web modules can be deleting or uninstalled in Manage Web Modules and portlet applications can be deleted using Manage Applications. Portlets can be deleted using Manage Portlets.

Before you begin

About this task

After selecting an object for deletion, the system waits until all outstanding requests for the portlet application or portlet have been completed and then removes all related files and resources from the portal server. Before removing a portlet application, you must first delete all of its copied portlet applications. If you have the portlet WAR file, you may reinstall the portlet components after deleting it from the system by updating the WAR file in Manage Web Modules. You can reinstall the deleted portlet after ensuring all portlet applications installed in the portlet WAR file have been removed and that the WAR file has been uninstalled.

Important: Do not delete any of the portal administration portlets.

For instructions about the tasks and detailed steps for deleting Web modules, portlet applications, or portlets refer to the **Manage Applications** or **Manage Portlets** helps.

Disabling anchors in portlet URLs

When you access a portlet in IBM WebSphere Portal Express, the portal appends an anchor to the portlet URL. If several portlets are arranged vertically on a page, this appended anchor forces the browser to scroll down to the portlet rather than display the start of the page. You might prefer to have the start of the page that is displayed, even if the link that was clicked points to a portlet, which is placed further down on the page. To achieve this, you can disable the anchors.

About this task

The use of the anchors depends on the configuration of your device, for example your desktop browser. It is controlled by a client device capability called `FRAGMENT_IDENTIFIER`. When the used device, for example the browser, has this capability enabled, the portal adds an anchor to the portlet URL. You can disable the addition of the anchors by deleting the capability `FRAGMENT_IDENTIFIER` from the appropriate clients. Click the **Administration menu** icon. Then, click **Portal Settings** > **Supported Clients**.

If you want to enable the anchors again, restore the `FRAGMENT_IDENTIFIER` capability. Use the same portlet to add it back to the appropriate clients.

Managing iWidgets in your portal

You can add iWidgets to your IBM WebSphere Portal Express.

About this task

Proceed as follows:

Procedure

1. Register the iWidget in the portal. You do this by running the portal configuration task `register-iwidget-definition`. This procedure places the iWidget in a clone of the wrapper portlet. After this the iWidget behaves like a portlet. For information about using the task see *Task register-iwidget-definition*.

2. Place the iWidget on a page by using the portal administration tools for adding portlet to pages. For example, you can use the Manage Pages portlet, the portal administration tools embedded in the theme, or the XML configuration interface (XML Access) (XMLAccess).

   "Task register-iwidget-definition"
   Run the portal configuration task `register-iwidget-definition` to register individual iWidget definitions on WebSphere Portal Express.

   "Task refresh-iwidget-definitions" on page 1252
   Use this configuration task to refresh iWidget definitions in the portal. This task affects all iWidget definitions that are referenced through absolute HTTP or HTTPS URLs in addition to iWidget definitions that are referenced through WebDAV URIs.

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related tasks:

"Manage pages portlets" on page 1273
Use Manage Pages to create, edit, activate, order, and delete pages and external web pages and labels. Available tasks depend on which item is selected. Each page can contain multiple pages. All pages on which you have the User or greater role are displayed in a navigation menu. You must expand pages to access nested pages. The options that you see are dependent upon your access level.

Related reference:

"Putting a portlet on a page" on page 2882
Portal Model REST services allow you to put portlets on pages.

"Task register-iwidget-definition"
Run the portal configuration task `register-iwidget-definition` to register individual iWidget definitions on WebSphere Portal Express.

Related information:

"Customizing pages" on page 1285
The page customizer contains portlets for editing the layout, content, and appearance of pages. It also provides the Wires portlet, which allows users to set up connections between cooperative portlets on a page, and the Locks portlet, which allows users to lock and unlock containers and container content. You can configure the settings for these portlets to show a certain set of functions, restricting basic users from performing more advanced tasks.

Task register-iwidget-definition

Run the portal configuration task `register-iwidget-definition` to register individual iWidget definitions on WebSphere Portal Express.

Identify the iWidget definition by an absolute URL that points to the iWidget definition XML file. Executing this task downloads the iWidget definition XML file from the specified location and creates a corresponding iWidget Wrapper portlet clone. If an iWidget Wrapper portlet clone for the given iWidget URL exists already, no new portlet clone is created, but the existing portlet is updated with the information loaded from the specified iWidget definition XML file.

Syntax: Invoke this task as part of the ConfigEngine script file as follows:

- Linux: `./ConfigEngine.sh register-iwidget-definition -DIWidgetDefinition=`*`IWidget Definition URL`* `-DIWidgetCatalog=dav:fs-type1/iwidgets/`*`IWidget_name`*`/catalog.xml -DPortalAdminPwd=`*`wpsadmin`* `-DWasPassword=`*`wpsadmin`*
- IBM i:
  - **From the UserData directory:** `ConfigEngine.sh register-iwidget-definition -DIWidgetDefinition=`*`IWidget Definition URL`* `-DIWidgetCatalog=dav:fs-type1/iwidgets/`*`IWidget_name`*`/catalog.xml -DPortalAdminPwd=`*`wpsadmin`* `-DWasPassword=`*`wpsadmin`*
- Windows: `ConfigEngine.bat register-iwidget-definition -DIWidgetDefinition=`*`iwidget_Definition_URL`* `-DIWidgetCatalog=dav:fs-type1/iwidgets/`*`IWidget_name`*`/catalog.xml -DPortalAdminPwd=`*`wpsadmin`* `-DWasPassword=`*`wpsadmin`*

Mandatory parameters that you can specify through the command line or in `wkplc.properties`:

`WasUserid`
User ID for WebSphere Application Server

`WasPassword`
Password that corresponds to the user ID for WebSphere Application Server

`PortalAdminId`
Administrator ID for WebSphere Portal Express

`PortalAdminPwd`
Password that corresponds to the administrator ID for WebSphere Portal Express

`WPS_SOAP_PORT`
SOAP port that connects the portal server to remote connections

Mandatory parameters that you can specify only through the command line:

`IWidgetDefinition`
URL of the iWidget definition XML file

Optional parameters that you can specify only through the command line:

`IWidgetCatalog`
URL of an IBM Mashup Center catalog XML file. If you specify this parameter, the task `register-iwidget-definition` parses the referenced catalog XML file to register IWidget definitions or to refresh existing IWidget Wrapper portlet clones. If the parsed catalog XML file contains an entry with a definition element that points to the same iWidget definition file as the parameter `IWidgetDefinition` or the IWidget Wrapper portlet clone that is identified by the parameter `PortletDefinition` , the titles and descriptions from the matching catalog entry are considered for creating or updating the IWidget Wrapper portlet clone. More precisely, the titles and descriptions from the catalog entry are only set on the IWidget Wrapper portlet clone if the corresponding iWidget definition does not define the titles or descriptions in its `idescriptor` item set. When you run the task `register-iwidiget-definition` and provide the IWidgetCatalog parameter but omit `IWidgetDefinition` and `PortletDefinition`, IWidget Wrapper portlet clones for all iWidget definitions referenced in the given catalog XML file are created or updated.

PortletDefinition

Unique name or serialized ObjectID of an iWidget Wrapper portlet clone to refresh. If you specify this parameter, the task `register-iwidgetdefinition` performs a refresh of the referenced iWidget Wrapper portlet clone. This means that the iWidget definition referenced by the existing iWidget Wrapper portlet clone is parsed again to update the portlet.

PortletUniqueName

If this parameter is specified, a new iWidget Wrapper portlet will be created with a unique name as specified by this parameter. If this parameter is specified, the task always creates a new iWidget Wrapper portlet clone independent of the given **IWIdgetDefinition** URL that is already being registered at portal.

Note: If the given unique name is already assigned to some other resource, the task fails and no IWidget Wrapper portlet is created.

Assumptions/Prerequisites: WebSphere Portal Express is running. If WebSphere Portal Express is not running, the task will start it.

Error Conditions: None

Task dependencies: None

Tasks invoked: None

Examples: You can register the iWidget definition XML file located at `http://server_name:port_number/someWidget/someWidget/someWidget.xml` as follows:

```
./ConfigEngine.sh register-iwidget-definition
   -DPortletUniqueName="someWidget"
   -DIWidgetDefinition=http://server_name:port_number/someWidget/someWidget/someWidget.xml
   -DPortalAdminPwd=wpsadmin -DWasPassword=wpsadmin
```

You can refresh the IWidget Wrapper portlet clone with the unique name *someWidget*:

```
./ConfigEngine.sh register-iwidget-definition -DPortletDefinition="someWidget"
   -DPortalAdminPwd=wpsadmin -DWasPassword=wpsadmin
```

You can register the iWidget definition XML file located at `http://server_name:port_number/someWidget/someWidget/someWidget.xml` considering titles and descriptions from the Mashup Center catalog XML file located at `http://server_name:port_number/someWidget/someWidget/catalog.xml`:

```
./ConfigEngine.sh register-iwidget-definition
   -DIWidgetDefinition=http://server_name:port_number/someWidget/someWidget/someWidget.xml
   -DIWidgetCatalog=http://server_name:port_number/someWidget/someWidget/catalog.xml
   -DPortalAdminPwd=wpsadmin -DWasPassword=wpsadmin
```

You can refresh the IWidget Wrapper portlet clone with the unique name *someWidget*, considering titles and descriptions from the Mashup Center catalog XML file located at `http://server_name:port_number/someWidget/someWidget/catalog.xml`:

```
./ConfigEngine.sh register-iwidget-definition
   -DPortletDefinition="someWidget"
   -DIWidgetCatalog=http://server_name:port_number/someWidget/someWidget/catalog.xml
   -DPortalAdminPwd=wpsadmin -DWasPassword=wpsadmin
```

You can perform a bulk registration of iWidget definition XML files or a refresh of IWidget Wrapper portlet clones by using an Mashup Center catalog XML file:

```
./ConfigEngine.sh register-iwidget-definition
   -DIWidgetCatalog=http://server_name:port_number/catalog.xml
   -DPortalAdminPwd=wpsadmin -DWasPassword=wpsadmin
```

Notes:

1. The URL can point to the portal server itself or to an external server. Configure the portal AJAX proxy to allow access to that server. For details refer to the following section.
2. If the definition element of an entry in the Mashup Center catalog XML file contains a relative URL, the URL is resolved into an absolute form using the URL from the **IWidgetCatalog** parameter as a base URL. Only that absolute URL is used for further processing such as accessing the iWidget definition XML file. The absolute URL is also compared with the **IWidgetDefinition** parameter value or the iWidget definition URL of the IWidget Wrapper portlet clone identified by the **PortletDefinition** parameter in order to determine whether or not to consider titles and descriptions from the catalog XML file entry when parsing creating or updating the IWidget Wrapper portlet clone.
3. The **register-iwidget-definition** task only updates titles and descriptions of an IWidget Wrapper portlet clone when you set the following portlet preferences to `true`: `com.ibm.portal.replace.titles` and `com.ibm.portal.replace.descriptions`.
4. When the titles and the descriptions of an IWidgetWrapper portlet clone are set using values from an IBM Mashup Center catalog XML file, the portlet preferences `com.ibm.portal.replace.titles` (default: true) and `com.ibm.portal.replace.descriptions` are set to `false` on the IWidget Wrapper portlet clone. Those portlet preferences prevent the titles and the descriptions of the IWidget Wrapper portlet clone from being overwritten when performing subsequent updates. This is particularly useful when running the **register-iwidget-definition** task to update all IWidget definitions registered in WebSphere Portal Express. The task only considers iWidget definitions that comply with the iWidget specification and does not process IBM Mashup Center catalog XML files.
5. When you refresh IWidget Wrapper portlet clones, values of iWidget attributes (items of the iWidget attributes item set) are not updated unless the attributes are flagged as read only in the iWidget definition XML file. As a result, values of iWidget attributes customized after registering the iWidget in WebSphere Portal Express are prevented from getting lost during a refresh operation.

Registering an iWidget hosted on a Portal server

If you want to register iWidgets deployed as WAR or EAR files directly on your portal server, you can use server relative URLs pointing to the corresponding iWidget definition XML files; for example:

```
./ConfigEngine.sh register-iwidget-definition
     -DIWidgetDefinition=/someWidget/someWidget/someWidget.xml
     -DPortalAdminPwd=wpsadmin -DWasPassword=wpsadmin
```

Remember: Do not forget the leading slash "/" in your iWidget definition URL.

Registering an iWidget hosted on an external server

If you use an iWidget definition URL that points to a server different from your portal server, you need to make sure that your current AJAX proxy configuration

allows accessing this server. For details about this refer to the topics about AJAX proxy Configuration. One way of configuring this is to make sure that the URL to the iWidget definition XML file is mapped to the default_policy dynamic policy in the global AJAX proxy configuration file. You can map a given URL, for example `http://server_name:port_number/someWidget/someWidget/someWidget.xml`, to that policy by adding a custom property such as the following to the WP ConfigService Resource Environment Provider in the WebSphere Application Server administrative console:

```
wp.proxy.config.urlreplacement.default_policy.someID=http://some.server.com:10039/*
```

Notes:

1. The updated policy will not be effective until you restart either the portal server or the AJAX proxy Configuration enterprise application running on the portal server.
2. If you use a URL prefix, do not omit the trailing asterisk (*).

Registering light weight iWidgets stored in the WebDAV file store

Additional to HTTP or HTTPs based iWidget definition URLs, you can also use WebDAV file store URIs pointing to iWidget definition XML files located in the WebDAV file store. Such URIs have the following format:

```
dav:fs-type1/path_to_your_IWidget_definition_XML_file
```

For example, you might have an archive or compressed file with a light weight iWidget that contains an iWidget definition XML file called `someWidget.xml` in the file root folder of the archive or compressed file. In this case you typically first copy the archive or compressed file to the WebDAV file store by using a generic WebDAV client or by using the configuration task `webdav-deploy-zip-file` as follows:

```
./ConfigEngine.sh webdav-deploy-zip-file
     -DZipFilePath=/tmp/SomeWidgetPackage.zip
     -DTargetURI=dav:fs-type1/iwidgets/SomeWidget/
     -DPortalAdminPwd=wpsadmin -DWasPassword=wpsadmin
```

This extracts your light weight iWidget package represented by the file `/tmp/SomeWidgetPackage.zip` into the folder `/iwidgets/SomeWidget` of the WebDAV file store. Make sure to add the trailing slash (`/`) in the parameter `TargetURI`. After the extraction you can register the iWidget definition XML file `someWidget.xml` as follows:

```
./ConfigEngine.sh register-iwidget-definition
     -DIWidgetDefinition=dav:fs-type1/iwidgets/SomeWidget/someWidget.xml
     -DPortalAdminPwd=wpsadmin -DWasPassword=wpsadmin
```

Task refresh-iwidget-definitions

Use this configuration task to refresh iWidget definitions in the portal. This task affects all iWidget definitions that are referenced through absolute HTTP or HTTPS URLs in addition to iWidget definitions that are referenced through WebDAV URIs.

Refreshing an iWidget definition in this context means, reloading the iWidget definition XML files and updating the corresponding iWidget Wrapper portlet clone accordingly. You can run the task in a synchronous or asynchronous manner. If you run this task asynchronously, the configuration task completes immediately after you start a corresponding asynchronous system task. You can use this mode to avoid timeout problems that might occur if there are a high number of iWidget definitions to be refreshed. The completion of the system task is indicated in `SystemOut.log`.

Notes about this task:

- `refresh-iwidget-definitions` updates the titles and the descriptions of an iWidget Wrapper portlet clone only if you set the following portlet preferences to `true`: **`com.ibm.portal.replace.titles`** and **`com.ibm.portal.replace.descriptions`**. If these parameters do not exist in the installed iWidget, you must add them and set their values to `true`.
- If the lines with the iWidget `item id="title"` and `item id="description"` in the iWidget XML file do not include the option `readOnly="true"` set, the default command does not work. To correct this issue, you can choose between the following options:
  - In the portlet XML file, add the `readOnly="true"` option to the lines with `item id="title"` and `item id="description"` as follows:

    ```
    <iw:item id="title" lang="en" value="iWidget Title" readOnly="true">
      ...
    <iw:item id="description" lang="en" value="iWidget Description" readOnly="true">
    ```

  - When you run the ConfigEngine task, include the option `-DForceRefresh=true` option as follows:

    ```
    ./ConfigEngine.sh refresh-iwidget-definitions
                      -DIWidgetDefinition=/iWidget/iWidget.xml
                      -DForceRefresh=true
    ```

  - Add the following portlet parameter to the installed iWidget: `com.ibm.portal.replace.attributes = true`.
- When you refresh an iWidget Wrapper portlet clone, values of iWidget attributes, or the items of the iWidget attributes item set, are not updated. The values are updated if the attributes are flagged as read-only in the iWidget definition XML file. As a result, values of iWidget attributes that are customized after you register the iWidget in WebSphere Portal Express are prevented from becoming lost during a refresh operation.

Syntax: Invoke this task as part of the ConfigEngine script as follows:

- Linux: `./ConfigEngine.sh refresh-iwidget-definitions`
- IBM i:
  - **From the `UserData` directory:** `ConfigEngine.sh refresh-iwidget-definitions`
- Windows: `ConfigEngine.bat refresh-iwidget-definitions`

Mandatory parameters that you can specify through the command line or in `wkplc.properties`:

`WasUserid`
User ID for WebSphere Application Server.

`WasPassword`
Corresponding password for WebSphere Application Server.

`PortalAdminId`
User ID for WebSphere Portal Express.

`PortalAdminPwd`
Corresponding password for WebSphere Portal Express.

Optional parameters that you can specify only through the command line:

`Synchronous`
Specify one of the following values:

The value `true` to run the task in synchronous mode. This value is the default setting.

The value `false` to run this task in asynchronous mode.

Scheduling the `refresh-iwidget-definitions` task

You can schedule the task `refresh-iwidget-definitions` by using the following task: `com.ibm.portal.services.RefreshIWidgetDefinitionsTask`. See *XML configuration reference* for instructions.

Administering managed pages

You can run advanced administration tasks for managed pages, such as generating URLs for projects or working with projects by using scripts, or working with vanity URLs.

"Project URL generation" on page 1255
You can redirect request processing to a specific project by generating URLs with the ProjectIdentificationService API, or the REST API. Request processing operates either completely within the scope of a project or completely outside the scope of a project. You cannot switch projects during request processing.

"Access control for managed pages" on page 1256
Access control for managed pages provides more capabilities than access control for standard portal pages. In addition to the access control features available for pages through portal administration, you can also apply IBM Web Content Manager features, like workflow and syndication, to access control.

"Portal Scripting Interface and project support" on page 1260
With the Portal Scripting Interface, you can create Jacl or Jython scripts to automate the management of projects.

"Portal Scripting Interface and web content libraries" on page 1270
With the Portal Scripting Interface, you can create Jacl or Jython scripts to automate the management of web content libraries. Using the DocumentLibrary bean with the Portal Scripting Interface, you can create and delete libraries, retrieve a list of libraries, and retrieve library attributes.

"XML configuration interface and managed pages" on page 1271
You can use the XML configuration interface (XML Access) to manipulate managed pages just as you can for other portal resources.

"Lost-found site area" on page 1273
When Web Content Manager cannot determine the proper location of a page item in the Portal Site library, the page item is stored in the **lost-found** site area. This site area ensures that you can recover pages and any content that is stored beneath the pages when a problem occurs.

Related concepts:

"Vanity URLs" on page 2094
You can associate vanity URLs with portal pages and labels. Vanity URLs are short URLs that people can easily remember. They are shorter than full WebSphere Portal Express URLs. They are sometimes also called marketing URLs. You can publish vanity URLs for marketing campaigns through different channels, such as email or print. This way, you can use vanity URLs to direct customers to a specific portal page or content item. Interested site visitors who want to view your campaign can then remember or copy the short vanity URL and type it into the browser address field.

Related reference:

"Staging to production list" on page 2487
The items in the production list can be included in the staging to production Portal Application Archive (PAA) file. However, if you are not using the staging to production PAA file, use this list to determine the tools that are required to move your artifacts from the staging server to the production server. You can also use this list to determine what tool to use for the items not included in the staging to production PAA file.

Project URL generation

You can redirect request processing to a specific project by generating URLs with the ProjectIdentificationService API, or the REST API. Request processing operates either completely within the scope of a project or completely outside the scope of a project. You cannot switch projects during request processing.

When a request originates from within a project, the request URL contains a project identifier for that project. The project information is included only in the URL and is not bound to the session. The project identifier can be an object ID (OID), as used by the portal, or a universally unique identifier (UUID), as used by Web Content Manager. To direct request processing to a specific project, you must generate a URL for the project and then render the URL.

Java API

To generate URLs that target a project by using the Java API in the portal, you can use the ProjectIdentificationService API with the StateManagerService API:

- The ProjectIdentificationService provides methods to create a `ServerContext` object, based on the identifier of the target project and the current `ServerContext` object.
- The project-specific `ServerContext` object can then be used to retrieve a `URLFactory` object from the state manager service. All URLs generated with this factory contain the project ID.

This example constructs a portal URL to the current navigational state for a new project:

```
// construct a server context for the project
final ServerContext projectCtx = projectService.createServerContext(
  projectID, stateService.getServerContext());

// access the URL factory to create a URL
final URLFactory urlFct = stateService.getURLFactory(projectCtx);

// construct a URL to the current state
final EngineURL url = urlFct.newURL(Constants.SMART_COPY);
url.writeDispose(out);

// done with URL generation
urlFct.dispose();
```

This example constructs a portal URL to a URI in a specific project:

```
// construct a server context for the project
final PocServerContext projectCtx = projectService.createServerContext(
  projectID, pocService.getServerContext());

// access the URL factory to create a URL
final DisposablePocURLFactory urlFct = pocService
  .getURLFactory(projectCtx);

// construct a URL to the current state
```

```
final PocURL url = urlFct.newURL(PocURLFactory.LATE_BINDING);
url.setMode(Constants.VALUE_DOWNLOAD);
url.setURI(new URI("test:abc"));

// serialize
url.writeDispose(out);

// done with URL generation
urlFct.dispose();
```

REST API

If your application uses the Representational State Transfer (REST) architecture, you can use the remote APIs provided with the portal to construct project-specific URLs.

Access control for managed pages

Access control for managed pages provides more capabilities than access control for standard portal pages. In addition to the access control features available for pages through portal administration, you can also apply IBM Web Content Manager features, like workflow and syndication, to access control.

When you create a managed page in the portal, a corresponding page item is created in a web content library. You can view and change access control settings for a managed page in two ways:

- By navigating to the page and using the site toolbar
- By opening the corresponding page item in the web content authoring portlet

Regardless of the method you use to change an access control setting, the corresponding element is automatically updated. This synchronization ensures that effective permissions are coordinated between the portal page and the web content page item.

Special considerations

As managed pages integrate features from portal pages and Web Content Manager, there are special considerations that apply with access control for managed pages.

Unified set of applicable roles with different effective capabilities

With managed pages, portal pages and Web Content Manager are aware of the same roles; however, some roles are effectively ignored in Web Content Manager. For example, the roles of Privileged User and Markup Editor are used with portal pages to support features such as personalizing a page. In Web Content Manager, these roles have no effect on access control.

When you perform a web content action on a managed page, like previewing, publishing, or syndicating the page, Web Content Manager accounts for the portal roles. This awareness ensures that pages retain their appropriate permissions from the roles. For details on the portal roles, see *Roles*.

Virtual groups in Web Content Manager (authors, owners, creators)

In Web Content Manager you can grant access to virtual groups (authors, owners, creators) through the web content authoring portlet or as part of a workflow stage. Portal pages do not provide an equivalent mechanism. When you grant permissions on a page item to users or groups with the virtual groups, direct role mappings are assigned on the portal page. These role mappings ensure that equal permissions are applied.

The owner virtual group, however, is limited to a single owner for page items in Web Content Manager. The owner of the portal page is automatically synchronized with the owner of the page item. This owner has the same set of allowed actions as the Manager role, as described in the *Ownership* section of *Roles.*

Important: If you are using author or creator groups for access control management in Web Content Manager, use only the authoring portlet to perform access control tasks. Do not use the site toolbar in the portal interface to revoke permissions, because doing so can lead to a potentially complex assignment of permissions.

Traversal support for portal pages

With portal pages, traversal support provides implicit permissions that enable users to navigate through a page hierarchy. For example, a user might have permission to access a child page but might not have permission to access the parent page. Because of traversal support, the user is permitted to navigate to the child page. See *Roles* for details on traversal support.

However, traversal support is not provided for web content items. Content authors that use the authoring portlet must be assigned the User role on all pages higher than the child page to navigate to editable content. Without this access permission, the editable content is not visible in the authoring portlet, even though the author can access the page. Typical page administration tasks can still be performed from the page.

Permissions granted through virtual resources

With traditional portal pages, you can grant permissions on the virtual resources PORTAL and CONTENT_NODES that inherit permissions to the complete page hierarchy. This inheritance is described in *Resources.* You can also specify a similar inheritance for web content libraries that inherit from the root node.

Because permissions for managed pages are synchronized between portal pages and page items in Web Content Manager, such inheritance is problematic. This inheritance can result in different effective permissions on portal pages and content items. Although you can manage permissions correctly either through the page or the authoring portlet, the preferred approach is through the page. If you grant permissions to the entire page hierarchy with inheritance, grant this permission on the root resource for the page hierarchy (`wps.content.root` page). As the permission on this page node is synchronized to the corresponding page item in Web Content Manager, the effective permissions are automatically synchronized throughout the hierarchy.

Access control permissions managed by workflows

When working with managed pages, you can apply access control to page items through workflow stages and actions, as described in *Workflow and change management* In addition to permissions from the workflow, you can also modify permissions on the page with the site toolbar. Changes that you make with the site toolbar override the access permissions in effect with the current workflow stage. When the next workflow stage is entered, changes from the site toolbar are reset and the permissions specified by the workflow stage take effect.

Access control permissions for site areas and pages work only under Web Content Manager, but not under WebSphere Portal Express

Set the access permissions for the portal site library. Click the

Administration menu icon. Then, click **Portal Content** > **Web Content Libraries**. These access permissions restrict what users can do with a portal page site area item only under Web Content Manager. For example, only a user with the appropriate access permission can add content under a page or add a workflow. However, a user without such access permission might still be able to create, update, or delete pages on the WebSphere Portal Express side, if the user has the appropriate access permission on the page.

External security support
You cannot use externalized roles or role mappings with managed pages. Pages cannot be externalized while being edited in a project. Similarly, externalized resources cannot be added to a project.

Required permissions

The following permissions are required for typical actions with managed pages.

Table 169. Required permissions for typical actions with managed pages

Action	Required permission
Access a project view in the site toolbar	User on the WCM_REST_SERVICE virtual resource
View a project in the site toolbar	• User on the WCM_REST_SERVICE virtual resource, in addition to the permissions that are required to view a specific project • User on the selected project
Create a project	• Contributor on the Portal Site library • User on the WCM_REST_SERVICE virtual resource
Create new items	You set the access permission for creating new items at the library level, not at the item level. To create a new item, a user must have at least the following access permissions: • Contributor access on a library • Editor access on an item-type If a user has access permission to create an item type, the user can also create folders and projects.
Create a draft of a published page by editing the page in a project	• Editor on the page • User on the selected project
Create a draft of a published page with the **Create Draft** action in the site toolbar.	• User on the page and Approver on the corresponding web content page item. For details, see “Approver role for creating draft pages” on page 1260. • User on the selected project
Create a draft child page under a parent page in a project	• Contributor or Editor on the parent page • User on the selected project

Table 169. Required permissions for typical actions with managed pages (continued)

Action	Required permission
Preview a project	• Can Run As User on the USERS virtual resource • The user that is impersonated requires at least User access to the current portal page. If an anonymous user does not have access to the page, the **As Unauthenticated User** preview option is not available in the site toolbar. In addition, if you select the **As User** preview option, you cannot select users that do not have access to the page. • User on the selected project By default only users and unauthenticated users that have explicit access to the project can preview the project. You can globally assign access for users or unauthenticated users to view all items in all libraries and projects in a specific virtual portal or the default virtual portal. To assign these rights, use the **Set root access** setting in the library administration portlet. Click the **Administration menu** icon. Then, click **Portal Content** > **Web Content Libraries**.
Create web content by adding web content viewer to a page. The viewer is configured to create and render content from a web content library.	• Editor on the page • User on the viewer portlet • No library permissions are enforced.
Perform inline editing of content on a page	• Editor on the page • Appropriate permissions on the library that contains the content

For the required permissions for portal pages and web content items, see *Access permissions* for portal pages and *User roles and access* for web content items.

The default set of access control permissions for anonymous users and for members of the All Authenticated Users group are described in *Initial Access Control Settings.* With managed pages, the following default permissions exist:

- Anonymous users can view projects and have User access to the Portal Site library.
- Members of the All Authenticated Users group can create new projects and have Editor access to the Portal Site library. This access ensures that users can perform inline editing tasks. You can restrict access as needed with the library administration portlet.

To modify a portal page or page item, you require only those permissions that are needed to perform the action from the user interface or programming API. You do not also require permissions for the underlying synchronization actions that take place automatically. These automatic updates are performed with system privileges.

For example, you might add a portlet to a page by using the site toolbar. In this case, you require sufficient permissions on the page that you are editing and on the portlet that you are adding. However, you do not need additional permissions for the internal updates to the corresponding items in the web content library.

Approver role for creating draft pages

With managed pages, you can use a workflow to enable business users to create draft versions of pages that they are normally not allowed to edit. By using a workflow in this way, you accomplish two things:

- You provide business users with the ability to modify pages.
- You can still ensure that the drafts are reviewed and approved by technical users before the changes are published to the external site.

Typically a user with User access to a page has permission only to view the page. But if the user also has Approver access to the corresponding page item in the Portal Site library, the user can create page drafts. When a user has this access, the user can navigate to the portal page and use the site toolbar to create a draft.

To enable business users to create draft pages, complete the following steps:

1. In the Portal Site library, assign a workflow to the page items that correspond to the portal pages that you want users to modify. By default, page items are not managed in a workflow.
2. Edit the publish stage of the workflow, and update the access control properties to add the users to the Approver role.
3. Edit the initial draft stage of the workflow, and update the access control properties. Add the users to the roles that correspond to the permissions that the users require on the draft pages that they create.

Contributor role for creating child pages

Users with Contributor access to the published version of a page can create child pages under that page. When in edit mode on the parent page, contributors can use the site toolbar to create a child page.

Portal Scripting Interface and project support

With the Portal Scripting Interface, you can create Jacl or Jython scripts to automate the management of projects.

Using the Project bean with the Portal Scripting Interface, you can perform the following actions on projects:

- List all available projects
- Create and delete projects
- Set an active project
- Retrieve information about a specific project
- Retrieve locale-specific attributes for projects
- Submit a project for review
- Withdraw a project from review.
- Approve projects
- Decline a project
- Approve all documents of a project, that is all Web Content Manager items that are parts of a workflow

- Decline all documents of a project, that is all Web Content Manager items that are parts of a workflow
- Publish projects

To run commands with the Project bean, you can use the Portal bean to set a project as the context for subsequent commands.

List projects

To retrieve a list of projects, use the `listall` method. This method returns the names of the projects.

- Jacl syntax: `$Project listall`
- Jython syntax: `Project.listall()`

Jacl example:

```
wsadmin>$Project listall
'"TestProject1" "TestProject2"'
```

Jython example:

```
wsadmin>Project.listall()
'"TestProject1" "TestProject2"'
```

Create projects

To create a project, use the `create` method.

- Jacl syntax: `$Project create "`*`project_name`*`"`
- Jython syntax: `Project.create("`*`project_name`*`")`

Jacl example:

```
wsadmin>$Project create "TestProject1"
TestProject1
```

Jython example:

```
Project.create("TestProject1")
'TestProject1'
```

Note: If you create a project with the Portal Scripting Interface, the project is not listed with the recent projects in the project menu.

Delete projects

To delete a project, use the `delete` method.

- Jacl syntax: `$Project delete "`*`project_name`*`"`
- Jython syntax: `Project.delete("`*`project_name`*`")`

Jacl example:

```
wsadmin>$Project delete "TestProject1"
```

Jython example:

```
wsadmin>Project.delete("TestProject1")
```

Set active project

For commands that you want to run within a project, use the `setproject` method of the Portal bean to specify the project. When invoking the `setproject` method, you identify the active project with the name of the project. If you invoke the `setproject` method without specifying a project name, the active project is cleared. When you set the project during a session, the project is active immediately.

To set the active project, you must establish a user session with the portal by using the **login** command of the Portal bean.

- Jacl syntax: `$Project setproject "project_name"`
- Jython syntax: `Portal.setproject("project_name")`

Jacl example:

```
wsadmin>$Portal setproject "TestProject1"
```

Jython example:

```
wsadmin>Portal.setproject("TestProject1")
```

Retrieve project details

Retrieve project details with the `details` method. This method returns the following information about the project:

- Universally Unique Identifier (UUID)
- State
- Name
- Title
- Approvers
- Outstanding approvals
- Approval mode, that is whether all approvers need to approve or only one approver needs to approve
- History, that is which actions have been performed on the project before.

Syntax:

- Jacl syntax: `$Project details "project_name"`
- Jython syntax: `Project.details("project_name")`

Jacl example:

```
wsadmin>$Project details "TestProject2"
uuid                 : 47c7f3bd-b004-4b94-a6b5-397272d69eb7
state                : ACTIVE
name                 : TestProject2
title                : TestProject2
approvers            : wpsadmins
outstanding approvals: wpsadmins
approval mode        : all
history              : 9/24/13 1:02 PM: Document created by wpsadmin
                       9/24/13 1:06 PM: Submitted for review by wpsadmin
                       9/24/13 1:06 PM: Project state changed to Review
                       9/24/13 6:34 PM: Rejected by wpsadmin
                       9/24/13 6:34 PM: Project state changed to Rejected
                       9/24/13 6:34 PM: Project state changed to Active
                       9/24/13 6:35 PM: Submitted for review by wpsadmin
                       9/24/13 6:35 PM: Project state changed to Review
                       9/24/13 6:35 PM: Rejected by wpsadmin
                       9/24/13 6:35 PM: Rework project with more detail.
                       9/24/13 6:35 PM: Project state changed to Rejected
                       9/24/13 6:35 PM: Project state changed to Active
items:
  testpagep2 (draft / new / publish pending)
```

Jython example:

```
wsadmin>print Project.details("TestProject2")
uuid                 : 47c7f3bd-b004-4b94-a6b5-397272d69eb7
state                : ACTIVE
name                 : TestProject2
title                : TestProject2
approvers            : wpsadmins
```

```
outstanding approvals: wpsadmins
approval mode        : all
history              : 9/24/13 1:02 PM: Document created by wpsadmin
                       9/24/13 1:06 PM: Submitted for review by wpsadmin
                       9/24/13 1:06 PM: Project state changed to Review
                       9/24/13 6:34 PM: Rejected by wpsadmin
                       9/24/13 6:34 PM: Project state changed to Rejected
                       9/24/13 6:34 PM: Project state changed to Active
                       9/24/13 6:35 PM: Submitted for review by wpsadmin
                       9/24/13 6:35 PM: Project state changed to Review
                       9/24/13 6:35 PM: Rejected by wpsadmin
                       9/24/13 6:35 PM: Rework project with more detail.
                       9/24/13 6:35 PM: Project state changed to Rejected
                       9/24/13 6:35 PM: Project state changed to Active
items:
  testpagep2 (draft / new / publish pending)
```

Retrieve translated attributes

If any project attributes are translated, such as the title or description, you can retrieve those attributes with the `nlsget` method. Specify the attribute with one of the following parameters:

- Title: **`title`** or **`t`**
- Description: **`description`**, **`descr`**, or **`d`**

Syntax:

- Jacl syntax: `$Project nlsget "`*`project_name`*`" `*`attribute_parameter`*` [`*`locale`*`]`
- Jython syntax:
  `Project.nlsget("`*`project_name`*`","`*`attribute_parameter`*`"[,"`*`locale`*`"])`

Jacl example:

```
wsadmin>$Project nlsget "TestProject1" descr en
This is the description for TestProject1.
```

Jython example:

```
wsadmin>print Project.nlsget("TestProject1", "descr", "en")
This is the description for TestProject1.
```

If you do not specify a value for the *`locale`* parameter, the currently selected locale is used.

Submit a project for review

To submit a project for review, use the `submitforreview` method. The project needs to be in active state, and all contained documents need to be in publish, pending, or deleted state. When you submit a project for review, the project moves into the review state.

- Jacl syntax: `$Project submitforreview "project_name"`
- Jython syntax: `Project.submitforreview("`*`project_name`*`")`

Jacl example:

```
wsadmin>$Project submitforreview TestProject

wsadmin>$Project details TestProject
uuid                 : 035c4488-973f-44a4-aa8e-92b9f36e2412
state                : REVIEW
name                 : TestProject
title                : TestProject
approvers            : wpsadmins
outstanding approvals: wpsadmins
approval mode        : single
```

```
history              : 10/16/13 6:23 PM: Document created by wpsadmin
                       10/16/13 6:26 PM: Submitted for review by wpsadmin
                       10/16/13 6:26 PM: Project state changed to Review
items:
  testpage (draft / new / publish pending)
```

Jython example:

```
wsadmin>Project.submitforreview("TestProject")

wsadmin>print Project.details("TestProject")
uuid                 : 035c4488-973f-44a4-aa8e-92b9f36e2412
state                : REVIEW
name                 : TestProject
title                : TestProject
approvers            : wpsadmins
outstanding approvals: wpsadmins
approval mode        : single
history              : 10/16/13 6:23 PM: Document created by wpsadmin
                       10/16/13 6:26 PM: Submitted for review by wpsadmin
                       10/16/13 6:26 PM: Project state changed to Review
items:
  testpage (draft / new / publish pending)
```

Withdraw a project from review

To withdraw a project from review, use the `withdrawfromreview` method. The project needs to be in review state. When you withdraw a project from review, the project moves into the active state.

- Jacl syntax: `$Project withdrawfromreview "project_name"`
- Jython syntax: `Project.withdrawfromreview("`*`project_name`*`")`

Jacl example:

```
wsadmin>$Project withdrawfromreview TestProject

wsadmin>$Project details TestProject
uuid                 : 035c4488-973f-44a4-aa8e-92b9f36e2412
state                : ACTIVE
name                 : TestProject
title                : TestProject
approvers            : wpsadmins
outstanding approvals: wpsadmins
approval mode        : single
history              : 10/16/13 6:23 PM: Document created by wpsadmin
                       10/16/13 6:26 PM: Submitted for review by wpsadmin
                       10/16/13 6:26 PM: Project state changed to Review
                       10/16/13 6:26 PM: Rejected by wpsadmin
                       10/16/13 6:26 PM: needs rework
                       10/16/13 6:26 PM: Project state changed to Rejected
                       10/16/13 6:26 PM: Project state changed to Active
                       10/16/13 6:27 PM: Submitted for review by wpsadmin
                       10/16/13 6:27 PM: Project state changed to Review
                       10/16/13 6:28 PM: Withdrawn from review by wpsadmin
                       10/16/13 6:28 PM: Project state changed to Active
items:
  testpage (draft / new / publish pending)
```

Jython example:

```
wsadmin>Project.withdrawfromreview("TestProject")

wsadmin>print Project.details("TestProject")
uuid                 : 035c4488-973f-44a4-aa8e-92b9f36e2412
state                : ACTIVE
name                 : TestProject
title                : TestProject
approvers            : wpsadmins
```

```
outstanding approvals: wpsadmins
approval mode        : single
history              : 10/16/13 6:23 PM: Document created by wpsadmin
                       10/16/13 6:26 PM: Submitted for review by wpsadmin
                       10/16/13 6:26 PM: Project state changed to Review
                       10/16/13 6:26 PM: Rejected by wpsadmin
                       10/16/13 6:26 PM: needs rework
                       10/16/13 6:26 PM: Project state changed to Rejected
                       10/16/13 6:26 PM: Project state changed to Active
                       10/16/13 6:27 PM: Submitted for review by wpsadmin
                       10/16/13 6:27 PM: Project state changed to Review
                       10/16/13 6:28 PM: Withdrawn from review by wpsadmin
                       10/16/13 6:28 PM: Project state changed to Active
items:
  testpage (draft / new / publish pending)
```

Approve projects

To approve the drafts in a project, use the `approve` method. Additional to the project, you can add a comment as a second parameter. If the project setup requires a comment, the comment is mandatory. If not, the comment is optional.

The command approves only the project. The drafts in the project need to be approved separately, for example by using the `approvedocuments` method. Before you can approve a project, all of its documents must be in publish, pending, or deleted state.

- Jacl syntax:

  ```
  $Project approve "project_name"
  $Project approve "project_name" "comment"
  ```

- Jython syntax:

  ```
  Project.approve("project_name")
  Project.approve("project_name", "comment")
  ```

Jacl example:

```
wsadmin>$Project approve TestProject "ok"

wsadmin>$Project details TestProject
uuid                 : 035c4488-973f-44a4-aa8e-92b9f36e2412
state                : PENDING
name                 : TestProject
title                : TestProject
approvers            : wpsadmins
outstanding approvals:
approval mode        : single
history              : 10/16/13 6:23 PM: Document created by wpsadmin
                       10/16/13 6:26 PM: Submitted for review by wpsadmin
                       10/16/13 6:26 PM: Project state changed to Review
                       10/16/13 6:26 PM: Rejected by wpsadmin
                       10/16/13 6:26 PM: needs rework
                       10/16/13 6:26 PM: Project state changed to Rejected
                       10/16/13 6:26 PM: Project state changed to Active
                       10/16/13 6:27 PM: Submitted for review by wpsadmin
                       10/16/13 6:27 PM: Project state changed to Review
                       10/16/13 6:28 PM: Withdrawn from review by wpsadmin
                       10/16/13 6:28 PM: Project state changed to Active
                       10/16/13 6:28 PM: Submitted for review by wpsadmin
                       10/16/13 6:28 PM: Project state changed to Review
                       10/16/13 6:28 PM: Approved by wpsadmin
                       10/16/13 6:28 PM: ok
                       10/16/13 6:28 PM: Project state changed to Pending
items:
  testpage (draft / new / publish pending)
```

Jython example:

```
wsadmin>Project.approve("TestProject1")
wsadmin>print Project.details("TestProject1")
uuid                 : 035c4488-973f-44a4-aa8e-92b9f36e2412
state                : PENDING
name                 : TestProject
title                : TestProject
approvers            : wpsadmins
outstanding approvals:
approval mode        : single
history              : 10/16/13 6:23 PM: Document created by wpsadmin
                       10/16/13 6:26 PM: Submitted for review by wpsadmin
                       10/16/13 6:26 PM: Project state changed to Review
                       10/16/13 6:26 PM: Rejected by wpsadmin
                       10/16/13 6:26 PM: needs rework
                       10/16/13 6:26 PM: Project state changed to Rejected
                       10/16/13 6:26 PM: Project state changed to Active
                       10/16/13 6:27 PM: Submitted for review by wpsadmin
                       10/16/13 6:27 PM: Project state changed to Review
                       10/16/13 6:28 PM: Withdrawn from review by wpsadmin
                       10/16/13 6:28 PM: Project state changed to Active
                       10/16/13 6:28 PM: Submitted for review by wpsadmin
                       10/16/13 6:28 PM: Project state changed to Review
                       10/16/13 6:28 PM: Approved by wpsadmin
                       10/16/13 6:28 PM: ok
                       10/16/13 6:28 PM: Project state changed to Pending
items:
  testpage (draft / new / publish pending)
```

Decline projects

To decline the drafts in a project, use the `decline` method. Additional to the project, you can add a comment as a second parameter. If the project setup requires a comment, the comment is mandatory. If not, the comment is optional. The command only declines the project. The drafts in the project remain in their current state. For a project to be declined, the project needs to be in review state.

- Jacl syntax:

  ```
  $Project decline "project_name"
  $Project decline "project_name" "comment"
  ```

- Jython syntax:

  ```
  Project.decline("project_name")
  Project.decline("project_name", "comment")
  ```

Jacl example:

```
wsadmin>$Project decline TestProject "needs rework"

wsadmin>$Project details TestProject
uuid                 : 035c4488-973f-44a4-aa8e-92b9f36e2412
state                : ACTIVE
name                 : TestProject
title                : TestProject
approvers            : wpsadmins
outstanding approvals: wpsadmins
approval mode        : single
history              : 10/16/13 6:23 PM: Document created by wpsadmin
                       10/16/13 6:26 PM: Submitted for review by wpsadmin
                       10/16/13 6:26 PM: Project state changed to Review
                       10/16/13 6:26 PM: Rejected by wpsadmin
                       10/16/13 6:26 PM: needs rework
                       10/16/13 6:26 PM: Project state changed to Rejected
                       10/16/13 6:26 PM: Project state changed to Active
items:
  testpage (draft / new / publish pending)
```

Jython example:

```
wsadmin>Project.decline("TestProject1")
wsadmin>print Project.details("TestProject1")
uuid                 : 035c4488-973f-44a4-aa8e-92b9f36e2412
state                : ACTIVE
name                 : TestProject
title                : TestProject
approvers            : wpsadmins
outstanding approvals: wpsadmins
approval mode        : single
history              : 10/16/13 6:23 PM: Document created by wpsadmin
                       10/16/13 6:26 PM: Submitted for review by wpsadmin
                       10/16/13 6:26 PM: Project state changed to Review
                       10/16/13 6:26 PM: Rejected by wpsadmin
                       10/16/13 6:26 PM: needs rework
                       10/16/13 6:26 PM: Project state changed to Rejected
                       10/16/13 6:26 PM: Project state changed to Active
items:
  testpage (draft / new / publish pending)
```

Approve all documents of a project

To approve all documents of a project, use the `approvedocuments` method. It approves all Web Content Manager items that are parts of a workflow.

- Jacl syntax: `$Project approvedocuments` *project_name*
- Jython syntax: `Project.approvedocuments("`*project_name*`")`

Jacl example:

```
wsadmin>$Project approvedocuments TestProject

wsadmin>$Project details TestProject
uuid                 : 5dffe3e4-59d1-4abf-8697-3e62a29a0cbf
state                : PENDING
name                 : TestProject
title                : TestProject
approvers            :
outstanding approvals:
approval mode        : single
history              : 10/16/13 6:14 PM: Document created by wpsadmin
                       10/16/13 6:16 PM: Project state changed to Pending
items:
  testpage (draft / new / publish pending)
```

Jython example:

```
wsadmin>Project.approvedocuments("TestProject")

wsadmin>Project.details("TestProject")
uuid                 : 5dffe3e4-59d1-4abf-8697-3e62a29a0cbf
state                : PENDING
name                 : TestProject
title                : TestProject
approvers            :
outstanding approvals:
approval mode        : single
history              : 10/16/13 6:14 PM: Document created by wpsadmin
                       10/16/13 6:16 PM: Project state changed to Pending
items:
  testpage (draft / new / publish pending)
```

Decline all documents of a project

To decline all documents of a project, use the `declinedocuments` method. This method declines all Web Content Manager items that are parts of a workflow.

- Jacl syntax: `$Project declinedocuments` *project_name*

- Jython syntax: `Project.declinedocuments("project_name")`

Jacl example:

```
wsadmin>$Project declinedocuments TestProject
wsadmin>$Project details TestProject
uuid                : 5dffe3e4-59d1-4abf-8697-3e62a29a0cbf
state               : ACTIVE
name                : TestProject
title               : TestProject
approvers           :
outstanding approvals:
approval mode       : single
history             : 10/16/13 6:14 PM: Document created by wpsadmin
                      10/16/13 6:16 PM: Project state changed to Pending
                      10/16/13 6:18 PM: Project state changed to Active
items:
  testpage (draft / new)
```

Jython example:

```
wsadmin>$Project.declinedocuments("TestProject")
wsadmin>$Project.details("TestProject")
uuid                : 5dffe3e4-59d1-4abf-8697-3e62a29a0cbf
state               : ACTIVE
name                : TestProject
title               : TestProject
approvers           :
outstanding approvals:
approval mode       : single
history             : 10/16/13 6:14 PM: Document created by wpsadmin
                      10/16/13 6:16 PM: Project state changed to Pending
                      10/16/13 6:18 PM: Project state changed to Active
items:
  testpage (draft / new)
```

Publish projects

To publish a project, use the `publish` method. Before you can publish a project, all items in the project must be approved and the project needs to be in publish pending state.

- Jacl syntax: `$Project publish "project_name"`
- Jython syntax: `Project.publish("project_name")`

Jacl example:

```
wsadmin>$Project publish TestProject

wsadmin>$Project details TestProject
uuid                : 035c4488-973f-44a4-aa8e-92b9f36e2412
state               : PUBLISHED
name                : TestProject
title               : TestProject
approvers           : wpsadmins
outstanding approvals:
approval mode       : single
history             : 10/16/13 6:23 PM: Document created by wpsadmin
                      10/16/13 6:26 PM: Submitted for review by wpsadmin
                      10/16/13 6:26 PM: Project state changed to Review
                      10/16/13 6:26 PM: Rejected by wpsadmin
                      10/16/13 6:26 PM: needs rework
                      10/16/13 6:26 PM: Project state changed to Rejected
                      10/16/13 6:26 PM: Project state changed to Active
                      10/16/13 6:27 PM: Submitted for review by wpsadmin
                      10/16/13 6:27 PM: Project state changed to Review
                      10/16/13 6:28 PM: Withdrawn from review by wpsadmin
                      10/16/13 6:28 PM: Project state changed to Active
                      10/16/13 6:28 PM: Submitted for review by wpsadmin
```

```
                          10/16/13 6:28 PM: Project state changed to Review
                          10/16/13 6:28 PM: Approved by wpsadmin
                          10/16/13 6:28 PM: ok
                          10/16/13 6:28 PM: Project state changed to Pending
                          10/16/13 6:29 PM: Project state changed to Publishing
                          10/16/13 6:29 PM: Project state changed to Published
        items:
          testpage (published)
```

Jython example:

```
        wsadmin>Project.publish("TestProject1")
        wsadmin>print Project.details("TestProject1")
        uuid                : 035c4488-973f-44a4-aa8e-92b9f36e2412
        state               : PUBLISHED
        name                : TestProject
        title               : TestProject
        approvers           : wpsadmins
        outstanding approvals:
        approval mode       : single
        history             : 10/16/13 6:23 PM: Document created by wpsadmin
                              10/16/13 6:26 PM: Submitted for review by wpsadmin
                              10/16/13 6:26 PM: Project state changed to Review
                              10/16/13 6:26 PM: Rejected by wpsadmin
                              10/16/13 6:26 PM: needs rework
                              10/16/13 6:26 PM: Project state changed to Rejected
                              10/16/13 6:26 PM: Project state changed to Active
                              10/16/13 6:27 PM: Submitted for review by wpsadmin
                              10/16/13 6:27 PM: Project state changed to Review
                              10/16/13 6:28 PM: Withdrawn from review by wpsadmin
                              10/16/13 6:28 PM: Project state changed to Active
                              10/16/13 6:28 PM: Submitted for review by wpsadmin
                              10/16/13 6:28 PM: Project state changed to Review
                              10/16/13 6:28 PM: Approved by wpsadmin
                              10/16/13 6:28 PM: ok
                              10/16/13 6:28 PM: Project state changed to Pending
                              10/16/13 6:29 PM: Project state changed to Publishing
                              10/16/13 6:29 PM: Project state changed to Published
        items:
          testpage (published)
```

Examples

These examples demonstrate a typical command sequence to create a page within a specific project. Each example script performs the following operations:

- Establishes a user session.
- Creates a project (`myproject`).
- Retrieves the details for the `myproject` project.
- Sets the active project to the `myproject` project.
- Locates the `Home` page in the portal.
- Creates the `testpage1` page as a child page of the `Home` page. This operation takes place within the context of the active project.
- Clears the active project.
- Terminates the user session.

Jacl example:

```
        $Portal login
        set myproject [$Project create "My new project"
        $Project details $myproject
        $Portal setproject $myproject
```

```
$Content find any un ibm.portal.Home select
$Content create page testpage1 html shared public
$Portal setproject
$Portal logout
```

Jython example:

```
Portal.login()
myproject = Project.create("My new project")
Project.details(myproject)
Portal.setproject(myproject)
Content.find("any", "un", "ibm.portal.Home", "select")
Content.create("page","testpage1","html", "shared", "public")
Portal.setproject()
Portal.logout()
```

Portal Scripting Interface and web content libraries

With the Portal Scripting Interface, you can create Jacl or Jython scripts to automate the management of web content libraries. Using the DocumentLibrary bean with the Portal Scripting Interface, you can create and delete libraries, retrieve a list of libraries, and retrieve library attributes.

Create libraries

To create a library, use the create method.

- Jacl syntax: `$DocumentLibrary create "`*`library_name`*`"`
- Jython syntax: `DocumentLibrary.create("`*`library_name`*`")`

Jacl example:

```
wsadmin>$DocumentLibrary create "Library1"
"library1"
```

Jython example:

```
wsadmin>DocumentLibrary.create("Library1")
'"library1"'
```

Delete libraries

To delete a library, use the delete method.

- Jacl syntax: `$DocumentLibrary delete "`*`library_name`*`"`
- Jython syntax: `DocumentLibrary.delete("`*`library_name`*`")`

Jacl example:

```
wsadmin>$DocumentLibrary delete "Library1"
```

Jython example:

```
wsadmin>DocumentLibrary.delete("Library1")
```

List libraries

To retrieve a list of libraries, use the listall method.

- Jacl syntax: `$DocumentLibrary listall`
- Jython syntax: `DocumentLibrary.listall()`

Jacl example:

```
wsadmin>$DocumentLibrary listall
"template page content" "wiki template v70" "blog template v70" "web resources v70"
"portal site" "web content templates" "blog solo template v70" "web content"
"library1"
```

Jython example:

```
wsadmin>DocumentLibrary.listall()
'"template page content" "wiki template v70" "blog template v70" "web resources v70"
"portal site" "web content templates" "blog solo template v70" "web content"
"library1"'
```

Retrieve library details

Retrieve library details with the details method. This method returns the following information about the library: the Universally Unique Identifier (UUID), state, name, and title.

- Jacl syntax: `$DocumentLibrary details "library_name"`
- Jython syntax: `DocumentLibrary.details("library_name")`

Jacl example:

```
wsadmin>$DocumentLibrary details "Library1"
uuid       : 64fa541a-a189-4ed6-8a6f-4c3dcc148295
name       : library1
title      : Library1
description:
enabled            : true
deletion prohibited: false
```

Jython example:

```
wsadmin>print DocumentLibrary.details("Library1")
uuid       : 13b06eb0-52c7-415b-9a93-4195968aa2a3
name       : library1
title      : Library1
description:
enabled            : true
deletion prohibited: false
```

XML configuration interface and managed pages

You can use the XML configuration interface (XML Access) to manipulate managed pages just as you can for other portal resources.

Project scope

When you use the XML configuration interface with managed pages, processing occurs either completely within a project or completely outside a project.

When you import a published page and specify a project scope, the page is created as a draft page in the project.

Important: You cannot export a draft page and then import that page as a draft in another project.

Important: Web Content Manager items, such as workflow items, categories, or keywords, are not exported or imported by the XML configuration interface (XML Access). You need to use the XML configuration interface (XML Access) library export and import feature instead.

To run the **xmlaccess** command from within a project, specify the project either with the project name or with the object ID of the project. The command uses the following format, depending on how you identify the project:

```
xmlaccess -in input_file -url http://hostname:port_number/wps/config/$project/project_name
xmlaccess -in input_file -url http://hostname:port_number/wps/config/$project/project_object_id
```

For example:

- `xmlaccess -in Export.xml -url http://www.example.com:10039/wps/config/$project/myproject`
- The following command must be entered all in one line:

  ```
  xmlaccess -in Export.xml -url
  http://www.example.com:10039/wps/config/$project/
  Z6QReDeN9E86O46P9CGJMK633P8JMG6J1P8MM47MPD6MMCC63PI3IL6GPD63R46J1
  ```

Note: Linux: You might have to precede the dollar sign ($) with a backslash (\) to prevent `$project` from being interpreted as an environment variable. For example:

```
xmlaccess -in Export.xml -url http://www.example.com:10039/wps/config/\$project/myproject
```

Use transaction processing with the XML configuration interface

Because pages are stored in the Portal Site library in Web Content Manager, each page has corresponding objects in the JCR database. You must be aware of this relation when you create, update, or delete pages with the XML configuration interface. If **`xmlaccess`** processing is interrupted, it can result in a mismatch between the page state and database state.

WARNING: If you redeploy your site daily, your JCR size increases because of page versions. Periodically clean up your versions to reduce the JCR size. Go to "Clearing version history" on page 1194 for information.

To ensure that page and database information for a page remain synchronized, use the `transaction-level` attribute of the `request` element in the XML file. For more information about using the `transaction-level` attribute, go to *XML configuration reference.*

Example:

```
<request
    type="update"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
    transaction-level="resource">
```

Excluding pages from being managed in Web Content Manager

By default, you can manage all WebSphere Portal Express pages in Web Content Manager except for the portal administration pages. When an administrator creates or imports a page, the portal checks whether the parent page has a portal page site area in Web Content Manager. If it does, the portal creates a portal page site area for the new page in Web Content Manager as well.

You can exclude pages from being managed in Web Content Manager. You can prevent the creation of the corresponding page item in Web Content Manager. The page is then not available to be managed in managed pages. It is good practice to exclude only pages on first-level nodes, for example the Search or Applications pages. To do so, run an XML configuration interface script and use the `content-mapping-info` element with the attribute `has-system-mapping`:

has-system-mapping=true | false
Use this flag as follows:

true When the XML script creates or imports a page, the portal creates a page item in Web Content Manager as well.

false When the XML script creates or imports a page, the portal does not create a page item in Web Content Manager.

(not specified)

If you do not set a value for this flag in the XML script, it follows the portal where the page is imported:

- If managed pages are enabled, it creates a page item in Web Content Manager as well.
- If managed pages are disabled, it does not create a page item in Web Content Manager.

Note: There is no indication in the user interface whether a page is managed in Web Content Manager or not. Therefore, excluding parts of a site can lead to an inconsistent portal user experience for portal site visitors.

Lost-found site area

When Web Content Manager cannot determine the proper location of a page item in the Portal Site library, the page item is stored in the **lost-found** site area. This site area ensures that you can recover pages and any content that is stored beneath the pages when a problem occurs.

The following causes are typical reasons that a page might be stored in the lost-found site area:

- The synchronization process between the portal and Web Content Manager finds page items in the Portal Site library that do not have corresponding pages in the portal.
- You move a managed page beneath an unmanaged page in the portal page hierarchy. In this case, the page item for the managed page is stored in the lost-found site area.

The lost-found site area is a read-only location. Although you cannot edit items in the lost-found site area, you can view the items and copy content that you want to save.

Manage pages portlets

Use Manage Pages to create, edit, activate, order, and delete pages and external web pages and labels. Available tasks depend on which item is selected. Each page can contain multiple pages. All pages on which you have the User or greater role are displayed in a navigation menu. You must expand pages to access nested pages. The options that you see are dependent upon your access level.

About this task

Use the Manage Pages portlet to complete the following tasks:

- Create, reorder, delete, and edit the properties of pages, labels, and URLs
- Reorder pages, labels, and URLs
- Assign access to pages, labels, and URLs
- Move pages to a new location in the portal hierarchy

Both administrators and users with appropriate access can create and delete pages. Users can delete only the pages that they create or the pages for which they have at least Manager access. User can create pages from the site toolbar.

"Pages and page types: derived and hidden pages"
Understanding the behavior of derived pages, and hidden pages, and the differences between them can help you manage pages and create the correct type of page for your needs.

"Selecting pages" on page 1279
Pages that you can modify in Manage Pages appear as links in a table. By clicking a link to select a page, you can navigate to other nested pages to perform page tasks. You can also select labels and URLs in Manage Pages.

"Using friendly URLs" on page 1280
You can associate friendly URLs with portal pages and labels. You and your users can use these friendly URLs to access specific portal pages or labels by using a human readable path, which is easy to remember.

"Task refresh-page-layout" on page 1284
Use this configuration task to refresh the page layout for all pages assigned a given page layout template.

"Customizing pages" on page 1285
The page customizer contains portlets for editing the layout, content, and appearance of pages. It also provides the Wires portlet, which allows users to set up connections between cooperative portlets on a page, and the Locks portlet, which allows users to lock and unlock containers and container content. You can configure the settings for these portlets to show a certain set of functions, restricting basic users from performing more advanced tasks.

Related information:

Tuning for many pages

Pages and page types: derived and hidden pages

Understanding the behavior of derived pages, and hidden pages, and the differences between them can help you manage pages and create the correct type of page for your needs.

A page is an organization element that defines how content is displayed. A page can contain portlets and other pages. Derived pages are children of the original pages and have specific behavior that you must be aware of.

Derived pages inherit the properties of the original page. Creating a derived page is equivalent to creating a new, specialized layer on the original page. The original page and the new layer are aggregated together at rendering time. The new layer is contained within and controlled by the original page. You can give administrative access to other users by referencing an existing page while you maintain the content and layout from the original page. The following implications apply to derived pages:

- If content is locked on the page that is referenced, content is locked on all derived pages that reference that page.
- If a portlet is deleted from the page that is referenced, the portlet is deleted from all pages that reference that page. All individual user settings for that portlet are also lost.
- The user must have User role access to the original page to access the derived page. Therefore, private pages cannot be shared in this manner.

Changes made to the original parent page might be reflected to the derived pages that reference it. Layers can be created on other layers to create a chain of cascading pages, referred to as delegated page specialization. This process means that a root page can be created, and the top-level administrator can decide the

initial layout and content of the page. The next level administrator can then control and modify a specialized layer of the root page, adding more content and layout. This process can continue down a chain of page managers and submanagers. Managers or submanagers in the chain see only their individual layer of the chain. However, they must have the User role for every layer higher than theirs to see the content of the previous layers. A user is only able to see a layer of the page if appropriate access is given. Here are some examples to illustrate this concept.

John, the superadministrator, creates a page Home and titles it Home. Brandy, a subadministrator, manages the next level of the home page, named Home_operations. Brandy determines what more content needs be added to the Home page for employees in the operations group. Nick, the next level administrator, manages the next level of the Home page, Home_operations_transportation. Nick determines the content that needs to be available on the Home page for employees in the transportation department. Nick, as the transportation page administrator, must have the Manager role for Home_operations_transportation to change the page. The changes that Nick makes affects all users, and the User role for Home_operations and the User role for Home. Nick must have the User role on every layer that combines to create the Home_operations_transportation level. Desi, a user of the Home_operations_transportation page, must have the User role for Home_operations_transportation, and she must also have the User role for Home_operations and the User role for Home. When Desi, the user logs on to the portal, she sees one Home page. This Homepage is an aggregation of all the layers that are associated with the root Home page.

Notes:

- If you delete a page that is referenced by another page, all pages that reference the deleted page are also deleted.
- The markup that is specified for the root page cannot be modified on derived pages. The whole derivation tree structure with all layers supports the markup that is specified on the root page.
- You can disable the ability to change the title and description of derived pages by setting the **allow.derived.titles** parameter in the portal WP Configuration Service in the WebSphere Integrated Solutions Console. For details, see the description for the **allow.derived.titles** parameter in the topic about the portal Configuration Service.

  "Behavior of derived pages in combination with locks and changing access permissions" on page 1276
  Using and altering locks in conjunction with access permissions on the parent pages may result in changes on derived pages depending on the complexity of the derivation structure. The following scenario describes the behavior of derived pages.

Related tasks:

"Hiding and displaying pages in the navigation" on page 1803
By default, pages that you create are displayed in the navigation of the portal site. If you do not want a page that you create to appear in the navigation, you can hide the page by setting the **com.ibm.portal.Hidden** page parameter to `true`. While this parameter does not affect your portal access control settings for the page, it is hidden from the navigation.

Hidden pages

Hidden pages do not show in the portal, but contain portlets that can be opened from other pages. The hidden pages do not appear in the site navigation, but are started from generated links in portlets or theme code. The feature to set a page

that you create as a hidden page is not available in the UI. Follow the steps in the topic *Hiding and displaying pages in the navigation* to set a page as hidden. For ease of administration and conserving system resources, you can place and manage hidden pages in one place. An example is the Edit Page Properties portlet: users can start it from a link in the theme, but the portlet instance itself is on a hidden page in the content model.

Some scenarios and use cases require such hidden pages. The easiest way to create a hidden page for such purposes is to create the page as a child of the **Hidden Pages** label, which is a child of the content root. This label is hidden from the navigation. It is a container for hidden pages in the portal, and it minimizes the cycles that are required to render the top-level navigation links while still providing support for hidden pages.

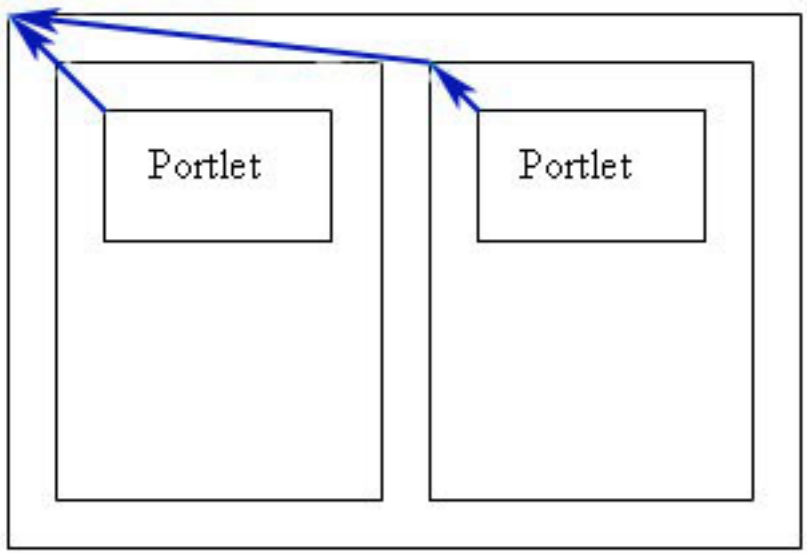
Set your custom theme when you create a hidden page, do not inherit the theme. If you use legacy themes, be aware that they render the top-level navigation that is based on the level that is specified in theme policy or page metadata to render the navigation at the appropriate level. If you use such a theme and you create a new hidden page under the Hidden Pages label, set the metadata value `com.ibm.portal.themepolicy.topNavigationStartLevel` for the page to 3.

Behavior of derived pages in combination with locks and changing access permissions

Using and altering locks in conjunction with access permissions on the parent pages may result in changes on derived pages depending on the complexity of the derivation structure. The following scenario describes the behavior of derived pages.

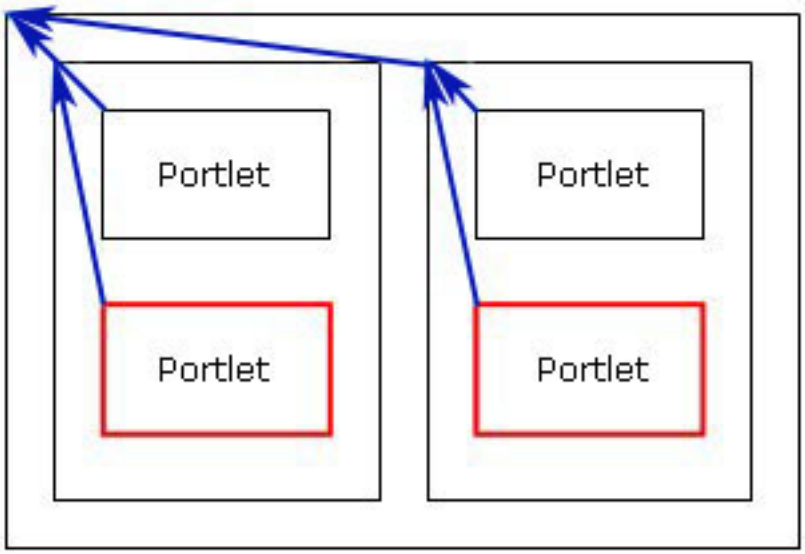
An administrator with the editor role creates a page with a two-column layout and places a portlet in each column. The blue arrows in the following figure indicate the child-to-parent relationship of the components of the page:

Two column layout:



Then a user with an editor role creates an explicit derivation from this page (under **Advanced Options**, selects **A page that uses content from a shared page**) and adds two portlets to the explicitly derived page, one portlet in each column as indicated in the following figure:

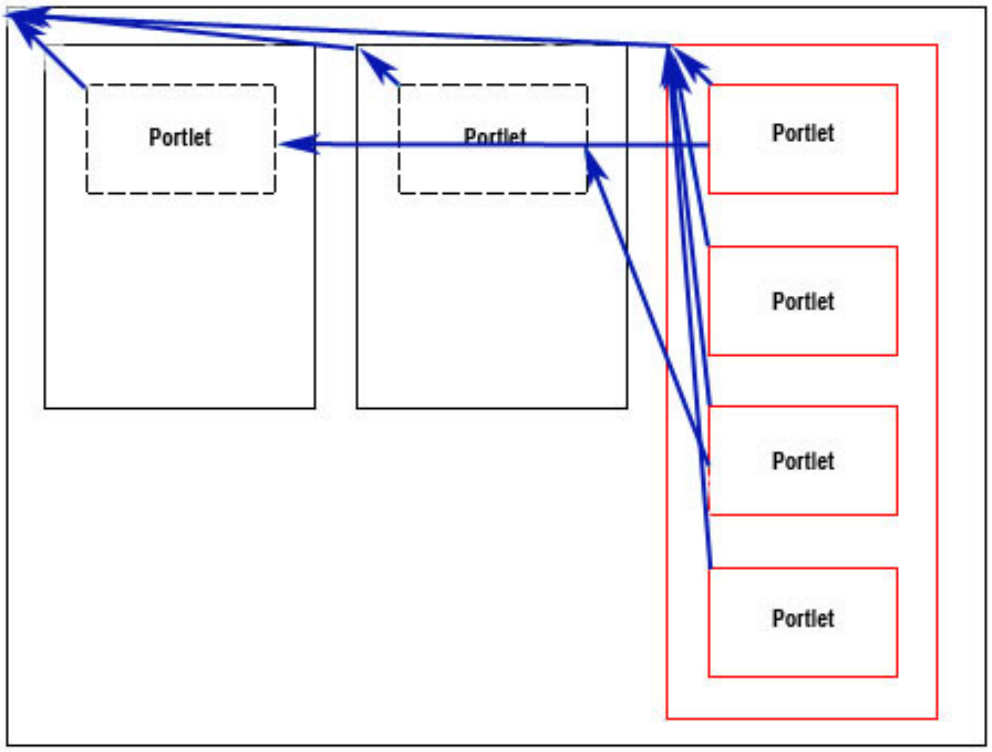
Two column four portlet layout:



The red boxes around the two new controls indicate that these controls are on a new *layer* of the page. The original page content and the new layer together comprise the complete page.

The user with an editor role creates a third column next to the two existing columns and moves all portlets to the new column. The new column exists on the layer of the derived page (red box); whereas, the original portlets are moved into the column through additional information (green arrows) as indicated in the following figure:

Three column layout:



Rendering the explicitly derived page shows all four portlets arranged vertically as shown in the previous picture.

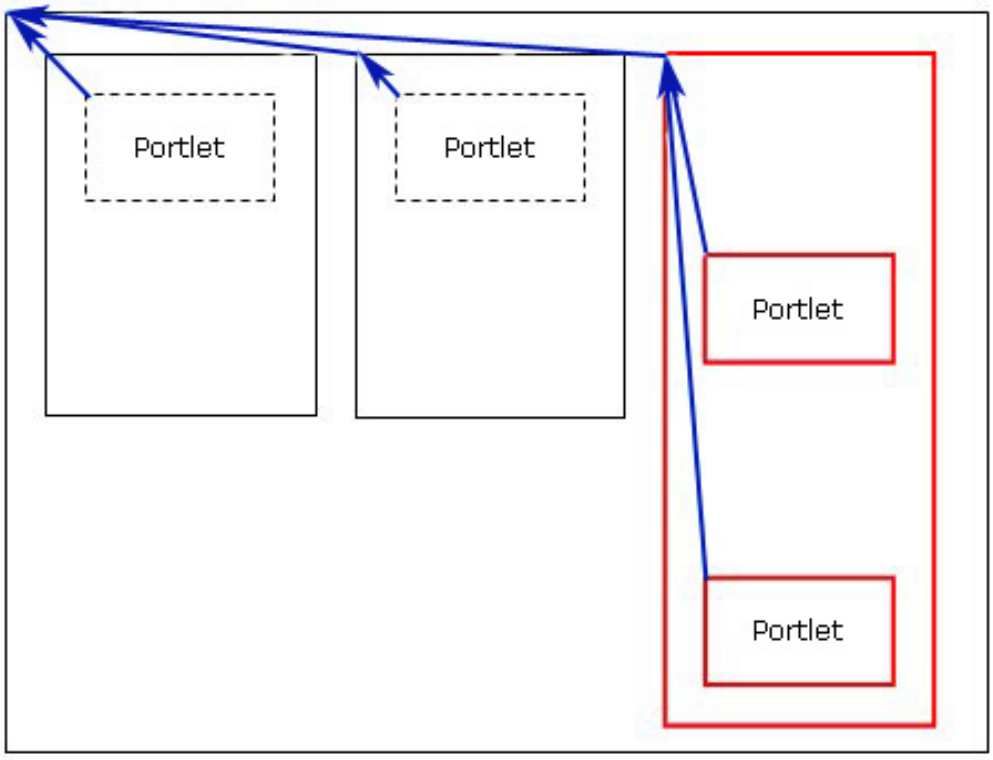
Next, the administrator locks all containers and portlets on the original page. This has no impact on the aggregated pages (original page and derived page). The administrator then removes the editor permission from the user who created the derived page and assigns only privileged user permissions to that user.

If this privileged user navigates to the derived page, the following issues occur:

- The administrator set the locks on the original page, which enforces the layout of the original page for the derived page. Therefore, the information that causes the original portlets to be moved into the new column (the green arrows in the three column layout figure) is deleted.
- Due to the locks, the additions on the extra layer of the derived page (red boxes) are suppressed. The privileged user sees the derived page exactly as the original page.

If the administrator reassigns editor permissions to the user for the derived page, the layout appears differently to the user from any of the stages of the derived page. The layer of additions to the derived page becomes visible again but the information about moving the original portlets has been deleted as indicated in the following figure:

Three column layout with missing information about moving the portlets:



The same principle applies to other layer scenarios in similar ways. Movement information of elements from the original page is deleted; whereas, actions on the layer of the derived page; for example adding portlets, rows, or columns; may persist after reassigning permissions as previously described.

Related information:

IBM WebSphere Portal Page Derivation Concepts

Selecting pages

Pages that you can modify in Manage Pages appear as links in a table. By clicking a link to select a page, you can navigate to other nested pages to perform page tasks. You can also select labels and URLs in Manage Pages.

About this task

Manage Pages allows you to create, modify and delete pages, URLs, and labels. Any page that you are able to modify appears in a table, and the icons that display depend upon the tasks that you have permission to perform. If links exist in the **Page title** column, use those links to navigate to child pages of the selected page. As you perform page tasks, you will leave Manage Pages, but you will return to Manage Pages after completing each task.

For example, if you are able to modify the title of the page titled **My page**, clicking the Edit Page Properties icon takes you to Properties to change the page title and returns you to when you complete your changes.

- Displayed icons and links are dependent upon the tasks you have permission to perform.
- Upon completion of each task, you will be returned to Manage Pages.
- To access pages that are nested under others, click the Page title.

Using friendly URLs

You can associate friendly URLs with portal pages and labels. You and your users can use these friendly URLs to access specific portal pages or labels by using a human readable path, which is easy to remember.

Before you begin

For a friendly URL to work for a specific page, you must define a friendly URL name for every page or label in the path of the portal page hierarchy that leads to that page. You can do this in the page properties. Friendly URLs take the following general form:

`http://host_name:port_number/PortalServer_root/portal/page_id/[!ut/p/encoded_portal_suffix]`

The `page_id` portion of the friendly URL is made up of the friendly URL names of all pages in the path of the page hierarchy. This path begins at the content root and ends with the page for which you want to give your users a friendly URL.

Example: You have a portal page that is named **Products** in the user interface and has a friendly URL name `products`. Under this Products page you have another page, which is named **Appliances** and has a friendly URL name `appliances`. To access the **Appliances** page, users can type the following friendly URL into the browser address field:

`http://www.example.com:10039/wps/portal/products/appliances`

About this task

You can configure a friendly URL by using the portal toolbar or the Manage Pages administration portlet. To configure a friendly URL by using the Manage Pages portlet, proceed as follows:

Procedure

1. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
2. Locate the page for which you want to configure a friendly URL.
3. Click the **Edit Page Properties** icon.
4. In the field **Friendly URL name**, type the friendly name for the page.
5. Click **OK** to save your changes.
6. Repeat this procedure for every page or label in the path of the portal page hierarchy that leads to the target page.

What to do next

To make up the full IBM WebSphere Portal Express URL, the portal appends a suffix to that friendly URL. This suffix represents the current state of the page and its components. Some scenarios require short and fully human readable URLs that omit the state information. For information about how to configure short stateless URLs see *Using friendly URLs without state information*.

"Using friendly URLs without state information" on page 1281
By default, WebSphere Portal Express URLs include navigational state information. If you configure pages for friendly URLs, the portal appends the state information to the friendly URLs. Some scenarios require short and fully

human readable URLs that omit the state information. For such scenarios, you can configure friendly URLs so that the portal does not show that state information.

Using friendly URLs without state information

By default, WebSphere Portal Express URLs include navigational state information. If you configure pages for friendly URLs, the portal appends the state information to the friendly URLs. Some scenarios require short and fully human readable URLs that omit the state information. For such scenarios, you can configure friendly URLs so that the portal does not show that state information.

Before you begin

If you are on combined cumulative fix 4 or earlier, go to "CF04 and earlier: Using friendly URLs without state information" on page 3631.

About this task

State information is an encoded aggregation of the navigational state of the portal and the portlet.

- The portal state includes page selection, expansions, label mapping, and action targets.
- The portlet state includes render parameters, window state, and portlet mode.

The presence of state information within the URL enables characteristics of dynamic websites, such as usage of bookmarks or the **Back** button. For example, when state information is included, users can bookmark a page and later return to the exact same state of that page.

Some scenarios require short and fully human readable URLs that omit the state information:

- You do not want the URL to make the impression that it references dynamic content.
- You want the URL to contain only information that a human person can read and interpret.
- You want the URL to easily fit into the address field of the web browser.
- Internet search engines expect static URLs that reference only one resource or web page for the time that the page exists.
- Internet search engines prefer short and friendly URLs.

For such scenarios, you can configure WebSphere Portal Express as follows:

- You configure themes to always display only short friendly URLs without state information.
- You configure pages that use that theme to display friendly URLs.

The configuration applies to all pages that use that theme and that are configured to display friendly URLs.

Notes:

- If you configure your portal to show stateless friendly URLs, you gain improved URL readability at the cost of losing the state functionality:
  - Portal URLs always point to the default state of a page because they do not contain the state information.
  - If a user clicks the **Back** button, or refreshes a page by clicking the **Refresh** button or the page title, the page moves back into the default View mode.

- If a user views a page and creates a bookmark, the bookmark later opens the page in the default View mode.
- Stateless friendly URLs do not contain information about the language of the page. The portal determines the language for the page by using the following steps:
  1. First, the portal looks for the user preference.
  2. If the user preference is not set, the portal looks for the preferred language that is set in the browser. If the page is a public page, the user is an anonymous user. In this case, the portal also looks for the preferred language that is set in the browser.
  3. If the portal cannot determine a preferred language setting for the portal or the browser, it applies the default language that is defined for the portal.

  You might want to present language-specific portal pages with stateless friendly URLs. For more information, see step 8 on page 1283 of the following procedure.

- Make sure that the JSPs of your theme provide a `<base>` tag in the header section of your markup. For example, you can use the portal tag `<portal-core:stateBase/>`. For more information, read *<portal-core/> tags*.

Procedure

1. Optional: Disable friendly URL redirects. If the incoming URL does not contain the friendly URL prefix of the addressed page, a URL redirect adds state information to the URL. To disable URL redirects, set the custom property `friendly.redirect.enabled` to the value `false` in the Resource Environment Provider (REP) WPConfigService in the WebSphere Integrated Solutions Console. If the property is not listed there, add it and set it to `false`. For information about this property and how to set it, see the topics *Configuration service* and *Setting service configuration properties*.
2. In the theme that you want to configure for stateless friendly URLs, set the parameter `com.ibm.portal.theme.hasBaseURL` to `true`. You can update the theme parameter by using the XML configuration interface. The following example shows a sample XML script:

```
<?xml version="1.0" encoding="UTF-8"?>

<request
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
 type="update">

 <!-- This sample sets the hasBaseURL Tag in the Portal 8.5 Theme. -->
 <portal action="locate">
 <theme action="update" uniquename="ibm.portal.85Theme" >
 <parameter name="com.ibm.portal.theme.hasBaseURL"
  type="string" update="set">true</parameter>
 </theme>
 </portal>
</request>
```

3. Make sure that no generated URLs in the theme include state information. In the default theme, you can complete this step by modifying the `navigation.jsp` and `sideNavigation.jsp` files. Use these steps to change the file `navigation.jsp`:

a. Change to the directory for the file navigation.jsp. You can find the file in the following location: *PortalServer_root*\theme\wp.theme.themes\default85\installedApps\DefaultTheme85.ear\DefaultTheme85.war\themes\html\dynamicSpots
b. Open the file navigation.jsp with an editor.
c. Search for the string node.urlGeneration and change it to one of the following stringa: CF05 node.urlGeneration.noNavigationalState

Use the following steps to change the file sideNavigation.jsp:

a. Change to the directory *PortalServer_root*\theme\wp.theme.themes\default85\installedApps\DefaultTheme85.ear\DefaultTheme85.war\themes\html\dynamicSpots.
b. Open the file sideNavigation.jsp with an editor.
c. Search for the string node.urlGeneration and change it to the following string: CF05 node.urlGeneration.noNavigationalState

4. Change the mobileNavigation.jsp file:
   a. Change to the directory *PortalServer_root*\theme\wp.theme.themes\default85\installedApps\DefaultTheme85.ear\DefaultTheme85.war\themes\html\dynamicSpots.
   b. Open the file mobileNavigation.jsp with an editor.
   c. Locate the string ?uri=nm:oid:${nodeID} and change it to the following string: CF05 ${node.urlGeneration.noNavigationalState}.
5. Change the mobileNavigationFeed.jsp file:
   a. Change to the directory *PortalServer_root*\theme\wp.theme.themes\default85\installedApps\DefaultTheme85.ear\DefaultTheme85.war\themes\html\dynamicSpots.
   b. Open the file mobileNavigationFeed.jsp with an editor.
   c. Search for the string node.url and change it to the following string: CF05 node.urlGeneration.noNavigationalState
6. Optional: If you are using IBM Web Content Manager, you might want the IBM Web Content Manager Rendering portlet to also display the friendly and stateless URLs. In this case, implement a plug-in that translates the IBM Web Content Manager URLs into the required custom format. For instructions and sample code for such a plug-in, see *Example 2, Generate a friendly URL for web content*.
7. Define friendly URL names for pages. For information about how to define friendly URLs, read *Using friendly URLs*.
8. Optional: You might want to present language-specific portal pages with stateless friendly URLs to your site visitors. In this case, structure your portal site to reflect which pages are targeted for specific countries or regions. You can create a node for a specific page, and then create language-specific child pages under that node. For example, in the node home, you create pages in English, French, and German. Your site visitors can then access the page in their preferred language by using one of the following friendly URLs:

   ```
   http://www.myco.com/wps/home/en/shop
   http://www.myco.com/wps/home/fr/shop
   http://www.myco.com/wps/home/de/shop
   ```

9. Restart the Portal server.

Results

The portal now no longer displays the navigational state information within the URLs.

Related reference:

"Example 2: Generate a friendly URL for web content" on page 3240
This example demonstrates a content URL generation filter that generates a friendly URL for web content.

"Configuration Service" on page 297
The portal Configuration Service is responsible for collecting the most essential configuration data of the IBM WebSphere Portal Express engine.

Related information:

"<portal-core/> tags" on page 2791
The <portal-core/> tags are used to provide portal core functionality such as entering the main render flow as well as URL-related aspects of the page.

Task refresh-page-layout

Use this configuration task to refresh the page layout for all pages assigned a given page layout template.

Description: You need to run this task if you have manually modified page layout template files stored in the folder `layout-templates` of the portal file store and you want those changes to be reflected on existing pages. The task reads the data of a specific page layout template from the portal file store and update all pages that are currently assigned that page template accordingly. Depending on the number of page affected, this task can run for a several minutes.

Assumptions/Prerequisites: WebSphere Portal Express is running. The execution of this task requires that the portal is running. If the portal is not running, the task starts the portal.

Usage: Reference the layout template by its file store URI, for example as follows:

```
/fs-type1/themes/Portal8.5/layout-templates/
```

Note: Make sure to provide the full URI here, especially if the URI has a trailing slash (/).
You can execute the task in synchronous or asynchronous manner. If you execute it asynchronously, the configuration task completes immediately after starting a corresponding asynchronous system task. You can use this mode to avoid timeout problems that might occur if a high number of pages are to be refreshed. The completion of the system task is indicated by a corresponding message in the file `SystemOut.log`.

Syntax: Start this task as part of the ConfigEngine script file as follows:

- For Linux: `./ConfigEngine.sh refresh-page-layout-template`
- For IBM i: From the UserData directory: `ConfigEngine.sh refresh-page-layout-template`
- For Windows: `ConfigEngine.bat refresh-page-layout-template`

Mandatory parameters to be specified on the command line or in the file `wkplc.properties` :

- **`WasUserid`** The WebSphere Application Server user ID
- **`WasPassword`** The WebSphere Application Server password

- **PortalAdminId** The portal administrator user ID
- **PortalAdminPwd** The portal administrator password

Mandatory parameter to be specified on the command line only:

- **PageLayoutTemplate** The file store URI of the page layout template

  Note: Make sure to provide the full URI here, especially if the URI has a trailing slash (/).

Optional parameters to be specified on the command line only:

- **VirtualPortalContext** or **VirtualPortalHost** Use this parameter to identify the virtual portal. Only pages contained in the specified virtual portal are refreshed. If you omit this parameter, by default no virtual portal page layout is refreshed.
- **Synchronous (=true)** Use this parameter to specify the execution mode. The default is synchronous. If you want to run this task in asynchronous manner, specify false .

Example:

```
./ConfigEngine.sh refresh-page-layout-template
     -DPageLayoutTemplate=dav:fs-type1/layout-templates/2Columnequal/
     -DVirtualPortalContext=virtual_portal_context_url
```

Customizing pages

The page customizer contains portlets for editing the layout, content, and appearance of pages. It also provides the Wires portlet, which allows users to set up connections between cooperative portlets on a page, and the Locks portlet, which allows users to lock and unlock containers and container content. You can configure the settings for these portlets to show a certain set of functions, restricting basic users from performing more advanced tasks.

Skins represent the border around a portlet, including its title bar. Users with appropriate access can select skins for individual portlets using the Appearance portlet of the page customizer. To set a skin for portlets, access the Page Customizer and select the Appearance portlet. See the portlet help for further information.

A page is deactivated when you start making changes to it. While editing the page, all changes are effective immediately and cannot be undone or cancelled. Other users will not be able to access the page and icons that launch the edit mode of portlets on the page are deactivated until you commit the changes by clicking **Done**.

Editing a shared page can have different results, depending on the role assigned to the user editing the page.

- Users with the Editor role for a page can make changes that affect all users of the page.
- Users with the Privileged User role for a page can only make changes to a private copy, or layer, of the page. The changes do not affect the other users of the page. The page must have an initial layout with at least one container before users with the Privileged User role can make any changes to the page.
- Users with the User role for a page cannot edit the page at all.

Log in to the portal and use one of the following methods to access the portlet:

- Navigate to the page you want to change and select **Edit Page Layout** from the drop-down menu. The drop-down menu for a page is located on the tab for that page. The Edit Layout portlet opens the current page for editing. Other portlets in the Page Customizer are available if you have appropriate access.
- After creating a new page, you are directed to the Edit Layout portlet to edit the layout and content of the new page. Other portlets in the Page Customizer are available if you have appropriate access. To create a new page, select **New Page** from the drop-down menu. The drop-down menu for a page is located on the tab for that page.
- Navigate to the Manage Pages portlet and click the **Edit Page Layout** icon in the appropriate row.

"Locking content on a page"
Use the Locks portlet to set permissions for moving containers or content, or for deleting portlets. This allows you to lock content to certain locations on a page. Any element on the page, such as a row container, a column container, or a portlet, can be locked or unlocked for many variations that give a user control of what can be modified. On a page where content and layout should be preserved, both can be locked, preventing other users from altering the arrangement of containers or portlets on the page.

"Connections between portlets"
The Portlet Wiring Tool allows you to configure connections, or *wires*, between *cooperative portlets*. Cooperative portlets can exchange information, or *properties*, with each other through the property broker. Properties are exchanged either by prompting the user with a Click-to-Action menu or automatically using preconfigured wires. As a result, portlets on the page can react in an integrated and unified manner to the user's actions.

"Themes and skins" on page 1287
A theme determines the global appearance of a page. The purpose of this is to ensure visual consistency. Themes affect the navigational structure, the banner, the colors and fonts, the available portlet skins, and other visual elements of a page. A skin determines the frame that is displayed around a portlet.

Locking content on a page

Use the Locks portlet to set permissions for moving containers or content, or for deleting portlets. This allows you to lock content to certain locations on a page. Any element on the page, such as a row container, a column container, or a portlet, can be locked or unlocked for many variations that give a user control of what can be modified. On a page where content and layout should be preserved, both can be locked, preventing other users from altering the arrangement of containers or portlets on the page.

Before you begin

In order to modify the locks for a certain page, a user must have Editor role on that page. If a user has appropriate access rights, the user can determine what other users are able to do with a page.

About this task

Be aware of the effect that locks and access permissions can have on pages that inherit content from a shared page. To lock or unlock content on a page:

Connections between portlets

The Portlet Wiring Tool allows you to configure connections, or *wires*, between *cooperative portlets*. Cooperative portlets can exchange information, or *properties*,

with each other through the property broker. Properties are exchanged either by prompting the user with a Click-to-Action menu or automatically using preconfigured wires. As a result, portlets on the page can react in an integrated and unified manner to the user's actions.

Setting up a wire between portlets allows the user's transfer selection choices to be saved. When specific interactions are performed, the wire can be used to automatically transfer properties to target portlets without displaying the pop-up menu prompting the user for more information. The Portlet Wiring Tool allows you to view the properties that portlets on the page can send or receive. If a match is available between two portlets, you can create a wire between the two portlets. Existing wires may also be deleted using the tool.

The Portlet Wiring Tool also provides the functionality to implement cross-page portlet communication. Cross-page wires allow portlets to exchange properties across pages. Before creating a cross-page wire, the target page must have receiving actions defined as global on its portlets. Setting an action as global will also make that action available to Click-to-Action menus. To set an action as global, navigate to the target page and select **Edit Page Layout** from the drop-down menu on the title bar. Then select the **Wires** tab and click **Manage Actions**. This will bring up a listing of the portlets on a page and their corresponding receiving actions.

As an alternative to the Portlet Wiring Tool, wires can also be created interactively in the portlet. Depending on the browser, users with sufficient permissions can create wires by holding either the **CTRL** or **ALT** keys and clicking an icon or hotspot in the portlet that is already associated with a Click-to-Action function. A dialog is displayed that allows the user to create a wire to other portlets on the page.

The wiring tool allows wires to be created in situations which are not handled by the interactive approach. For example, the tool does not require the existence of Click-to-Action menus to initiate wire creation It can be used to create multiple wires from a single source property (using the interactive approach, a single source can be wired to a single target or all targets, not an arbitrary subset). Wire creation or deletion is subject to the access control checks as described later.

In order to view the tool itself, users must possess at least "User" role permissions on the page and the portlet. Further access checks are performed before allowing the user to view, create, or delete wires between portlets. The user must possess at least "User" role permissions on a page and the wired portlets on it to be able to view wires for the page. Users may also be able to create or delete personal wires, which affect their view of the page, or create or delete public wires, which affect all users' view of the page. Users must possess at least "Privileged User" role permissions on the page and "User" permissions on the portlets to be able to create or delete personal wires, while at least "Editor" role permissions are required on the page and "User" permissions on the portlets to be able to create or delete public wires.

Themes and skins

A theme determines the global appearance of a page. The purpose of this is to ensure visual consistency. Themes affect the navigational structure, the banner, the colors and fonts, the available portlet skins, and other visual elements of a page. A skin determines the frame that is displayed around a portlet.

Use the Themes and Skins portlet to do the following:

- Set the portal default theme
- Set the portal default skin
- Associate skins with a theme
- Set the default skin for a theme
- Add new themes and skins to the portal
- Delete themes and skins from the portal.

Click the **Administration menu** icon. Then, click **Portal User Interface** > **Themes and Skins**. Refer to the respective portlet helps for more instructions.

Notes:

1. If you remove a theme, the references to that theme and the links between that theme and the related skins are also deleted. If you want to remove the skins that are related to the removed theme as well, apply special care to remove only skins that are related to no other theme than the deleted one. The skins are associated to the portlets. Therefore, if you have a skin that is related to several themes, and you delete one of those themes, then the skin still shows under the other themes.
2. Deleting a theme or skin does not remove the `/theme` or `/skin` directory from the server.
3. Some of the theme and skin titles might not appear correctly if your language preference uses DBCS characters. Correct the display of these titles, change the character set used by your language preference for HTML markup to `UTF-8`.

Related information:

"Changing the character set for a language" on page 1427
The character set is stored in the database. This is the character set used for the response to the user. You can change the character set for a language.

Managing theme capabilities

You can administer the theme module framework. This includes deploying resources that can be cached, configuring capability filters, and disabling the automatic prerequisite loading feature.

"Deploying themes with cacheable resources"
Data sources are used in a portal to serve content. Some resources are cached and other resources can define cache settings and pass them to a data source. In a production environment, use caching. When debugging themes, disable caching.

Deploying themes with cacheable resources

Data sources are used in a portal to serve content. Some resources are cached and other resources can define cache settings and pass them to a data source. In a production environment, use caching. When debugging themes, disable caching.

Caching information for combined requests is computed by the framework based on the individual data sources that are combined. However, in some cases the data sources do not provide caching information, such as in the case of static file inclusion. For example, with the resource data source, content is addressed with `res` in the `plugin.xml` file and the file serving servlet does not set the cache information.

To make resources cacheable, the resource aggregator provides a generic mechanism where it defines the cache settings for a single URI, URI patterns or

contribution types. These settings are then passed to the data source, which determines whether these settings are considered. The resource data source includes them, for example, where the dav data sources for the filestore provides their own settings. These settings are then considered for content in the `plugin.xml` file, which is addressed with `res`. Content addressed with `dav` in the `plugin.xml` file has its own settings and does not use the following parameters.

Caching information is set in the WP ConfigService resource environment provider. The available settings are:

com.ibm.wps.resourceaggregator.cache.info.<id>.type
The values can be set as `uri` or `contributiontype`.

com.ibm.wps.resourceaggregator.cache.info.<id>.re
The value is a regular expression that can be matched against a URI when `uri` is defined as the type, or a contribution type when `contributiontype` is set for the type.

com.ibm.wps.resourceaggregator.cache.info.<id>.max-age
The value is the maximum age of the resource in seconds.

com.ibm.wps.resourceaggregator.cache.info.<id>.cache-scope
The value is `public` when this resource can be cached in an external caching infrastructure or `private` if it cannot.

com.ibm.wps.resourceaggregator.cache.info.<id>.user-context
The value is `false` if this resource is shared between users, and `true` if the value is user specific.

Sample of entries in WP ConfigService:

- `com.ibm.wps.resourceaggregator.cache.info.0.type = "uri"`
- `com.ibm.wps.resourceaggregator.cache.info.0.re = ".*\.(js|css)"`
- `com.ibm.wps.resourceaggregator.cache.info.0.max-age = "86400"`
- `com.ibm.wps.resourceaggregator.cache.info.0.cache-scope = "public"`
- `com.ibm.wps.resourceaggregator.cache.info.0.user-context = "false"`
- `com.ibm.wps.resourceaggregator.cache.info.1.type = "contributiontype"`
- `com.ibm.wps.resourceaggregator.cache.info.1.re = "config_static"`
- `com.ibm.wps.resourceaggregator.cache.info.1.max-age = "100000"`
- `com.ibm.wps.resourceaggregator.cache.info.1.cache-scope = "public"`
- `com.ibm.wps.resourceaggregator.cache.info.1.user-context = "false"`

The following entries are set by default:

- Expiry is set to `1 day`.
- User context is set to `false`.
- Cache-scope is set to `public`.

There are configuration tasks that automatically add or remove these entries from the WP ConfigService resource environment provider. To add these entries, run the **`set-resourceaggregation-cache-info`** configuration task . If you want to remove entries from the WP ConfigService, then run the **`remove-resourceaggregation-cache-info`** configuration task .

Disabling caching on development systems

Usually, the profiles and contributions are determined once, when the portal server is started and then they are used unchanged. Updates to the portal are not included until a server restart occurs, for performance reasons.

To see the changes to profiles and contributions immediately, the `resourceaggregation.development.mode` property to `true` within the WP ConfigService resource environment provider.

Managing community pages

Community associations associate a portal page with a community in IBM Connections.

"Managing community associations"
You can create, view, modify, or delete community associations on a page with the Page Associations window, the XML configuration interface, and Portal Scripting Interface.

"Creating community associations during page template instantiation" on page 1293
When creating a page template, you can require that a community association is created when a page is created from the template.

"Community associations and APIs" on page 1295
Although community associations are typically managed through the portal user interface, you might want to access the associations programmatically.

Managing community associations

You can create, view, modify, or delete community associations on a page with the Page Associations window, the XML configuration interface, and Portal Scripting Interface.

Procedure

To create a community association, complete the following steps, according to the method you want to use.

- Create a community association in the user interface with the Manage Associated Communities window.
  - Open the Manage Associated Communities window from the site toolbar.
    1. In the site toolbar, click the **Page** > **General** > **Associated community**. If an association exists, it is displayed.
    2. Select **Associated community** to display the Manage Associated Communities window.

       Note: The **Community** section is displayed only if an IBM Connections server is configured for the portal.
  - Open the Associations window from the page properties portlet.
    1. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
    2. Go to the page, and click the **Edit Page Properties** icon for the page.
    3. In the **Advanced options** section, click **I want to edit associations**.
    4. Click the **Community** tab. If an association exists, it is displayed.

Note: The **Community** tab is displayed only if an Connections server is configured.

1. Create the association. You can create an association with one of the following methods:
   - Click **Select Community**. The portal queries the Connections server and lists available communities. Select the community that you want to associate with the page.

     Depending on the number of available communities, only a subset of all communities might be listed. If the Connections server is not running, no communities are listed.

     You can filter the list of communities by entering text in the **Find community** field. The list of communities is narrowed to only those communities that contain the search text in the community title.
   - To associate the page with the same community that is associated with the parent page, click **Use default community from parent page**.
2. Optional: To automatically grant page access to members of the associated community, click **Limit access to this page to only community members**.

   Activating this feature results in the following changes:
   - The User role is assigned to the virtual user group that represents the community and the page. This access is in addition to any access that you explicitly grant to the page.
   - Role blocks are defined for the page for the User role and the Privileged User role. These role blocks prevent corresponding access privileges that are granted to the parent page from being propagated to this page.

   This setting is displayed only if you installed and enabled the IBM Connections adapter for VMM users and groups. For details, see *Automatically grant page access to community members*.

   Access control changes from this setting:

   When you select the **Limit access to this page to only community members** setting, the following changes are made:
   - The User role on the page is assigned to the virtual user group that represents the members of the associated community.
   - Role blocks are added on the page for the User role and the Privileged User role.

   When you clear the **Limit access to this page to only community members** setting, the following changes are made:
   - The User role on the page is removed for the virtual user group that represents the members of the associated community.
   - The role blocks on the page for the User role and Privileged User role are removed.

   If you change the community that is associated with the page, the following changes are made:
   - The User role on the page is removed for the virtual user group that represents the members of the previously associated community.
   - The User role on the page is assigned to the virtual user group that represents the members of the newly associated community.
   - If role blocks do not exist on the page for the User role and Privileged User role, the role blocks are created.

Activating this feature is restricted to users that are granted all roles that are required if the corresponding access control modifications are performed manually with the access control administration portlets. For more information about the roles that are required when you select **Limit access to this page to only community members**, see *Access permissions*.

3. Optional: If you want child pages of the page to be automatically associated with the same community as this page, select **Copy updated association to all *number* direct child pages**.

   This setting associates the community to all child pages where the user has sufficient access to update page associations. If the user does not have the required access for a page, it is not updated.

   Depending on the configuration of the server, the window displays the number of pages and nesting levels that are affected based on a threshold. If the number of affected pages exceeds the configured threshold value, this option is not displayed. For details on setting the page thresholds, see *Configuring support for community pages*.

   Note: When community associations for a page are copied to child pages, only the community associations are copied. The value of the **Limit access to this page to only community members** setting is not copied to child pages because of access control inheritance. This inheritance automatically enables members of the mapped community to view the child pages.

- Create a community association by using the XML configuration interface (**xmlaccess** command).

  When you are defining the association in the XML import file, use the <content-mapping-info> element, and specify a content mapping scope of `ibm.connections` for an individual nested <content-mapping> element.

  For additional information about the XML configuration interface and <content-mapping-info> elements, see *XML configuration interface and content associations*. This XML sample shows how to map a page to a community. In this example, the page has the unique name `unique-name-of-the-page-to-be-updated`, and the community is specified with the community UUID `some-ibm-connections-community-uuid` in Connections.

```
<request type="update" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd">
    <portal action="locate">
        <content-node action="update"  domain="rel" uniquename="unique-name-of-the-page-to-be-upda
            <content-mapping-info>
                <content-mapping content-id="some-ibm-connections-community-uuid"
                                 default="true" scope="ibm.connections"/>
            </content-mapping-info>
        </content-node>
    </portal>
</request>
```

Related concepts:

"Access permissions" on page 1537
Learn about sensitive operations for resources and the roles that are required to perform those operations. Sensitive operations include common tasks such as viewing portlets on specific pages and complex, high-risk tasks like running XML configuration interface scripts.

Related tasks:

"Automatically grant page access to community members" on page 759
If you want community members to automatically be able to access the page, without explicitly configuring access, you must enable that feature. Community membership must be integrated with portal security before you can enable this

feature.

Related reference:

"XML configuration interface and content associations" on page 2076
With the XML configuration interface (**xmlaccess** command), you can perform batch updates of content associations or export associations to import into another portal. Content association information is represented in the XML configuration schema by content-mapping-info elements.

Related information:

 REST API and content associations
If you are creating or extending an application and want to manage content associations with that application, you can use portal remote APIs. These APIs retrieve a list of content associations and then create, update, or delete associations.

Creating community associations during page template instantiation

When creating a page template, you can require that a community association is created when a page is created from the template.

About this task

When creating the community association, you have several options:

- You can associate the new page with a community that is automatically created during page template instantiation.
- You can associate the new page with a community that is associated with the parent page of the new page.
- You can associate the new page with a specific community that already exists.

Important: To create communities automatically when creating pages from a template, the user creating the page must be authorized on the Connections server to create communities.

"Creating communities during community page instantiation"
When you create a template for a community page, you can configure the template to create a community when you create a page from the template.

"Creating associations to existing communities during page instantiation" on page 1294
When you create a template for a community page, you can configure the template to associate the page created from the template with an existing community. In this case, the community is not modified in any way.

Related tasks:

"Managing community associations" on page 1290
You can create, view, modify, or delete community associations on a page with the Page Associations window, the XML configuration interface, and Portal Scripting Interface.

Creating communities during community page instantiation

When you create a template for a community page, you can configure the template to create a community when you create a page from the template.

About this task

In this case, when the page is created, a community is created in IBM Connections, and the new page is associated with the new community. The name of the new community is derived from the title of the new page.

Important: To create communities automatically when creating pages from a template, the user creating the page must be authorized on the Connections server to create communities.

Procedure

There are two ways to specify that a page template requires a community association.

- Add a community association to the page template by editing the page associations in the user interface or by using the XML configuration interface. See *Managing community associations* for details.

  The new community has the following characteristics:
  - The same title as the new page.
  - The same community description as the community that is associated with the page template.
  - The same **Access** setting as the community that is associated with the page template.
  - The user who creates the page is designated as the owner of the new community.
- Define parameters on the template page.
  1. Set the **ibm.portal.instantiation.community.create.new** parameter on the template page with a value of `true`.

     You can set this parameter by editing the page properties in the user interface or by using the XML configuration interface (**xmlaccess** command).

     The new community has the following characteristics:
     - The same title as the new page.
     - An empty community description.
     - An **Access** setting of `public`.
     - The user who creates the page is designated as the owner of the new community.
  2. Optionally, you can specify the **Access** setting for the community by specifying the **ibm.portal.instantiation.community.access.type** parameter on the template page.

     You can specify one of the following values: `public`, `restricted`, `moderated`.

Creating associations to existing communities during page instantiation

When you create a template for a community page, you can configure the template to associate the page created from the template with an existing community. In this case, the community is not modified in any way.

About this task

Procedure

Associate the new page with a specific community or with the community that is associated with the parent page of the new page.
Define the association on the template page by setting page parameters in one of the following ways:

- Edit the page properties in the user interface.
- Use the XML configuration interface (**xmlaccess** command).
- To associate page instances that are created from this template with a specific community, set the **ibm.portal.instantiation.community.ref** parameter.

  The value of the parameter is the universally unique ID (UUID) of the target community. The UUID of the community is identified by the **communityUuid** parameter in the URL for the community in IBM Connections.

  For example, in the following URL:

```
https://www.example.com:port_number/communities/service/html/communityview?communityUuid=e9f88a24-3dec-446a-91f3-90118d7e22c3
```

  The community UUID is `e9f88a24-3dec-446a-91f3-90118d7e22c3`.

  Note: You cannot use the **ibm.portal.instantiation.community.ref** parameter in combination with the **ibm.portal.instantiation.community.create.new** parameter. If both parameters are specified for a page template, the **ibm.portal.instantiation.community.ref** parameter takes precedence, and the **ibm.portal.instantiation.community.create.new** parameter is ignored.
- To associate page instances that are created from this template with the community that is associated with the parent page of the new page, set the **ibm.portal.instantiation.community.parent** parameter. Specify a value of `true` for the parameter.

  If there is no community association for the parent page, the new page is created without a community association.

  However, if you want to ensure that a community is created in this case, set the **ibm.portal.instantiation.community.create.new** parameter with a value of `true`. If the parent page has no community association, a new community is created with the same characteristics as described in *Creating communities during community page instantiation.*

Related tasks:

"Creating communities during community page instantiation" on page 1293
When you create a template for a community page, you can configure the template to create a community when you create a page from the template.

Community associations and APIs

Although community associations are typically managed through the portal user interface, you might want to access the associations programmatically.

For example, a portlet on a specific page might need to communicate with a community in IBM Connections that is associated with the page. For this type of access, the content mapping service supports several APIs:

- Java API
- REST API
- Portal Scripting Interface
- XML configuration interface (**xmlaccess** command)

The content mapping service provides general access to associations of portal resources with other types of resources. Associations to resources of different types are grouped together and separated from one another by *scopes*. Community associations are managed in the `ibm.connections` scope.

Related reference:

"XML configuration interface and content associations" on page 2076
With the XML configuration interface (**`xmlaccess`** command), you can perform batch updates of content associations or export associations to import into another portal. Content association information is represented in the XML configuration schema by content-mapping-info elements.

Related information:

Java API for Content Mapping Service

Portal Scripting Interface and content associations
With the Portal Scripting Interface, you can create scripts to automate the management of content associations. Using the ContentMapping bean with the Portal Scripting Interface, you can add, modify, and remove content associations.

Managing your site

Earlier versions of WebSphere Portal Express provided the Resource Manager portlet for performing site management. With Version 8.0 of WebSphere Portal Express and Web Content Manager, this site management functionality has been replaced by the new functionality for managing pages.

About this task

Managed pages are portal pages that are managed in Web Content Manager. You can manage your sites and move portal pages from one server to another by using the Web Content Manager syndication mechanism. This has the following advantages over the previous site management feature:

- You can create drafts of portal pages in the context of a project. You can preview these drafts, including possible updates by other coworkers on the same project.
- You can have staggered versions of pages.
- You can define a workflow for the pages. The workflow can include security settings for the pages. For example, you can remove edit rights from a page after the page is approved and published.
- You can submit draft pages for approval. You can also have the project approved for publication.
- You can preview the project as if it was published content.
- You can publish a project or individual pages to the subscriber server by one of the following mechanisms:
  - manually
  - based on date and time
  - automatically, when all items of a project have been approved and are pending for publication.

The only option to use the Resource Manager portlet and the site management functionality with WebSphere Portal Express Version 8.5 is to use a WebSphere Portal Express Version 7.0 and its Resource Manager portlet. For information about site management with the Resource Manager portlet, see the *WebSphere Portal Express Version 7 product documentation: Managing your site*.

Tagging and rating

Get an overview of the administrative tasks related to tagging and rating.

Attention: To use tagging and rating with your virtual portal, ensure that the **web resource v7.0** and **web content templates 3.0** libraries exist on the virtual portal.

Portal administrators can do the following tasks:

- Enable or disable tagging and rating for portal users. Tagging and rating are enabled in the IBM WebSphere Portal Express Portal 8.5 theme. You can disable and enable tagging and rating globally for the portal as a whole. To do so, use the following configuration properties:
  - To disable and enable tagging portal wide, use the property `com.ibm.wps.cp.tagging.isTaggingEnabled`.
  - To disable and enable rating portal wide, use the property `com.ibm.wps.cp.rating.isRatingEnabled`.

  these properties are available in the WP CP configuration service Resource Environment Provider in the WebSphere Integrated Solutions Console. For details about portal service configuration properties and how to set them refer to the topics about Portal configuration services and Setting service configuration properties.
- Add custom content that users can tag and rate, for example, pages, portlets, content that is managed by Web Content Manager, or custom content such as books in a bookstore.

  Note: If you want your users to tag and rate custom content, you must write code. The code must allow customers to find this content with the public APIs. You must also add the resources that you want your users to tag to the portal. Custom content is anything apart from portlets, portal pages, and Web Content Manager resources. For information, read *Enabling your own custom content to be tagged and rated*.
- Assign access rights to users for tagging and rating content. For details, read *Security for tagging and rating*.
- Administer the tag clouds. Administrators can set multiple parameters that affect the tag cloud. Two examples are described here in detail. For a list of the possible tag cloud parameters, read *The tagging and rating user interface* and the tagging and rating reference topics.
  - Scope tag clouds to the kind of resources for which you want tags to be displayed. A default tag cloud displays all tags that users applied in the portal. Administrators can change this behavior. For example, you can limit a tag cloud to display tags that were applied only to pages, portlets, or books. You scope tag clouds by setting either categories or type schemas for them. The tag cloud represents only portal content that matches the categories or type schemas that you specified. You can scope each single tag cloud instance differently. Scoping tag clouds also affects the result lists of resources that are shown when users click a tag. The result list shows only resources of the scoped type. For example, if a tag cloud is scoped to the type photo, the result list shows only photo images.
  - Configure the tag cloud behavior. Tag clouds can react differently to tag clicks:
    - Either the user is redirected to the portal Tag Center and a result list of resources is displayed to which this tag was applied is displayed.

- Or the tag cloud shows the clicked tags as shared render parameters, so that, for example, other portlets on the same page can display corresponding content.

- Stage a tagspace to a different portal system, for example, when you move your portal to a staging server or upgrading your portal to a new version. For details about how to do so, read *Using the XML configuration interface to administer tags and ratings*.
- Obtain statistics about the tagging and rating behavior of the portal users.
- Administrators can also do all tasks that portal users can do as described in the topics about portal user tasks.

  "Introduction to tagging and rating" on page 1299
  Tagging and rating are features that support collaboration and interaction between users when using Web content.

  "What is new in tagging and rating" on page 1301
  With IBM WebSphere Portal Express Version 8.5, the tag and rating widgets have been replaced by new enhanced versions of these widgets.

  "How tagging and rating works in the portal" on page 1303
  Use these topics for general administrative information about tagging and rating in the portal.

  "The tagging and rating user interface" on page 1309
  Learn about the user interface that IBM WebSphere Portal Express provides for users to work with tags and rating.

  "Tagging and rating for static pages" on page 1324
  Tagging or rating for static pages works only with the dialog widgets of earlier portal versions.

  "Enabling your own custom content for tagging and rating" on page 1324
  Enabling your own custom content for tagging and rating works only with the dialog widgets of earlier portal versions.

  "Federating tags" on page 1324
  WebSphere Portal allows the federation of remote tagging systems, such as Connections.

  "Configuration reference for tagging and rating" on page 1330
  Developers can customize the tagging and rating user interfaces. The parameter reference lists and explains all available configuration parameters and their values. The topic about CSS provides information about how to change the design of the user interfaces.

  "Security for tagging and rating" on page 1342
  For administering which users can tag and rate content, the portal provides virtual resources for tagging and rating and roles on these virtual resources.

  "Using the XML configuration interface to administer tags and ratings" on page 1343
  You can use the XML configuration interface to manage tagging and rating in the portal. For example, you can move tagspaces and ratings between portal versions or for staging purposes.

  "Hints and tips for tagging and rating" on page 1346
  Learn about some hints and tips that apply to tagging and rating. Some hints and tips might help developers and portal administrators, others might help portal users.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

"Blogs" on page 1999
Use blogs and blog libraries to provide news and commentary on a variety of subjects pertinent to your intranet and extranet sites. Blogs and blog libraries typically combine text with graphics and links to other blogs and web sites. Entries that you create and post are arranged in reverse-chronological order, with the newest entry displayed first. Readers can post comments about your entries, fostering discussions and online networking. You can manage your own blog entries and comment on other blog entries. You can also incorporate tagging and rating as you would with other WebSphere Portal content.

Related reference:

"CP Configuration Service for tagging and rating" on page 312
The CP Configuration Service provides the properties for tagging and rating.

"Enabling your own custom content for tagging and rating" on page 1324
Enabling your own custom content for tagging and rating works only with the dialog widgets of earlier portal versions.

"Security for tagging and rating" on page 1342
For administering which users can tag and rate content, the portal provides virtual resources for tagging and rating and roles on these virtual resources.

"Using the XML configuration interface to administer tags and ratings" on page 1343
You can use the XML configuration interface to manage tagging and rating in the portal. For example, you can move tagspaces and ratings between portal versions or for staging purposes.

Introduction to tagging and rating

Tagging and rating are features that support collaboration and interaction between users when using Web content.

Rather than relying on dedicated authors to be responsible for adding content, knowledge-sharing communities of users now collectively contribute to Web content. These systems grow quickly and are not as coordinated as they were when teams of dedicated authors were responsible for planning and creating content. For this reason, new methods of organizing and structuring content are required.

Tagging and rating are common and effective collaboration techniques that support the collective wisdom of communities:

Tagging

Tagging means assigning keywords as metadata to resources to describe or evaluate the content of these resources. These keywords are tags. For example, a user can apply the tag `websphere` to a page that provides information about IBM WebSphere products. Tagging content provides users the ability to describe, and better categorize, resources. By tagging content, users make it easier to find and highlight the importance or quality of content, which, in turn, benefits the other users in the knowledge-sharing community. Tagging provides better search results, as the search criteria is descriptive and users do not have to scan the content for the keywords.

A user typically assigns more than just one tag to a single resource. For example, a user can assign a book the tags `portal`, `web20`, and `computer_science` . This allows for multiple ways of finding the book, like putting multiple copies of the same book on different shelves.

Tagging has become one of the most popular techniques to allow users and entire user communities to classify, organize, and structure content autonomously. By tagging content users add valuable meta information and even lightweight semantics to Web content. Tagging allows non-expert users to develop folksonomies that categorize content available in the system.

Tag
A tag is a keyword that users apply to describe or evaluate the content of the resource that they tag. A user can apply the same private tag to multiple resources. This means that an individual user can use the same certain tag more than once, but only once per resource. By default users can only delete tags that they have applied themselves. Users with the MANAGER role have more rights.

Tag cloud
Tag clouds help users find and retrieve resources by showing aggregated views of tags that users have applied to specific resources. Tag clouds provide different visual representations of all the tags that users have applied. For example, depending on the configuration settings, tag clouds can display all available tags or how often specific tags occur. Tag clouds usually highlight the popularity or importance of specific tags by increasing the size of the tags or changing the color of the tags.

Tagspace
A tagspace represents the number of occurrences of a certain tag. It contains the name of the tag, the number of occurrences and the information whether the current user has applied this tag as a private tag. To represent the tag cloud, the portal loads the tagspaces rather than retrieving the single tag instances from the database and accumulating the numbers.

Rating
Rating is the evaluation or assessment of something, in terms of quality, for example, (as with a critic rating a novel), quantity (as with an athlete being rated by his or her statistics), or a combination of both. Examples of quality ratings are critics who rate books or customers who rate purchased products for quality. Examples of quantitative ratings can be the performance data of cars or computers. Users rate resources by assigning numeric or other values to resources to indicate how much they like the resource. Thus, a rating is a value associated to a resource. Ratings are selected from a range of possible values where one end of the range usually refers to "like" and the other end to "dislike".

Rating allows users and entire communities to express which content they like or dislike.

By default users can only delete ratings that they have applied themselves. Users with the MANAGER role have more rights.

Resource
A user can apply the same rating to multiple resources. A resource is

anything that can be uniquely identified and addressed in a portal. For example, this can be a portal page, or portlet, or an item for sale in an internet shop.

Public and private tagging and rating

Users can assign public or private tags and ratings:

Private tagging and rating
With private tagging and rating users can see only their own tags and ratings. This allows users to browse through their Web content and retrieve important resources fast. A user typically assigns more than just one tag to a single resource. For example, a user can assign a book the tags `portal`, `web20`, and `computer_science`. This allows for multiple ways of finding the book, like putting multiple copies of the same book on different shelves.

Public tagging and rating
In a collaborative context users can see tags and ratings given by their collaborative community. First, users assign tags to resources just like under private tagging, creating their own way to browse through system content. Public tagging becomes powerful when multiple users start to tag resources and tags become part of the community pool of tags. This way tagging creates a bridge between personal and community knowledge and enables collective intelligence. Everyone's tags together form a kind of community consensus about an resource. The tags that are used most often might be the ones describing an item best.

Community and personal tags and ratings

Tags can be personal tags or community tags. All statements made for tags in the following apply also to ratings.

Personal tags
Personal tags are applied by a specific user. When a user tags a resource, the user applies the tag to the resource as a personal tag. Public tags can be seen by everyone who belongs to the community. Private tags can only be seen by the user who created the tag.

Community tags
Community tags are assigned by the community, that is by all other users in the system. Every public tag that a user assigns turns into a community tag, because it is visible to the community.

What is new in tagging and rating

With IBM WebSphere Portal Express Version 8.5, the tag and rating widgets have been replaced by new enhanced versions of these widgets.

Before IBM WebSphere Portal Express Version 8.5, tagging and rating provided two user interfaces with two different types of widgets:

- The default portal tagging and rating user interface featured several elements: dialog widgets for tagging and rating, the tag cloud, the Tag Center, and a feature for browsing tags. The dialog tag and rating widgets opened in separate pop-up windows.
- The alternative user interface provided inline widgets for tagging and rating. If you added such an inline widget to a portal resource, this widget was integrated into the portal resource. The user viewed the user interface for tagging or rating as part of the portal resource.

With IBM WebSphere Portal Express Version 8.5, both the dialog and inline tag and rating widgets of previous portal versions are deprecated. They are replaced by a

single pair of new enhanced inline widgets. They provide the functionality of both types of the old widgets. Users view them as part of the portal resource.

The new inline widgets have the following advantages:

- The new widgets combine the functionality of the dialog widgets and the inline widgets of earlier portal versions.
- With the new widgets, users can create, modify, and delete tags and ratings that they have assigned themselves. With the earlier inline widgets, user could only view tags and ratings. s
- The widgets provided by WebSphere Portal Express Version 8.0 and earlier versions required Dojo support. The new widgets in WebSphere Portal Express Version 8.5 are light-weight, as they work without Dojo.
- The new widgets provide easy and convenient access to various actions, for example switching between different scopes and viewing rating distribution.

For more detailed information about the new tag and rating widgets, read *The tag and rating widgets*.

To use the new widgets after a portal upgrade to version 8.5, you need to enable the widgets by running a configuration task. For more information, read *Enabling the new tag and rating widgets after a portal upgrade*.

In a default WebSphere Portal Express Version 8.5 installation, the new tag and rating widgets are enabled for specific profiles. You can enable and disable the new widgets for additional profiles. For more detailed information, read *Enabling and disabling the tag and rating widgets for additional profiles*.

The Dojo tagging and rating menu options of the earlier default widgets are still available in WebSphere Portal Express Version 8.5. They are enabled for specific profiles. You can enable and disable them for additional profiles. For more detailed information, read *Enabling and disabling the Dojo tagging and rating options for additional profiles*.

Related tasks:

"Enabling and disabling the tag and rating widgets for additional profiles" on page 1340
In a portal installation, the tag and rating widgets are available for specific portal profiles. You can enable the widgets for more other profiles by adding them to the profile.

"Enabling and disabling the Dojo tagging and rating options for additional profiles" on page 1341
In a portal installation, the Dojo tagging and rating menu options for portal pages and portlets are available for a specific portal profile. You can enable these options for other profiles by adding them to the profile.

"Enabling the new tag and rating widgets after a portal upgrade" on page 917
If you upgrade your IBM WebSphere Portal Express from an earlier version to Version 8.5 and want to use the new tag and rating widgets, you must first enable blogs and wikis. Then, complete the following task to ensure that tag and rating widgets are enabled.

Related reference:

"The tag and rating widgets" on page 1310
The portal provides one widget each for tags and for ratings.

How tagging and rating works in the portal

Use these topics for general administrative information about tagging and rating in the portal.

"How public and private tags and ratings work in the portal"
Users can choose between applying a tag or rating as a private or public tag or rating.

"Tagging and rating with anonymous users" on page 1304
Anonymous users of the portal can tag and rate portal content, if an administrator adds the anonymous user to the appropriate roles on the virtual resources as described in the topic about Security for tagging and rating.

"Grouping tags and ratings via resource categorization" on page 1304
Users apply tags and ratings to resources. Users can tag and rate all resources that can be uniquely identified.

"Normalizing tags" on page 1305
The portal provides several options for normalizing tags. Normalization is a process of transforming a text fragment, such as a tag, into another, more generic representation. This bundles different spellings or grammatical versions of the same lexical word that users might use as tags, for example `color, Color, COLOR, colour, colors, colored`.

"Type-ahead feature for the deprecated tag widget" on page 1306
The tag widget from earlier IBM WebSphere Portal Express versions provided a type-ahead feature. With portal V 8.5, that tag widget is deprecated. The type-ahead feature makes it easier for users to find suitable tags. Type-ahead supports users when they work with tags. For example, when users apply tags using the tag widget, or when they search for tags, for example by using the open search functionality, type-ahead provides users suggestions for tags that other users have applied already before. Type-ahead can also help reduce the number of variants of tags.

"Filtering content for tagging" on page 1307
You can use filtering mechanisms to control which terms users can use and which terms they cannot use as tags. The portal provides both a blacklist and a whitelist filter.

How public and private tags and ratings work in the portal

Users can choose between applying a tag or rating as a private or public tag or rating.

The differences between private and public tags and ratings are described in the following. All statements made for tags in the following are also applicable to ratings.

Private tags

- A private tag can be seen only by the user who applied it
- A user can apply the same private tag only once to one specific resource.
- A user can apply the same private tag to multiple resources. This means that an individual user can use the same tag more than once, but only once per resource.
- Users can only delete private tags that they have applied themselves. Users with a Manager role have more rights. For more information see the topic about *Security for tagging and rating*.

Public tags

- All public tags and ratings are visible to all users who belong to the community and who have access to the resource itself, independent of who created the tag or rating.
- A user can apply the same public tag only once to one specific resource.
- A user can apply the same public tag to multiple resources. This means that an individual user can use the same tag more than once, but only once per resource.
- The same public tag can be applied to the same specific resource by multiple users. In other words, all users can apply the same tag or rating to a resource once; as a result, the resource can have the same tag applied to it multiple times.
- Users can only delete public tags that they have applied themselves. Users with a Manager role have more rights. For more information see the topic about *Security for tagging and rating*.
- Public tags are also community tags.

Related reference:

"Security for tagging and rating" on page 1342
For administering which users can tag and rate content, the portal provides virtual resources for tagging and rating and roles on these virtual resources.

Tagging and rating with anonymous users

Anonymous users of the portal can tag and rate portal content, if an administrator adds the anonymous user to the appropriate roles on the virtual resources as described in the topic about Security for tagging and rating.

As the portal cannot distinguish one anonymous user from another, there are some side effects that need to be understood:

- For tagging, the same user can normally not apply the same tag to the same resource more than once, but an anonymous user can apply the same tag more than once.
- For rating, the same user can normally rate a single resource only once, but an anonymous user can rate the same resource more than once.
- Anonymous users cannot view their personal tags, and they cannot delete tags or ratings.
- Anonymous users cannot tag or rate a page with the Search and Tag Center profile deployed on it.

Grouping tags and ratings via resource categorization

Users apply tags and ratings to resources. Users can tag and rate all resources that can be uniquely identified.

Examples:

- A URI that consists of a type schema and an SSP (scheme specific part), such as an article bar code: `item:item_bar_code_number`
- The ISBN number for a book: `book:isbn_number`.

These resources must be registered with the tagging and rating engine. The portal performs this registration whenever an existing resource is tagged or rated. To categorize tags and ratings, administrators can group resources. You can group resources by URI or by category specification:

Resource categorization per URI

When the portal registers resources, it assigns them a URI. The URI consists of a type schema and a scheme specific part (SSP). For example, users could register media as follows:

- some books under `content/books/action:/isbn`, where `content/books/action` is the type schema and `isbn` is the SSP.
- some other books under `content/books/thriller:/isbn`
- some DVDs under `content/dvds/action:/some_ID`
- some other DVDs under `content/dvds/romance:/some_ID`

Categorization of resources becomes important when tags or ratings are aggregated. For example, tags are aggregated in tag clouds. For details refer to the topic about Configuring the tag cloud. For example if a user wants to aggregate a tag cloud that shows all tags that are associated with the action books as previously mentioned, you could aggregate tags for resources that have been registered under the type schema `content/books/action`. For details about how to scope tag clouds for certain type schemas refer to the topic about Configuring the tag cloud.

Resource categorization per category specification

When resources are registered, they can be explicitly associated with one or more categories. A category is represented by a non-localized unique string based identifier. For example, users could register media as follows:

- some books under `content/books/action:/isbn` and the category `promotion`
- some other books under `content/books/thriller:/isbn` and the category `bestseller`
- some DVDs under `content/dvds/action:/some_ID` and the category `promotion`
- some other DVDs under `content/dvds/romance:/some_ID` and the category `bestseller`

For example, if you want to aggregate a tag cloud that shows all tags for promotion articles, you could aggregate tags for resources that have been associated with the category `promotion`, in the previous example action books and action DVDs. For details about how to scope tag clouds for certain type schemas refer to the topic about Configuring the tag cloud.

Normalizing tags

The portal provides several options for normalizing tags. Normalization is a process of transforming a text fragment, such as a tag, into another, more generic representation. This bundles different spellings or grammatical versions of the same lexical word that users might use as tags, for example `color`, `Color`, `COLOR`, `colour`, `colors`, `colored`.

The standard normalization algorithm provided by the portal removes all diacritical marks from all letters of words. This documentation refers to the diacritic free morphological variation of a word as the normalized form of the word.

This is important when you consider which tags you want to be aggregated together when they are displayed as part of a tag widget, or, even more important, part of a tag cloud.

You can control the normalization behavior by various configuration parameters. To configure normalization behavior, go to **Resources** > **Resource Environment** >

Resource Environment > **Providers** > **WP CPConfigurationService** > **Custom Properties**. You configure normalization by using the following parameter in the portal CP Configuration Service resource environment provider in the WebSphere Integrated Solutions Console:

normalization.displayNormalizedName = (false)
This parameter defines whether the normalized tag names are displayed (exposed) in the tag cloud. The default value for this parameter is `false`.

```
com.ibm.wps.cp.tagging.normalization.displayNormalizedNames = false
```

normalization.typeAhead = (normalized)
This parameter defines the tag names that the type ahead for tag creation mechanism displays to the user. Valid values are `normalized` and `unnormalized`. The default value for this parameter is `normalized`.

Note: The type-ahead feature works only with the dialog tag widget of the default tagging user interface of portal versions earlier than V 8.5. With WebSphere Portal Express V 8.5, the tag and rating widgets of earlier portal versions are deprecated.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Related reference:

"CP Configuration Service for tagging and rating" on page 312
The CP Configuration Service provides the properties for tagging and rating.

Type-ahead feature for the deprecated tag widget

The tag widget from earlier IBM WebSphere Portal Express versions provided a type-ahead feature. With portal V 8.5, that tag widget is deprecated. The type-ahead feature makes it easier for users to find suitable tags. Type-ahead supports users when they work with tags. For example, when users apply tags using the tag widget, or when they search for tags, for example by using the open search functionality, type-ahead provides users suggestions for tags that other users have applied already before. Type-ahead can also help reduce the number of variants of tags.

Note: The type-ahead feature works only with the dialog tag widget of the default tagging user interface of portal versions earlier than V 8.5. With WebSphere Portal Express V 8.5, the tag and rating widgets of earlier portal versions are deprecated. As the user starts typing tag text, one or more possible matches for that text fragment are found and immediately shown to the user. This immediate feedback allows users to select one of the listed options rather having to type the entire word or phrase they were looking for. User can also choose a closely related option from the presented list. Thus, the type-ahead feature allows users to explore the tag space as they type. This can make it easier to find the correct term they want to use as the tag.

Another advantage of the type-ahead feature is that it reduces tag space littering; different users use different spellings for certain terms. An example is the term Web 2.0. Users might spell it as `Web 2.0`, `Web2.0`, `Web 20`, `Web20`, or `Web2` . Semantically all these morphological variations refer to the same term. Therefore it would be inconvenient to present all these variations separately in a tag cloud. If most users have already entered the term as Web 2.0 and new users want to tag

something with a variation of this term and start typing, the type-ahead feature would suggest `Web 2.0`, and most users would probably select this term form the list. This reduces the amount of variation.

The type-ahead feature starts suggesting tags after a user types three characters in a type-ahead enabled input field, for example in the tag widget.

Related concepts:

"Searching for tagged content" on page 610
Users can search the tag space by using the browser search box.

Related reference:

"Type-ahead with the deprecated tag widget" on page 3187
The tag widget from earlier IBM WebSphere Portal Express versions provided a query for the type-ahead feature. With portal V 8.5, that tag widget is deprecated. The type-ahead feature makes it easier for users to find suitable tags. Type-ahead supports users when they work with tags. For example, when users apply tags using the tag widget, or when they search for tags, for example by using the open search functionality, type-ahead provides users suggestions for tags that other users have applied already before. Type-ahead can also help reduce the number of variants of tags.

Filtering content for tagging

You can use filtering mechanisms to control which terms users can use and which terms they cannot use as tags. The portal provides both a blacklist and a whitelist filter.

You enable and disable both filters in the WebSphere Integrated Solutions Console under the **Resource Environment Provider** > **CPConfigService** for tagging and rating. For details see the topics about *Setting service configuration properties* and the *CP Configuration Service for tagging and rating*.

Blacklist filter

The blacklist filter allows you to block selected terms from being used as tags, for example terms that could be perceived as offensive. If you enable the blacklist filter, the portal checks every term that users type as a tag before it is eventually applied and stored. If a user types a term listed on the blacklist, the portal blocks this tag and responds with a message. You can determine the terms that you want to be on the blacklist by using the XML configuration interface.

To enable the blacklist filter, the following configuration property in the CP Configuration Service to true:

```
com.ibm.wps.cp.filter.tagging.blacklist = true
```

Whitelist filter

The whitelist filter allows you to limit the set of tags that users can apply. If you enable the whitelist filter, the portal checks every term that users type as a tag against the whitelist before it eventually applies and stores it. If a user types a term that is **not** listed on the whitelist, the portal blocks this tag and responds with a message. You can set the terms that you want to be on the whitelist by using the XML configuration interface.

To enable the whitelist filter, add at least one entry to the whitelist filter database and set the following configuration property in the CP Configuration Service to `true`:

```
com.ibm.wps.cp.filter.tagging.whitelist = true
```

Note: The portal applies the filter only if you activate it **and** add at least one entry to the whitelist filter database.

You can configure both filter lists by using the XML configuration interface. The following examples show how to add or remove terms to the blacklist. To work with the whitelist, adapt the examples by changing the ID of the filter from `DefaultBlacklistFilter` to `DefaultWhitelist Filter` . Sample scripts for the XML configuration interface are located under the directory *`PortalServer_root`*`/doc/xml-samples` .

Example: Creating new words to the blacklist filter:

```
<request
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
    type="update" create-oids="true">
    <!-- sample for creating a filter with some sample entries -->
    <portal action="locate">
        <filter-instance action="update" id="DefaultBlacklistFilter">
            <filter-data value="badword_1" action="update"/>
            <filter-data value="badword_2" action="update"/>
        </filter-instance>
    </portal>
</request>
```

This sample snippet registers the two words `badword_1` and `badword_2` with the blacklist filter for all locales. Note that you must specify `DefaultBlacklistFilter` for the attribute `id` of the tag filter. You can optionally specify the attribute `locale` for the tag filter data. After you run this XML script, users will not be able to use `badword_1` or `badword_2` as tags.

Example: Creating or deleting individual words:

```
<request
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
    type="update" create-oids="true">
    <!-- sample for creating a filter with some sample entries -->
    <portal action="locate">
        <filter-instance action="update" id="DefaultBlacklistFilter">
            <filter-data value="badword_1" update="delete"/>
            <filter-data value="badword_3" update="update"/>
        </filter-instance
    </portal>
</request>
```

This sample snippet removes the term `badword_1` and adds `badword_3` from the blacklist.

Example: Deleting all terms from the blacklist filter:

```
<request
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
    type="update" create-oids="true">
    <!-- sample for deleting a whole filter -->
    <portal action="locate">
        <filter-instance action="delete" id="DefaultBlacklistFilter">
        </filter-instance>
    </portal>
</request>
```

This sample snippet deletes all terms from the blacklist.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Related reference:

"CP Configuration Service for tagging and rating" on page 312
The CP Configuration Service provides the properties for tagging and rating.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

The tagging and rating user interface

Learn about the user interface that IBM WebSphere Portal Express provides for users to work with tags and rating.

Portal tagging and rating offers the following user interface elements that users can use to work with tags and ratings:

- The tag and rating widgets:
  - The tag widget. This is an inline widget that you can add to your portal content and that users use to tag that portal content.
  - The rating widget. This is an inline widget that you can add to your portal content and that users use to rate that portal content.
- The tag cloud.
- The Tag Center.
- A feature for browsing tags.

Note: With IBM WebSphere Portal Express Version 8.5, both the dialog and inline tag and rating widgets of previous portal versions are deprecated. They are replaced by a single pair of new enhanced inline widgets. They provide the functionality of both types of the old widgets. Users view them as part of the portal resource.

You can configure the widgets by using a set of parameters. You can set these parameters globally for all tag or rating widgets in your portal in the WP CP configuration service. You can also set them for individual instances of the tag and rating widgets by using JavaScript parameters. All these parameters are listed and described in the topic about the WP CP configuration service and the parameter reference topics for tagging and rating.

"The tag and rating widgets" on page 1310
The portal provides one widget each for tags and for ratings.

"The portal Tag Center" on page 1320
The Tag Center is a separate portal page. It combines the tag cloud and the tags result list.

"The portal tag cloud" on page 1322
The portal provides a **tag cloud** for aggregating tags for multiple resources. Users can select tags from the tag cloud.

"Browsing tags" on page 1324
Besides tagging content and viewing tags users can browse tags. This can be useful for finding related or similar content.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Related reference:

"CP Configuration Service for tagging and rating" on page 312
The CP Configuration Service provides the properties for tagging and rating.

"The tag and rating widgets"
The portal provides one widget each for tags and for ratings.

"Parameter reference for the tag and rating widgets" on page 1330
You can configure each of the portal tagging and rating features to determine the look and functionality of these features. To do so, you configure the tag and rating widgets.

"Configuration reference for tagging and rating" on page 1330
Developers can customize the tagging and rating user interfaces. The parameter reference lists and explains all available configuration parameters and their values. The topic about CSS provides information about how to change the design of the user interfaces.

The tag and rating widgets

The portal provides one widget each for tags and for ratings.

With IBM WebSphere Portal Express Version 8.5, both the dialog and inline tag and rating widgets of previous portal versions are deprecated. They are replaced by a single pair of new enhanced inline widgets. They provide the functionality of both types of the old widgets. Users view them as part of the portal resource.

The new inline tag and rating widgets can tag or rate content items on portal pages or in portlets, Article Template content items, blogs, wikis, or your custom content items. These content items can be on a portal page or inside a portlet. If you add the tag or rating widget to such a portal resource, the widget is integrated into that portal resource. The user views the user interface for tagging or rating as part of the portal resource. If you want to enable users to tag or rate a portal page or portlet, you need to use the old dialog widgets. Read the WebSphere Portal Express Version 8.0 documentation.

"The tag widget" on page 1311
Users can use the tag widget to view, apply, and update tags that were applied to a resource.

"The rating widget" on page 1315
Users can use the rating widget to view, apply, and update ratings that were applied to a resource.

Related reference:

"Tag widget parameter reference" on page 1331
You configure specific tag widget instances by setting the JavaScript parameters that are listed here.

"Rating widget parameter reference" on page 1334
You configure rating widget instances by setting the Javascript parameters listed here.

"Configuration reference for tagging and rating" on page 1330
Developers can customize the tagging and rating user interfaces. The parameter reference lists and explains all available configuration parameters and their values. The topic about CSS provides information about how to change the design of the user interfaces.

"Parameter reference for the tag and rating widgets" on page 1330
You can configure each of the portal tagging and rating features to determine the look and functionality of these features. To do so, you configure the tag and rating widgets.

"Security for tagging and rating" on page 1342
For administering which users can tag and rate content, the portal provides virtual resources for tagging and rating and roles on these virtual resources.

"CP Configuration Service for tagging and rating" on page 312
The CP Configuration Service provides the properties for tagging and rating.

Related information:

 IBM WebSphere Portal V 8 Product Documentation

The tag widget:

Users can use the tag widget to view, apply, and update tags that were applied to a resource.

The tag widget displays tags for users to view directly in the page. When users use the tag widgets, they can perform the following tasks:

Switch between tag scopes:
Users can switch the tag scopes by using the drop-down menu in the widget.

Add new tags:
Users can add new tags by using the input field that shows when a user clicks the **Plus** icon. Users can click this icon if the tag scope is **My public tags** or **My private tags**. Users can add more than one tag by using commas or spaces as separators.

Go to the Tag Center
If the **tagClickActionMode** parameter is set to TAG_CENTER, users can click a tag to go to the portal Tag Center page.

View more tags
If the number of tags exceed the number of tags that is configured to be shown, the user can click **More tags**. The widget then shows the extra tags in a dialog.

Delete tags:
The user can delete tags that the user added. To do so, the user clicks the delete icon that is shown next to the tags. The delete option is available only if the tag scope is **My public tags** or **My private tags**. A user can delete only the tags that the user added.

Users cannot modify tags in the tag widget.

The tag widget has the following user interface controls:

Label Displays a title that represents the scope of the tags that are displayed.

Depending on the setting of the tag scope parameter for the widget instance, the tag widget shows one of the following tag sets.

The title indicates the scope of the tags that are shown:

Tags With this scope, the widget shows the community tags. Community tags are tags that other users applied to the resource as public tags.

My public tags
With this scope, the widget shows personal public tags. Personal public tags are tags that the user applied to this resource.

My private tags
With this scope, the widget shows personal private tags. Personal private tags are all private tags that the user applied to this resource. This scope is available only when the `privateTaggingEnabled` parameter is set to `true`.

The scope of the tags that the widget shows depends on the setting of the tag scope parameter. For more information about this parameter, read the *Tag widget parameter reference*. You can set this parameter to one of the following values:

`COMMUNITY_PERSONAL_PUBLIC`
This is the default setting. With this setting, the tag widget shows the community tags and the personal public tags of the user

`PERSONAL_PUBLIC`
With this setting, the tag widget shows the personal public tags of the user.

`PERSONAL_PRIVATE`
With this setting, the tag widget shows the personal private tags of the user.

Tags The available tags are displayed after the title. The tag list uses commas as separators. Users can click the tags. Clicking a tag redirects the user to the tag center with the tag cloud. The tag cloud view scope depends on the setting of the **`tagClickActionMode`** parameter and on the scope that the user selected, **All** or **My public tags**. For more information about this parameter, read the *Tag widget parameter reference*. You can set this parameter to one of the following values:

`TAG_CENTER`
With this setting, the tag cloud shows the **All** view.

More tags
Users can configure how many tags the widget shows at the same time. If there are more tags than the user configured for display, the user can click **More tags**. The widget then shows the extra tags in a dialog.

Arrow for drop-down menu
Users can use this menu to switch between the tag scopes. The options in the drop-down menu depend on the tag scope that is displayed. For example, if the widget is in the **Tags** scope and shows the community tags, then clicking the drop-down menu shows an option that is called **My public tags**.

Plus (+) icon
The widget shows this icon. When a user clicks this icon, the widget opens an input field where the user can type tags.

"Adding the tag widget to your portal content"
By default, the tag widget is available for Web Content Manager article template pages and blogs and wikis. You can also add tag widgets to your portal content as required.

"Customizing the tag widget" on page 1314
The user interface of the tag widget consists of Web Content Manager HTML components. You can customize the tag widgets by modifying one or more of these components. For example, you can change the order of the user interface elements, or you can remove a field that you do not want to show in the user interface. The components are listed here.

Related reference:

"Tag widget parameter reference" on page 1331
You configure specific tag widget instances by setting the JavaScript parameters that are listed here.

"Properties for the tag widget" on page 315
View the properties for the tag widget.

Adding the tag widget to your portal content:

By default, the tag widget is available for Web Content Manager article template pages and blogs and wikis. You can also add tag widgets to your portal content as required.

About this task

To include a tag widget in a portal page, a portlet or other portal content, insert the following HTML snippet into your JSP or other code:

```
<div id="tagginglight_${resource_ID}" class="wpTaggingLight">
    <div class="label">
        <label class="text" id="tagginglightlabeltext_${resource_ID}"></label>
    </div>
    <div class="menu" >
        <a id="tagginglightmenulink_${resource_ID}" role="link" tabIndex="0">
            <img src='img_Src' border="0"  title="" />
        </a>
    </div>
    <div class="divider1" id="tagginglightdivider1_${resource_ID}"></div>
    <span id="tagginglighttagslist_${resource_ID}" class="tags" resource_ID="my_ID"></span>
    <div class="divider2" id="tagginglightdivider2_${resource_ID}">
    </div>
    <div class="moretags" id="moretagslink_${resource_ID}">
        <a id="tagginglightmoretagslink_${resource_ID}" href="javascript:;"
           onclick='getMoreTags("${resource_ID}");'>
        </a>
    </div>
    <div class="divider3" id="tagginglightdivider3_${resource_ID}"></div>
    <div class="addtag">
        <a id="tagginglightAddTaglink_${resource_ID}">
            <img src='img_Src' border="" width="12" height="12"  title="" />
        </a>
    </div>
    <span id="tagginglightinput_${resource_ID}" class="addtag">
        <input id="tagginglightinputField_${resource_ID}" type="text" name="text">
    </span>
    <div id="tagginglighterrordivider_${resource_ID}"  class="errorMessagedivider"></div>
    <div class="errorMessageStyle" id="tagginglighterrorDiv_${resource_ID}">
        <img src="" border="" width="16" height="16"  title="" />
        <label id="tagginglighterrorlabel_${resource_ID}"  class="errorMessageText" ></label>
    </div>
</div>
```

resource_ID
For the resource ID, specify the identifier of the piece of content that shows the widget. This identifier needs to be unique. For example, for a portal page, specify the portal object ID of that page.

src

Specify the appropriate values for the src attribute for the images. To obtain these values, copy them from the src attributes for images from the Web Content Manager HTML - Tag Widget Light - Menu component. Proceed as follows:

1. In Web Content Manager, go to **Applications** > **Content** > **WCM Authoring** > **Libraries** > **Web Resources v70** > **Components**.
2. Select the check box for **HTML - Tag Widget Light - Menu**.
3. Click **Read** and copy the src attribute value that you found here.
4. Paste the value into your code for including the tag widget.

These parameters are mandatory. For information about the optional parameters, read the *Tag widget parameter reference*.

Related reference:

"Tag widget parameter reference" on page 1331
You configure specific tag widget instances by setting the JavaScript parameters that are listed here.

Customizing the tag widget:

The user interface of the tag widget consists of Web Content Manager HTML components. You can customize the tag widgets by modifying one or more of these components. For example, you can change the order of the user interface elements, or you can remove a field that you do not want to show in the user interface. The components are listed here.

About this task

Changes that you make here apply to all instances of the rating widget.

Some of the components are mandatory. Do not remove them.

To modify the components, navigate to **Applications** > **Content** > **WCM Authoring** > **Libraries** > **Web Resources v70** > **Components**.

HTML - Tagging Widget Light - Label:

This component is mandatory. It renders the default label for the tag widget. The tag widget shows one of the following tag sets. The shown tag set depends on the setting of the tag scope parameter for the widget instance. For more information about this parameter, read the *Tag widget parameter reference*.

Community tags

With the tag scope parameter set to this value, the tag widget shows tags that other users applied to the resource as public tags.

Personal public tags

With the tag scope parameter set to this value, the tag widget shows all public tags that the user applied to this resource.

Personal private tags

With the tag scope parameter set to this value, the tag widget shows all private tags that the user applied to this resource.

The following labels are shown for the corresponding tag scopes:

Tags This label indicates COMMUNITY_PERSONAL_PUBLIC tags.

My public tags
This label indicates PERSONAL_PUBLIC tags.

My private tags
This label indicates PERSONAL_PRIVATE tags.

HTML - Tagging Widget Light - Tags:
This component is mandatory. It renders the list of tags for the resource.

HTML - Tagging Widget Light - Menu:
This component is mandatory. It renders the drop-down menu that is used for switching between the different tag scopes.

HTML - Tagging Widget Light - More Tags:
This component is mandatory. It renders the **More tags** link that is used to show the extra tags that cannot be displayed by using the HTML - Tagging Widget Light - Tags component. When a user clicks **More tags**, the widget shows the extra tags.

HTML - Tagging Widget Light - Divider1:
This component renders the divider that separates the tags and the **More tags** link.

HTML - Tagging Widget Light - Divider2:
This component renders the divider that separates the **More tags** link and the drop-down menu.

HTML - Tagging Widget Light - Divider3:
This component renders the divider that separates the drop-down menu and the plus (**+**) icon.

HTML - Tagging Widget Light - ErrorMessageDivider:
If the widget shows an error message, this component renders the divider that separates the tags and the error message.

HTML - Tagging Widget Light - Add Tags - Image:
This component is mandatory. It renders the plus (**+**) icon for showing the input field for adding tags.

HTML - Tagging Widget Light - Add Tags - Input:
This component is mandatory. It renders the input field in which a users adds tags.

HTML - Tagging Widget Light - Messages:
This component is mandatory. If the widget shows an error message while a user is using the tag widget, this component renders the error message.

HTML - Tagging Widget Light:
This component contains all the components shown earlier in this list.

Related reference:

"Tag widget parameter reference" on page 1331
You configure specific tag widget instances by setting the JavaScript parameters that are listed here.

The rating widget:

Users can use the rating widget to view, apply, and update ratings that were applied to a resource.

The rating widget displays ratings for users to view directly in the page. The rating widget shows ratings for the resource that has the widget included. When users use the rating widget, they can perform the task given in the following list.

Note: For these tasks, a user needs the rights to edit the IBM Web Content Manager component library `Web Resources V70`.

Switch between rating scopes:
Users can view the rating value for other rating scopes. They can switch to another rating scope by clicking the rating menu.

Display Rating Description:
To display the rating description in the rating widget, users must do both of the following tasks:

- Add the Web Content Manager HTML component named `HTML - Rating Widget Light - Description` to the `HTML - Rating Widget Light`.
- Set the attribute `ratingDescription` to `ALL` in the `Web Resources v70\Components\HTML - Rating Widget Light - Description - Stars`.

The rating description is displayed only for community average ratings.

Add a rating:
Users can add personal public or personal private ratings. To do so, they switch to the **Personal public** or **Personal private** rating scope and then assign a rating by clicking the appropriate asterisk.

Change a rating:
User can change ratings that they assigned. To do so, they click the appropriate asterisk. Users can change a shown rating only if that rating is a personal public or personal private rating and the widget shows the **Personal public** or **Personal private** rating scope. The default rating view displays the community average rating. When a user clicks the asterisks, the scope changes to PERSONAL_PUBLIC and the rating value is updated.

Remove a Rating:
Users can remove a rating by using the **Delete** option in the **Action** menu. The Delete option is available only if a rating was assigned to the resource and the rating value is not zero (`0`).

Customize the Web Content Manager components in the rating widget:
Users can reorder, remove, and edit the properties of Web Content Manager components in the rating widget. The rating widget definition is available in the Web Content Manager component `Web Resources v70\Components\HTML - Rating Widget Light`. For example, to move the rating description before the rating stars, the user can move the rating description subcomponent before the rating asterisks in the component definition. Here is the list of all subcomponents:

- HTML - Rating Widget Light - Label
- HTML - Rating Widget Light - Stars
- HTML - Rating Widget Light - Menu
- HTML - Rating Widget Light - Description
- HTML - Rating Widget Light - Divider1
- HTML - Rating Widget Light - Divider2

The rating widget has the following user interface controls:

Label The label displays a title that represents the scope of the rating that is displayed. The rating widget shows the rating, depending on the setting of the rating scope parameter for the widget instance. For more information about this parameter, read the *Rating widget parameter reference*.

(The default scope has no label)
The default scope shows the average of all ratings that all users applied to the resource as public ratings.

My public rating
This scope shows all public ratings that the user applied to this resource.

My private rating
This scope shows all private ratings that the user applied to this resource.

The scope of the ratings that the widget shows depends on the setting of the rating scope parameter. For more information about this parameter, read the *Rating widget parameter reference*. You can set this parameter to one of the following values:

COMMUNITY_PERSONAL_PUBLIC
With this setting, the rating widget shows the community ratings, in other words the ratings that all users applied to this resource. This value is the default value.

PERSONAL_PUBLIC
With this setting, the rating widget shows the personal public ratings of the user.

PERSONAL_PRIVATE
With this setting, the rating widget shows the personal private ratings of the user.

Asterisks
The rating widget shows the rating by the number of asterisks. The more asterisks are highlighted, the better the rating is for the resource.

Drop-down menu
This menu can show the following options:

An option to switch between rating scopes
Users can select to a different rating scope. The options that the menu shows depend on the current rating scope. For example, if the current scope is Community personal public rating, the menu shows the option Personal public rating.

Rating distribution
Users can also view the distribution of ratings. To do so, the users select **Rating distribution** from the menu. This option is available only when the rating scope is Community personal public rating.

Rating description
The rating description shows the numerical rating value and the total number of ratings that users applied to this resource.

- By default the widget displays five asterisks (*****), where five stars are the highest rating and one star is the lowest rating that a resource can have.
- The rating widget shows a tooltip with a numeric representation of the rating value, for example `Rating: 3.4/5`.

"Adding the rating widget to your portal content" on page 1318
By default, the rating widget is available for Web Content Manager article template pages and blogs and wikis. You can also add rating widgets to your portal content as required.

"Customizing the rating widget" on page 1319
The user interface of the rating widget consists of Web Content Manager HTML components. You can customize the rating widgets by modifying one or more of these components. For example, you can change the order of the user interface elements, or you can remove a field that you do not want to show in the user interface. The components are listed here.

Related reference:

"Rating widget parameter reference" on page 1334
You configure rating widget instances by setting the Javascript parameters listed here.

"Properties for the rating widget" on page 317
View the properties for the rating widget.

Adding the rating widget to your portal content:

By default, the rating widget is available for Web Content Manager article template pages and blogs and wikis. You can also add rating widgets to your portal content as required.

About this task

To include a rating widget in a page or a portlet or other content, insert the following HTML snippet into your JSP or other code:

```
<div class="ratingLightWidget" resourceID= "${contentItem.URI}"
     id="ratingLightWidget_${contentItem.URI}">
    <div id="ratinglight_${contentItem.URI}" class="wpRatingLight">
        <div class="label">
            <div id="ratinglightlabeltext_${contentItem.URI}" class="text"></div>
        </div>
        <div class="menuShow" id="ratinglightmenuDIV_${contentItem.URI}" role="link">
            <a onkeypress="getRatingMenu('${contentItem.URI}')"
               onclick="getRatingMenu('${contentItem.URI}')"
               id="ratinglightmenulink_${contentItem.URI}" role="link" tabindex="0">
                <img alt="Rating Menu" src='img_Src' title="Rating Menu" border="0">
            </a>
        </div>
        <div id="ratinglightdivider2_${contentItem.URI}" class="divider2Show"></div>
        <div class="description">
            <div id="ratinglightdescriptionlabel_${contentItem.URI}"></div>
        </div>
        <div id="ratinglightdivider1_${contentItem.URI}" class="divider1Show"></div>
        <div class="stars" resourceid="${contentItem.URI}"
             id="ratinglightstars_${contentItem.URI}">
            <div style="width: 70px;" id="ratinglightstarsempty_${contentItem.URI}"
                 class="empty" tabindex="0">
                <div style="width: 0px;" id="ratinglightstarsfull_${contentItem.URI}"
                     class="full"></div>
            </div>
        </div>
        <div id="ratinglighterrordivider_${contentItem.URI}" class="errorMessagedivider"></div>
        <div class="errorMessageStyle" id="ratinglighterrorDiv_${contentItem.URI}">
            <img alt="Error - check console" src='errorimg_Src'
                 title="Error - check console" height="16" border="" width="16">
            <label id="ratinglighterrorlabel_${contentItem.URI}" class="errorMessageText"></label>
        </div>
    </div>
</div>
```

Specify the appropriate values for the parameters:

resource_ID
For the resource ID, specify the identifier of the piece of content that you want to show the widget. This identifier needs to be unique. For example, for a portal page, specify the portal object ID of that page.

***xxxxx*_src**

To obtain the values for the src attribute for the images, copy them from the src attributes for images from the Web Content Manager menu component. For each attribute, use the specific procedure:

img_Src

For the img_Src attribute, proceed as follows:

1. In Web Content Manager, go to **Applications** > **Content** > **WCM Authoring** > **Libraries** > **Web Resources v70** > **Components**.
2. Select the check box for **HTML - Rating Widget Light - Menu**.
3. Click **Read** and copy the src attribute value that you found here.
4. Paste the value into your code for including the rating widget.

errorimg_Src

For the errorimg_Src attribute, proceed as follows:

1. In Web Content Manager, go to **Applications** > **Content** > **WCM Authoring** > **Libraries** > **Web Resources v70** > **Components**.
2. Select the check box for **HTML - Rating Widget Light - Messages**.
3. Click **Read** and copy the src attribute value that you found here.
4. Paste the value into your code for including the rating widget.

These parameters are mandatory. For information about the optional parameters, read the *Rating widget parameter reference*.

Related reference:

"Rating widget parameter reference" on page 1334
You configure rating widget instances by setting the Javascript parameters listed here.

Customizing the rating widget:

The user interface of the rating widget consists of Web Content Manager HTML components. You can customize the rating widgets by modifying one or more of these components. For example, you can change the order of the user interface elements, or you can remove a field that you do not want to show in the user interface. The components are listed here.

About this task

Changes that you make here apply to all instances of the rating widget.

Some of the components are mandatory. Do not remove them.

HTML - Rating Widget Light - Label:

This component is mandatory. Do not remove it. This component renders the default label for the rating widget. The rating widget shows one of the following types of rating. The shown rating type depends on the setting of the rating scope parameter for the widget instance. For more information about this parameter, read the *Rating widget parameter reference*.

Personal public rating

With the rating scope parameter set to this value, the rating widget shows all public ratings that the user applied to this resource.

Personal private rating

With the rating scope parameter set to this value, the rating widget shows all private ratings that the user applied to this resource.

The following labels are shown for the corresponding rating scopes:

My public rating
This label indicates PERSONAL_PUBLIC rating.

My private ratings
This label indicates PERSONAL_PRIVATE rating.

HTML - Rating Widget Light - Menu:
This component is mandatory. Do not remove it. This component renders the drop-down menu that is used for switching between the different rating scopes.

HTML - Rating Widget Light - Divider1:
This component renders the divider that separates the menu and the rating asterisks.

HTML - Rating Widget Light - Stars:
This component is mandatory. Do not remove it. This component renders the asterisks that show the rating for the resource.

HTML - Rating Widget Light - Divider2:
This component renders the divider that separates the menu and the rating description.

HTML - Rating Widget Light - Divider3:
If the widget shows an error message, this component renders the divider that separates the rating description and the error message.

HTML - Rating Widget Light - ErrorMessageDivider:
If the widget shows an error message, this component renders the divider that separates the ratings and the error message.

HTML - Rating Widget Light - Messages:
This component is mandatory. Do not remove it. If the widget shows an error message while a user is using the rating widget, this component renders the error message.

HTML - Rating Widget Light:
This component contains all the components that are shown earlier in this list.

Related reference:

"Rating widget parameter reference" on page 1334
You configure rating widget instances by setting the Javascript parameters listed here.

The portal Tag Center

The Tag Center is a separate portal page. It combines the tag cloud and the tags result list.

Users can get to this page by one of the following ways:

- By clicking the **Tag Center** link from the portal menu bar.
- By selecting the option **Browse tags** from the page action menu or the portlet menu.
- By clicking a tag from a custom tag cloud that you placed on a page of your portal.
- By clicking a tag from an tag widget in a portal or custom resource, for example a portal page, a portlet, a wiki, or a blog.

By default the Tag Center page shows the following two portlets:

1. The **Tag Cloud** portlet. Refer to the topic about the tag cloud. Users can select different views in the Tag Cloud:
   - **All tags** view
   - **IBM Connections tags** view. The federation view can show Activities, Blogs, Bookmarks, Communities, Files, Forums, Profiles, or Wikis.
   - **Others tags** view
   - **My public tags** view
   - **Latest tags** view
   - **My private tags** view
2. The **Tag Results** portlet. It shows resources that have been tagged with the tags that the users selected from the tag cloud. For example, if a user selects the tag **IBM** from the tag cloud, the Tag Result List portlet shows resources that have been tagged with the tag IBM. The Tag Result List portlet displays for each resource its name, its optional description, the tags that have been assigned to it, and the average community rating. In the Tag Results portlet users can click icons to choose between a **Summary** view and a **Details** view. The Summary view is the default.

   Users can click the shown tags and resources:
   - If the user clicks a tag listed under a resource in the result list, both the tag cloud and the tag results list change to show the information for the newly selected tag just as if the user had selected the tag from the tag cloud:
     – The tag cloud highlights the tag that the user selected from the tag result list and changes to the view from which the user selected the tag in tag the result list.
     – The tag result list shows all resources that have been tagged with the newly selected tag.

     Example: Initially the user selects the tag TAG_1 from the **All tags** view of the tag cloud. The tag result list shows all resources that have TAG_1 applied. Then the user selects TAG_2 from the Your private tags under one of the resources in the tag result list. As a result, the tag cloud highlights TAG_2 and changes to the **My private tags** view, and the tag result list shows all resources that have been tagged with TAG_2.
   - If the users clicks a resource, the portal redirects the user to that resource.

The combination of view options selected in both the tag cloud and in the tag result list determine what the tag result list shows. Refer to the following table:

Table 170. How the selected view options of the Tag Cloud and Tag Results portlets determine what the Tag Results list shows

Tag cloud portlet view selected	**Tag result portlet**	**Tag result portlet**
	Summary view shows these tags:	**Details** view adds these tags to the view:
All tags view	Community tags personal public tags optional Connections tags	personal private tags
Connections tags view	Connections tags (federated tags)	description of the resource
Others' tags view	community tags	personal public tags
My public tags view	personal public tags	description of the resource

Table 170. How the selected view options of the Tag Cloud and Tag Results portlets determine what the Tag Results list shows (continued)

Tag cloud portlet view selected	Tag result portlet	Tag result portlet
Latest tags view	community tags personal public tags	personal private tags
My private tags view	personal private tags	description of the resource

Examples:

- User A views tags in the tag cloud by using the Community view. When user A clicks **TAG_3**, the Tag Results portlet lists the summary view of all resources that have been tagged with TAG_3. Each resource is listed with all community tags that users applied to it. When user A changes to the details view by clicking **Details view** icon, all personal public tags that user A has applied to the resource are displayed additionally.
- User B views tags in the tag cloud by using the Personal View. When user B clicks **TAG_4**, the Tag Results portlet lists the summary view of all resources that have been tagged with TAG_4. Each resource is listed with all personal public tags that user B has assigned to this resource. When user B changes to the detail view by clicking **Details view** icon, descriptions of the resources are displayed additionally.

Users can sort the results by resource type, date, resource title, and rating.

Note: Resources given in the Tag Result List portlet might not always show the tag that the user selected from the tag cloud to start the search. This depends on how the resource has been otherwise tagged by users. If other tags have been applied to the resource more often than the tag by which the user searched, then those tags will take precedence among the tags shown for that resource. By default the Tag Result List portlet shows the three most popular tags; you can configure this number for the portlet.

When users are on the Tag Center page, they can select additional tags from the cloud to narrow down their result list. Users can clear tags, either by clicking them in the cloud or by clearing them from the Tag Result List portlet.

The portal tag cloud

The portal provides a **tag cloud** for aggregating tags for multiple resources. Users can select tags from the tag cloud.

The tag cloud is available as a portlet and as a Dojo widget. You can add the tag cloud portlet to any page as required. You can also integrate the tag cloud widget into a theme or embed it into a portlet.

The tag cloud shows the tags in alphabetical order. The font size indicates how often the tag has been applied within a defined scope. The larger the font of a tag is, the more often it as been applied. Users can select from the following options:

- Users can view in a cloud view or in a list view:
  - In the cloud view users can determine whether they wan to view more or fewer tags by moving a slider.
  - The list view has a pagination bar.
- Users can select different tag cloud views to filter what is displayed in the tag cloud:

- The **All tags** view shows all tags that have been applied by users in the portal. This includes all community tags and all personal public and personal private tags of the user who is viewing the tags. You can also configure the **All tags** view to include tags from remote systems.
- The **IBM Connections tags** view shows all public tags from remote systems for the currently selected federation features. These tags can be embedded in the **All tags** view.
- The **Others** view shows all public portal tags applied by other users, except for the user's own personal public tags.
- The **My public tags** view shows only the public portal tags that the user who invoked the cloud has assigned.
- The **Latest tags** view shows only the portal tags that have been created most recently.
- The **My private tags** view shows only the private portal tags that the user who invoked the cloud has assigned.

Administrators can configure the following additional settings:

- Whether they want only related tags to be displayed when a users clicks a tag. For example, if a user clicks a tag TAG_1 Related means that the tag cloud then shows only the tag TAG_1 clicked by the user and all tags that have been applied to resources to which TAG_1 has also been applied. Tags that have been applied to resources without being in combination with TAG_1 are not shown.
- The minimum number of tags that they want to be displayed when the user moves the slider all the way to the minimum position
- The maximum number of tags that they want to be displayed, that is when the slider is moved all the way to the maximum position.
- The default number of tags that they want to be initially displayed and the corresponding position of the slider.

- Users can select between different scoping modes:
  - Users can change the scope for which they want to view tags, for example books or movies. Users or administrators can configure the available scopes, depending on their access rights. For details about this refer to the topic about Configuring the tag cloud.
  - Administrators can configure the available scopes by using the option **Edit shared settings** from the tag cloud menu. When a users selects a scope, only tags that have been assigned to resources in that scope are displayed. For more information and a scoping example see the topic about *Grouping tags and ratings via resource categorization*.
  - You can also scope the tag cloud in a way that it only shows tags that have been assigned to resources of a certain category or type schema. For details see *Configuring the tag cloud*.

For more details refer to the help topic about the *Tag cloud*.

The tag cloud portlet has several configuration options. You can configure each individual tag cloud portlet instance separately by using its **Edit shared settings** menu. For details see the topic about *Configuring the tag cloud*.

Related reference:

"Grouping tags and ratings via resource categorization" on page 1304
Users apply tags and ratings to resources. Users can tag and rate all resources that can be uniquely identified.

Browsing tags

Besides tagging content and viewing tags users can browse tags. This can be useful for finding related or similar content.

For example, when a user views a resource such as a weather portlet, the user might want to know if there are similar portlets available. In this case the user can select the **Browse tags** feature to search resources tagged with the same tags, and optionally additional ones.

For a page users select the **Browse Tags** option from the Actions menu link in the page header. For a portlet users select the **Browse Tags** option from the portlet menu.

Clicking **Browse Tags** redirects the user to the tag center. The tag cloud shows the **All** view. The user can now select tags and views as required.

Tagging and rating for static pages

Tagging or rating for static pages works only with the dialog widgets of earlier portal versions.

Starting with IBM WebSphere Portal Express Version 8.5, you can use tagging or rating for static pages only by using the dialog widgets of earlier portal versions. For information about how to do this, read the WebSphere Portal Express Version 8.0 documentation.

Note: Tagging or rating for static pages works only with the dialog widgets, but not with the inline widgets of earlier portal versions.

Related reference:

"The tagging and rating user interface" on page 1309
Learn about the user interface that IBM WebSphere Portal Express provides for users to work with tags and rating.

Enabling your own custom content for tagging and rating

Enabling your own custom content for tagging and rating works only with the dialog widgets of earlier portal versions.

Starting with IBM WebSphere Portal Express Version 8.5, you can enable your own custom content for tagging and rating only by using the dialog widgets of earlier portal versions. For information about how to do this, read the WebSphere Portal Express Version 8.0 documentation.

Note: Tagging or rating for your own custom content works only with the dialog widgets, but not with the inline widgets of earlier portal versions.

Federating tags

WebSphere Portal allows the federation of remote tagging systems, such as Connections.

Tags from remote tagging systems can be integrated in the WebSphere Portal Tag Cloud. When a tag from a remote system is selected in the Tag Cloud, the Tag Results portlet lists all federated resources to which users have applied the tag. The title of the remote resource, and the description if available, are displayed. The

title is preceded by an icon that symbolizes the resource type of the resource. For details about how you can specify an icon for a federated resource type see *Administration of tag federation*.

If a federated resource is selected two different kinds of behavior are supported:

1. Redirect to an external website where the resource is displayed. This external redirection is always possible.
2. Redirect to a page with a portlet that can display this resource. This internal redirection requires a portlet that can integrate a remote resource in WebSphere Portal.

For details about how you can define a portlet as a target of an internal redirection see *Administration of tag federation*.

Configuration settings for the WebSphere Portal tagging and rating features such as normalization, black- and white lists are also applied to the federated tags. That allows for example the scenario that remote tags are not visible in WebSphere Portal because they do not fit to the configured settings.

Federation of Connections tags

Connections comprises multiple features, such as Activities, Blogs, Bookmarks, Communities, Files, Forums, Profiles, and Wikis. Tags that belong to the resources of these features can be integrated in the Tag Cloud. When a Connections tag is selected, links with the titles of corresponding Connections resources are listed in the Tag Results portlet. When such a resource is selected, either a redirect to a Portal page with the corresponding Connections portlet is performed, or a redirect to the Connections website is made to display more detailed information about the resource.

For Blogs, Forums, Wikis and Profiles, the selected resource is rendered in a Connections portlet by default. For more information on how a target page for the redirect can be specified and how to deploy and configure the Connections POC Resolver see the Connections documentation. The Connections POC resolver is required to render tagged Connections resources within Connections Portlets.

If you do not want to have a connections resource rendered in a Connections portlet, you can set a Connections Site as target of a redirect for a particular Connections feature. For details see *Administration of tag federation*.

Note: The Connections portlets must be deployed into WebSphere Portal and configured accordingly .

The following specifics regarding Connections need to be mentioned:

Connections does not distinguish between private and public tags like WebSphere Portal does. Therefore, the Connections tags are available in the Connections tags view in the tag cloud for WebSphere Portal Express. The **All tags** view can be configured to include Connections tags. These tags are not integrated in the **Others' tags**, **My public tags**, or **My private tags** views.

The most frequently used tags are retrieved through the Connections feature. A limit of 100 tags is provided by Connections.

As Connections does not provide globalization information for tags nor for related resources, the same tag name, title and description of resources are displayed no matter which locale is selected by the user.

During configuration or administration of the Connections integration there are steps where the identifier of a Connections feature must be supplied. Use the following overview of all Connections features identifiers to select the correct feature identifier:

Table 171. Connections feature identifiers

Connections Feature	Feature identifier
Activities	activities
Blogs	blogs
Bookmarks	dogear
Communities	communities
Files	files
Forums	forums
Profiles	profiles
Wikis	wikis

"Administering tag federation"
When tags from remote systems, such as IBM Connections are integrated into your WebSphere Portal Express site, you need to schedule tasks to retrieve tags and related data from the remote system, and later to clean them up from the portal. You can also redirect the rendering of federated resources to IBM Connections and add icons to federated resources.

Administering tag federation

When tags from remote systems, such as IBM Connections are integrated into your WebSphere Portal Express site, you need to schedule tasks to retrieve tags and related data from the remote system, and later to clean them up from the portal. You can also redirect the rendering of federated resources to IBM Connections and add icons to federated resources.

"Importing federated tags and resources" on page 1327
When tags from remote systems, such as Connections are integrated into WebSphere Portal Express, you need to schedule a task to retrieve the tags and related data from the remote system.

"Cleaning up federated tags and resources" on page 1328
When federated tags are no longer integrated in WebSphere Portal Express, you invoke the task `com.ibm.wps.cp.tagging.federation.taskhandler.FederationDeletionTaskHandler` to remove unnecessary data.

"Redirecting to a Connections site" on page 1328
When a user clicks a Connections resource in the result list portlet, this resource is rendered within a Connections portlet, if that portlet exists and can handle this resource.

"Specifying an icon for a federated resource" on page 1329
When federated resources are displayed in the Tag Results portlet, you can have them preceded by a icon.

Related tasks:

"Configuring task to retrieve tags" on page 753
When you integrate Connections with your portal server, the portal uses a task to retrieve tags and related resources from the various Connections features (such as activities, blogs, bookmarks, communities, files, forums, profiles, and wikis). Then, the portal integrates the tags in the portal tag cloud. You can schedule the task to retrieve the tags to run periodically.

Importing federated tags and resources:

When tags from remote systems, such as Connections are integrated into WebSphere Portal Express, you need to schedule a task to retrieve the tags and related data from the remote system.

About this task

To retrieve tags and related resource for federation, schedule a task named `com.ibm.wps.cp.tagging.federation.taskhandler.FederationTaskHandler` WebSphere Portal Express by using the XML configuration interface (XMLAccess). In the following example, the XML task `triggerTask.xml` is scheduled to run the task `com.ibm.wps.cp.tagging.federation.taskhandler.FederationTaskHandler` once a day:

```
<?xml version="1.0" encoding="UTF-8"?>
   <request type="update" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd">
      <portal action="locate">
         <task action="create" name="com.ibm.wps.cp.tagging.federation.taskhandler.FederationTaskHandler">
            <startTime>22:00</startTime>
         </task>
      </portal>
   </request>
```

This tasks handles all Connections features.

You can also select and specify which Connections features you want a task to handle. For example, you can have data for Connections wikis and blogs collected on a different schedule than Connections files. Refer to the following example:

```
<?xml version="1.0" encoding="UTF-8"?>
<request type="update" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd">
     <portal action="locate">
          <task action="create" name="com.ibm.wps.cp.tagging.federation.taskhandler.FederationTaskHandler">
              <startTime>12:00</startTime>
              <parameter>wikis</parameter>
              <parameter>blogs</parameter>
        </task>
        <task action="create" name="com.ibm.wps.cp.tagging.federation.taskhandler.FederationTaskHandler">
              <startTime>07:00</startTime>
          <parameter>files</parameter>
        </task>
      </portal>
</request>
```

Notes:

- The `<parameter>` element contains the ID of the Connections feature. These IDs are listed on the page Federating Tags.
- Carefully consider which features you want to integrate into portal. With each Connections feature, there is potential for coupling large amounts of data.

To retrieve changes in the Connections tag cloud, you need to run the `FederationTaskHandler` script.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Cleaning up federated tags and resources:

When federated tags are no longer integrated in WebSphere Portal Express, you invoke the task `com.ibm.wps.cp.tagging.federation.taskhandler.FederationDeletionTaskHandler` to remove unnecessary data.

About this task

The following example XML script `triggerDeleteTask.xml` shows how you schedule the task to cleanup federated tags:

```
<?xml version="1.0" encoding="UTF-8"?>
    <request type="update" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd">
        <portal action="locate">
        <task action="create"
              name="com.ibm.wps.cp.tagging.federation.taskhandler.FederationDeletionTaskHandler"/>
</portal></request>
```

Notes:

- As this example does not explicitly define a schedule, the task is performed immediately.
- You can also select and specify the Connections features for which you want the cleanup to be run. For details about selecting features see the topic about Importing federated tags and resources.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Redirecting to a Connections site:

When a user clicks a Connections resource in the result list portlet, this resource is rendered within a Connections portlet, if that portlet exists and can handle this resource.

About this task

For more information about deployment and configuration of the portlets and the Connections POC Resolver, see the Connections documentation.

If you do not require this behavior, you can redirect to a Connections server website. You configure this redirection in the Resource Environment Provider for the WP CP Configuration Service for tagging and rating in the WebSphere Integrated Solutions Console. For each Connections feature, a custom property exists. The name of the property follows this pattern:

`com.ibm.wps.cp.tagging.federation.uri.scheme.`*`FederatorID`*

To redirect to the Connections server, use the following value pattern: `connections - FederatorID` .

This setting is used in the next run of the FederationTaskHandler that retrieves Connections tags and tagged resources. After that a redirect to the external Connections website is possible.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Related reference:

"CP Configuration Service for tagging and rating" on page 312
The CP Configuration Service provides the properties for tagging and rating.

Specifying an icon for a federated resource:

When federated resources are displayed in the Tag Results portlet, you can have them preceded by a icon.

About this task

To define an icon for a resource type, you specify a custom property with either a URI for the image or a relative path that is attached to the base URL of the Connections service of the resource type. The name of the custom property for the icon follows this pattern:

`com.ibm.wps.cp.tagging.federation.iconURL.FederatorID`

For more information about a Connections feature see the topic about Federating tags.

You configure the custom properties in the Resource Environment Provider for the WP CP Configuration Service for tagging and rating in the WebSphere Integrated Solutions Console.

Note: The icons are part of the portal installation. For theConnections features, the custom properties for the icons exist without a predefined path. Unless no icon URL is defined, the icons are loaded directly from WebSphere Portal Express.

Related concepts:

"Federating tags" on page 1324
WebSphere Portal allows the federation of remote tagging systems, such as Connections.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Related reference:

"CP Configuration Service for tagging and rating" on page 312
The CP Configuration Service provides the properties for tagging and rating.

Configuration reference for tagging and rating

Developers can customize the tagging and rating user interfaces. The parameter reference lists and explains all available configuration parameters and their values. The topic about CSS provides information about how to change the design of the user interfaces.

"Parameter reference for the tag and rating widgets"
You can configure each of the portal tagging and rating features to determine the look and functionality of these features. To do so, you configure the tag and rating widgets.

"CSS classes for tagging and rating" on page 1337
The portal tag and rating widgets allow for detailed look and feel customization by providing a customizable CSS class hierarchy.

"Enabling and disabling the tag and rating widgets for additional profiles" on page 1340
In a portal installation, the tag and rating widgets are available for specific portal profiles. You can enable the widgets for more other profiles by adding them to the profile.

"Enabling and disabling the Dojo tagging and rating options for additional profiles" on page 1341
In a portal installation, the Dojo tagging and rating menu options for portal pages and portlets are available for a specific portal profile. You can enable these options for other profiles by adding them to the profile.

Parameter reference for the tag and rating widgets

You can configure each of the portal tagging and rating features to determine the look and functionality of these features. To do so, you configure the tag and rating widgets.

How to configure the tag and rating widgets

You can configure all tag and rating widgets globally by setting parameters in the portal CP configuration service in the WebSphere Integrated Solutions Console. You can also configure individual instances of the tag and rating widgets by using the JavaScript parameters in the following topics. These settings become effective for the affected individual widget instance and override the settings set in the CP configuration service. The parameter settings take override priority by the following hierarchy:

1. Settings that are specified for an individual tagging or rating widget instance take highest priority.
2. If a parameter is not set for the individual widget instance, the settings specified in the CP Configuration Service takes effect.
3. If a parameter is not set by either of the two previous options, the default value defined in the widget class itself takes effect.

Example: In the CP Configuration Service, the attribute for custom labels `com.ibm.wps.cp.tagging.inline.customLabel` has no default value. The tag widget ABC instance includes the attribute `customLabel=MyTags`. As a result, all tag widgets in the portal have the custom label set to an empty string, as no default value is set. The tag widget ABC is the exception, because it has the attribute `customLabel=MyTags` set.

Most of the parameters exist as corresponding sibling parameters that you can set both globally in the CP Configuration Service and for individual widget instances

and in your code. The parameter in the CP Configuration Service consists of the prefix `com.ibm.wps.cp.` and the related parameter for configuring widget instances. Example:

- Parameter in the CP Configuration Service: `com.ibm.wps.cp.tagging.inline.maxResults`
- Parameter for an individual tag widget instance: `maxResults`.

Some of the parameters have no corresponding siblings and exist only for one of the two ways of configuring them. The parameters that exist only for configuring widget instances are mandatory, as they identify the resource for which the widget is called.

For details about the parameters in the CP configuration service and how to set them, read the topics about the portal *CP Configuration Service for tagging and rating* and *Setting service configuration properties*.

Usage notes for the parameter lists in the following topics:

- The lists show the parameters with their default if they have one. In these cases the parameters are shown with an equal sign between the parameter and the value. This is how you specify the setting when you include the widget declaratively. If you include the widget programmatically, use a colon instead of the equal sign.
- The parameters listed here are specific to the inline tag and rating widgets that were introduced with IBM WebSphere Portal Express Version 8.5. The dialog and inline widgets of earlier portal versions were deprecated with portal V 8.5. For information about the earlier widgets and their parameters, read the appropriate topics in the WebSphere Portal Express Version 8.0 product documentation.

  "Tag widget parameter reference"
  You configure specific tag widget instances by setting the JavaScript parameters that are listed here.

  "Rating widget parameter reference" on page 1334
  You configure rating widget instances by setting the Javascript parameters listed here.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Related reference:

"CP Configuration Service for tagging and rating" on page 312
The CP Configuration Service provides the properties for tagging and rating.

"How tagging and rating works in the portal" on page 1303
Use these topics for general administrative information about tagging and rating in the portal.

Related information:

IBM WebSphere Portal V 8 Product Documentation

Tag widget parameter reference:

You configure specific tag widget instances by setting the JavaScript parameters that are listed here.

Note: The parameters listed here are specific to the inline tag widget that was introduced with IBM WebSphere Portal Express Version 8.5. For information about the earlier widgets and their parameters, read the appropriate topics in the WebSphere Portal Express Version 8.0 product documentation.

To set the parameters, proceed as follows:

1. In IBM Web Content Manager, go to **Applications** > **Content** > **WCM Authoring** > **Libraries** > **Web Resources v70** > **Components**.
2. Select the check box for **HTML - Tagging Widget Light - Menu**.
3. Click **Edit**.

The following parameters are optional for the tag widget. They correspond to similar parameters in the WP CP Configuration Service.

countsEnabled = false
Use this parameter to specify whether the count of each community tag is displayed. The count shows how often the tag was applied by users. The default value is `false`. To show the count of each community tag, specify `true`. The count is then displayed in parentheses.

customLabel
Use this parameter to specify a non-localized custom label to describe the displayed tags. If you do not want any further labeling or if you want to keep the label short, you can specify an empty string.

customLabelCommunityTags = = (false)
Use this parameter to specify the non-localized label that you want to be used to describe the meaning of the community tagging that is displayed. The default value is `false`.

customLabelPersonalPublicTags = (false)
Use this parameter to specify the non-localized label that you want to be used to describe the meaning of the personal public tagging that is displayed. The default value is `false`.

customLabelPersonalPrivateTags = (false)
Use this parameter to specify the non-localized label that you want to be used to describe the meaning of the personal private tagging that is displayed. The default value is `false`.

customMessageNoTags
Use this parameter to specify a non-localized custom label that you want to be displayed if no tags are available. This case can occur when users did not assign tags to the piece of content yet. Non-localized means that the label does not change with the browser language. This parameter has no default value.

maxResults = 5
Use this parameter to specify the number of tags that are shown per resource. The default value is `5`.

order = DESC|ASC
Use this parameter to specify the order direction for displaying the tags. This parameter is related to the parameter `orderMetric` . It can take the following values:

DESC This value is the default value. It specifies descending order. For example, if the default `TAG_SPACE_COUNT_REVERSE_NAME` is specified

for the orderMetric parameter that is listed later, tags with the highest count and the lowest character in the alphabet are listed first.

ASC This value specifies ascending order. For example, if the order metric parameter is specified as orderMetric = TAG_SPACE_COUNT, tags with the lowest count are listed first.

orderMetric = TAG_SPACE_COUNT_REVERSE_NAME

Use this parameter to specify the order metric for the order by which the tags are displayed. To determine the actual order, use the parameter order that was listed earlier. The default value is TAG_SPACE_COUNT_REVERSE_NAME. This default value means that tags are shown first by the tag count, with resources with more tags shown before resources with fewer tags, then, if resources have the same number of tags, alphabetically. This parameter can take the following values:

- TAG_SPACE_NAME
- TAG_SPACE_COUNT
- TAG_SPACE_COUNT_NAME
- TAG_SPACE_COUNT_REVERSE_NAME. This value is the default value.
- TAG_SPACE_CREATION_DATE
- TAG_SPACE_LAST_MODIFIED_DATE

For more information, see the class com.ibm.portal.cp.Constants.OrderMetric in the portal Javadoc.

privateTaggingEnabled = (false)

Use this parameter to specify the PERSONAL_PRIVATE scope for the tags that you want to show in this widget. The default value is false. By this default, users cannot add private tags. If you set this parameter to true, users can also add private tags.

resourceCategories = ["*resrc_category_1*", "*resrc_category_2*", . . . "*resrc_category_n*"]

Use this parameter to specify an array of categories that are assigned to the resource for which the widget was called. Represent each category by a string, for example Books or a Web Content Manager category. A typical value is ["books", "action"].

resourcePrivate = true | false

You can set this parameter to avoid access control issues with private resources. Users can add private tags only to private resources. The default value is false.

resourceType

Use this parameter to specify the portal resource type. Specify either CONTENT_NODE or NAVIGATION_NODE. Specifying a value for this parameter is mandatory for portal resources only.

tagClickActionMode = TAG_CENTER | PUBLIC_RENDER_PARAMETER

Use this parameter to determine what happens when a user clicks a tag. Specify one of the following values:

TAG_CENTER

With this value, the widget redirects the user to the tag center. This value is the default value.

PUBLIC_RENDER_PARAMETER
With this value, the widget exposes a public render parameter with the tag name.

tagClickTransmitScopes = (false)
Use this parameter to specify whether scopes to which a tag belongs, for example categories, are transmitted, when a user clicks a tag. Setting this parameter set to `true` makes sense only if you also set the parameter `com.ibm.wps.cp.tagging.inline.tagsClickable` to `true`.

tagsClickable = true | false
Use this parameter to determine whether the tags can be clicked for redirection or public render parameter exposure. The default is `true`.

tagScope = (COMMUNITY_PERSONAL_PUBLIC)
Use this parameter to specify the scope of tags that you want to show in this widget. Specify one of the following values:

```
PERSONAL_PUBLIC|PERSONAL_PRIVATE|COMMUNITY_PERSONAL_PUBLIC
```

If you do not specify a value, the parameter defaults to `COMMUNITY_PERSONAL_PUBLIC`.

Related concepts:

"The tag widget" on page 1311
Users can use the tag widget to view, apply, and update tags that were applied to a resource.

Related tasks:

"Customizing the tag widget" on page 1314
The user interface of the tag widget consists of Web Content Manager HTML components. You can customize the tag widgets by modifying one or more of these components. For example, you can change the order of the user interface elements, or you can remove a field that you do not want to show in the user interface. The components are listed here.

Related reference:

"Properties for the tag widget" on page 315
View the properties for the tag widget.

Related information:

IBM WebSphere Portal V 8 Product Documentation

Rating widget parameter reference:

You configure rating widget instances by setting the Javascript parameters listed here.

Note: The parameters listed here are specific to the inline rating widget that was introduced with IBM WebSphere Portal Express Version 8.5. For information about the earlier widgets and their parameters, read the appropriate topics in the WebSphere Portal Express Version 8.0 product documentation.

To set the parameters, proceed as follows:

1. In IBM Web Content Manager, go to **Applications** > **Content** > **WCM Authoring** > **Libraries** > **Web Resources v70** > **Components**.
2. Select the check box for **HTML - Rating Widget Light - Stars**.
3. Click **Edit**.

Mandatory parameters

The following parameters are mandatory for the rating widget.

id = unique_ID
Use this attribute to specify a unique ID for parsing all surrounding `<div>` tags to enable your widgets

resourceID
Use this parameter to specify the identifier of the piece of content that will show the widget. This needs to be unique. For example, for a portal page specify the portal object ID of that page.

Optional parameters

The following parameters are optional for the rating widget. They correspond to similar parameters in the WP CP Configuration Service.

customLabel
Use this parameter to specify a non-localized custom label to describe the displayed ratings.

customLabelCommunityRatings
Use this parameter to specify the non-localized label that you want to be used to describe the meaning of the community rating that is displayed.

customLabelPersonalPublicRatings
Use this parameter to specify the non-localized label that you want to be used to describe the meaning of the personal public rating that is displayed.

customLabelPersonalPrivateRatings
Use this parameter to specify the non-localized label that you want to be used to describe the meaning of the personal private rating that is displayed.

deletingEnabled = (true)
Use this parameter to control whether the Delete Rating option is enabled in the inline widget. The default value is `true`.

nonCSSuiEnabled
Set this parameter to the value `true` for to enable rendering on a user interface that is accessible for iOS devices used with a bluetooth keyboard. The default value is `false`.

numStars = 5
Use this parameter to specify the number of stars or asterisks of which a rating consists. Specify a positive numeric value. The default value is `5`. Do not specify a value larger than the value specified for the property `com.ibm.wps.cp.rating.maxratingvalue` in the WP CP configuration service for tagging and rating. For more information about this property, read *General properties for tagging and rating*.

privateRatingEnabled = (false)
The default value of this parameter is `false`. By this default, users cannot add private ratings. For users to be able to add private ratings, set this parameter to `true`.

ratingDescription=NONE
Use this parameter to specify the type of description for community average ratings in the rating widget. Specify one of the following values:

RATING_VALUE
With this value, the rating description includes only the numerical rating value.

TOTAL_NO_RATING
With this value, the rating description includes only the total number of ratings.

ALL With this value, the rating description includes the numerical rating value and the total number of ratings.

NONE
With this value, the rating description is not shown. This is the default value.

If you do not specify a value, the parameter defaults to `NONE`.

ratingScope = (COMMUNITY_PERSONAL_PUBLIC)
Use this parameter to specify the scope of ratings that you want to show in this widget. Specify one of the following values:

```
PERSONAL_PUBLIC|PERSONAL_PRIVATE|COMMUNITY_PERSONAL_PUBLIC
```

If you do not specify a value, the parameter defaults to `COMMUNITY_PERSONAL_PUBLIC`.

resourceCategories = ["*resrc_category_1*", "*resrc_category_2*", . . . "*resrc_category_n*"]
Use this parameter to specify an array of categories assigned to the resource for which the widget was called. Represent each category by a string, for example `Books` or a Web Content Manager category. A typical value is `["books", "action"]`.

resourcePrivate = true | false
You can set this parameter to avoid access control issues with private resources. Users can add private ratings only to private resources. The default value is `false`.

resourceType = NAVIGATION_NODE | CONTENT_NODE
Use this parameter to specify the type of the resource for which the rating widget is called. This parameter is mandatory only for portal resources such as pages or portlets. Valid values are `NAVIGATION_NODE` or `CONTENT_NODE`.

Related concepts:

"The rating widget" on page 1315
Users can use the rating widget to view, apply, and update ratings that were applied to a resource.

Related tasks:

"Customizing the rating widget" on page 1319
The user interface of the rating widget consists of Web Content Manager HTML components. You can customize the rating widgets by modifying one or more of these components. For example, you can change the order of the user interface elements, or you can remove a field that you do not want to show in the user interface. The components are listed here.

Related reference:

"General properties for tagging and rating" on page 313
View the general properties for tagging and rating.

"Properties for the rating widget" on page 317
View the properties for the rating widget.

Related information:

IBM WebSphere Portal V 8 Product Documentation

CSS classes for tagging and rating

The portal tag and rating widgets allow for detailed look and feel customization by providing a customizable CSS class hierarchy.

Each widget has a set of tagging and rating specific CSS classes assigned to it. This allows administrators to customize either the complete set or a subset of the tag and rating widgets, or individual tag and rating widgets. To do this, the administrator modifies the CS class or classes for the widget that needs to be customized. For details refer to the information in the following tables.

Table 172. Available tag and rating widgets and CSS classes assigned to them

Widget name	Tagging and rating CSS class assigned to the widget
`TagCloud`	`trc trcTagCloud`
`AddTag dialog`	`trc trcTagging trcDialog trcDialogTagging`
`AddRating dialog`	`trc trcRating trcDialog trcDialogRating`
`InlineTag`	`trc trcTagging trcInline trcInlineTagging`
`InlineRating`	`trc trcRating trcInline trcInlineRating`

Table 173. Available CSS hierachy groups

Group name	CSS class that needs to be modified to customize the given group of widgets
All tag and rating widgets	`trc`
All tag widgets	`trcTagging`
All rating widgets	`trcRating`
All dialog based widgets	`trcDialog`
All inline widgets	`trcInline`

Table 174. Individual widgets CSS classes

Widget name	CSS class that needs to be modified to customize the given widget
TagCloud widget	`trcTagCloud`
AddTag dialog widget	`trcDialogTagging`
AddRating dialog widget	`trcDialogRating`
InlineTag widget	`trcInlineTagging`
InlineRating widget	`trcInlineRating`

Hierarchy of CSS classes

The following list shows all available tagging and rating CSS classes that can be used for customizing the appearance of tag and rating widgets in the order of

significance. The list shows the CSS classes in the order of their significance from low to high. More significant CSS classes shown further down in the list override definitions of the less significant CSS classes shown further up in the list.

1. `trc`
2. `trcTagging`
3. `trcRating`
4. `trcDialog`
5. `trcDialogTagging`
6. `trcDialogRating`
7. `trcInline`
8. `trcInlineTagging`
9. `trcInlineRating`

Administrators can override each CSS class that is used by the widget by adding customized CSS classes of the following format in file `widgets_combined.css` :

```
.trc <CSS classname to override> {
     // custom CSS definitions for tag and rating widgets in general
}

.trcTagging <CSS classname to override> {
    // custom CSS definitions for all tag widgets
}

.trcTagCloud <CSS classname to override> {
   // custom CSS definitions for the Tag Cloud widget only
}
```

Customizing tagging and rating specific CSS classes

The CSS classes used to customize the visual appearance of the tag and rating widgets are located in the following file: *PortalServer_root*`/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/modules/portalclient/css/trcWidgets.css` .

This file contains CSS classes specific to tagging and rating. They either start with the prefix `trc`, for example `trc Editable` or `trc Delete` or they are otherwise scoped to the tagging and rating context, for example `ul.trcEdit` . To customize the visual appearance of tag and rating widgets, you override these definitions in the CSS file. To do this, you add class definitions at the end of the file `trcWidgets.css` based on the CSS class hierarchy given by the Hierarchy of CSS classes list.

Examples: To change the link color for all tag and rating widgets to green, use the following CSS class declaration in the file `trcWidgets.css` :

```
.trc a {
    color: green;
}
```

If you do this, all tag and rating widgets show their links in a green color.

To change only the link color for the Tag Cloud to red, use the following CSS class declaration:

```
.trcTagCloud a {
    color: red;
}
```

A more specific declaration overrides a less specific declaration. Therefore you can have both declarations in the file `trcWidgets.css` at the same time. While the Tag Cloud shows its links in red, all other tag and rating widgets continue to show their links in green.

The following example shows how you can modify link appearance within all the available tag and rating widgets. It shows how to inherit from less significant CSS class definitions and how to overwrite inherited definitions to further customize a specific widget instance:

```
/* Give all links within tag and rating widgets a default background */
.trc a {
   background-color: #d0efff;
}

/* Make tagcloud links appear in a black color */
.trcTagCloud a {
  color: black;
}

/* Give all tag widgets a blue link color */
.trcTagging a {
  color: blue;
}

/* Give all rating widgets a green link color */
.trcRating a {
  color: green;
}

/* Links in all dialog widgets appear in italic,
   colors are inherited from trcRating/trcTagging */
.trcDialog a {
  font-style: italic;
}

/* Links in all inline widgets appear as underlined,
   colors are inherited from trcRating/trcTagging */
.trcInline a {
  font-style: underline;
}

/* links in Tag Widget Dialog inherit italic font-style from .trcDialog,
   but overwrite the color inherited from .trcTagging with another color */
.trcDialogTagging a {
  color: #770000;
}

/* links in Rating Widget Dialog keep the inherited link color defined by
   .trcRating, but change the inherited font-style, by setting it to none */
.trcDialogRating a {
  font-style: none;
}

/* modify the inherited color (from .trcTagging) and reset the font-style (from
   .trcDialog) to none. */
.trcInlineTagging a {
  color: green;
  font-style: none;
}

/* no changes for the inline rating widget. It simply uses the color and font-style
   from the super classes .trcRating and .trcInline instead */
.trcInlineRating a {
}
```

Enabling and disabling the tag and rating widgets for additional profiles

In a portal installation, the tag and rating widgets are available for specific portal profiles. You can enable the widgets for more other profiles by adding them to the profile.

About this task

In a default IBM WebSphere Portal Express V 8.5 installation, the tag and rating widgets are available with the following profiles:

- Basic Content profile: `profile_basic_content.json`
- Basic Content with Dojo profile: `profile_dojo_basic_content.json`
- Content Targeting Portlet profile: `profile_personalization.json`.

To enable the tag and rating widgets for other profiles, proceed as follows:

Procedure

1. Open the profile for which you want to enable the widgets.
2. Add the module `wp_tagging_rating_light` to the list of modules in the section `moduleIDs`.
3. Apply the modified profile. To do so, proceed as follows:
   a. Start a WebDAV client.
   b. Create a connection with the entry point to `/wps/mycontenthandler/dav/fs-type1` of WebSphere Portal Express Version 8.5.
   c. Go to **Themes** > **Portal 8.5** > **Profiles**.
   d. Upload the modified profile by using the **Upload** option in the WebDAV client.
   e. Log in to the portal server.
   f. Invalidate the cache with the Portal Theme Analyzer. Or, click **Administration** > **Utilities** > **Control Center** > **Invalidate cache**. CF06 Auto invalidation recognizes your changes automatically for WebDAV based themes. No further action is required. For more information, see Utilities.

   This step applies the changes that you made to the profile on the server.

Results

You can now use the tag and rating widgets with the modified profile.

Notes:

- The tag and rating widgets are available with the Basic Content profile. If you create a page by using the Article template, the portal applies the Basic Content profile by default. If you add a blog or a wiki to a page with a different profile and you want to have the tag and rating widgets that are shown, you must apply the Basic Content profile to the page.
- Enabling the widgets for a profile can influence performance.

To disable the widgets, remove the module `wp_tagging_rating_light` from the profile, and apply the modified profile on the server by the described procedure.

Enabling and disabling the Dojo tagging and rating options for additional profiles

In a portal installation, the Dojo tagging and rating menu options for portal pages and portlets are available for a specific portal profile. You can enable these options for other profiles by adding them to the profile.

About this task

In a IBM WebSphere Portal Express V 8.5 installation, the Dojo tagging and rating menu options for portal pages and portlets are available for the Search and Tag Center profile. This profile is named `profile_search_tag.json`. You can apply this profile to a page or portlet where Dojo tagging and rating options are required. After you apply this profile, the page action menu or portlet menu shows the tagging and rating options.

Notes:

- If you upgraded your WebSphere Portal Express from version 8.0 to 8.5, you also need to update the page to which you add the tagging and rating options with the new Portal V 8.5 theme. For more information, read *Post-migration steps for Tag and Search enter pages*.
- Enabling the Dojo tagging and rating menu options for a profile can influence performance.

To enable the Dojo tagging and rating options for other profiles, proceed as follows:

Procedure

1. Open the profile for which you want to enable the modules.
2. Add the module `wp_tagging_rating_menu` to the list of modules in the section `moduleIDs`.
3. Apply the modified profile. To do so, proceed as follows:
   a. Start a WebDAV client.
   b. Create a connection with the entry point to `/wps/mycontenthandler/dav/fs-type1` of WebSphere Portal Express V 8.5.
   c. Go to **Themes** > **Portal 8.5** > **Profiles**.
   d. Upload the modified profile by using the **Upload** option in the WebDAV client.
   e. Log in to the portal server.
   f. Invalidate the cache with the Portal Theme Analyzer. Or, click **Administration** > **Utilities** > **Control Center** > **Invalidate cache**. CF06 Auto invalidation recognizes your changes automatically for WebDAV based themes. No further action is required. For more information, see Utilities.

   This step applies the changes that you made to the profile on the server.

Results

You can now use the Dojo tagging and rating menu options with the modified profile. To disable the Dojo options for a profile, remove the module `wp_tagging_rating_menu` from the profile, and apply the modified profile on the server by the described procedure.

Related tasks:

"Tag and Search Center pages" on page 887
When you migrate to IBM WebSphere Portal Express Version 8.5, the migration process does not apply the Portal 8.5 theme to all portal pages. For example, this affects the Tag and Search Center pages. To continue to use your Tag Center and Search Center pages, you must update the theme for the pages to the Version 8.5 theme. You must also update the profile of the pages to the Search and Tag Center profile.

Security for tagging and rating

For administering which users can tag and rate content, the portal provides virtual resources for tagging and rating and roles on these virtual resources.

The portal provides the following virtual resources: `TAGS` and `RATINGS`. These resources allow you to determine user rights that related to tagging and rating. The following list explains which roles users require to perform tagging and rating operations. The user actions correspond to the normal portal roles. Privileges are inherited.

In a default portal installation, the group All Authenticated Portal Users has the CONTRIBUTOR role. There is no role assigned to the anonymous user. Please note that anonymous users need at least the CONTRIBUTOR role to be able to apply tags and ratings as they can only apply public tags or ratings, not private ones.

USER

Can view community tags and community ratings that other users have applied and all personal tags, both personal private and personal public.

PRIVILEGED USER

Includes USER actions.

Can view community tags and community ratings that other users have applied.

Can create and delete personal private tags and ratings.

CONTRIBUTOR

Includes USER actions.

Can view community tags and community ratings that other users have applied.

Can create and delete personal public tags and ratings, but cannot create or delete personal private tags and ratings.

MANAGER

Includes USER and CONTRIBUTOR actions.

Can view community tags and community ratings that other users have applied.

Can create and delete personal public tags and ratings.

Can delete community tags regardless of ownership.

Note:

- In a default portal installation, the group All authenticated users has CONTRIBUTOR and privileged USER access to tags and rating under virtual resources. If you want to test what an authenticated user with the default role assignments in the portal can do with tags and ratings, remove these permissions first.

- In a default portal installation, anonymous users have no role assigned. For anonymous users to create and delete personal public tags and ratings, assign them the CONTRIBUTOR role.

Related concepts:

"Access permissions" on page 1537
Learn about sensitive operations for resources and the roles that are required to perform those operations. Sensitive operations include common tasks such as viewing portlets on specific pages and complex, high-risk tasks like running XML configuration interface scripts.

Using the XML configuration interface to administer tags and ratings

You can use the XML configuration interface to manage tagging and rating in the portal. For example, you can move tagspaces and ratings between portal versions or for staging purposes.

The XML resources related to tagging and rating are `tag`, `rating`, and `custom-resource`. Portal resources and custom resources are tagged and rated by different ways:

- To tag or rate **portal resources**, you use their object IDs directly.
- To represent **custom resources**, for example books, you use the XML resource `custom-resource`. The object ID of the `custom-resource` resource is used as the `resourceref` attribute in the `tag` and `rating` tags.

The XML resource tags and their attributes are listed in the following.

Notes:

1. When you create tags, ratings, or custom resources, you need to specify all attributes except the ones marked as optional.
2. When you move tagspaces between portals, both the users who have applied the tags and the resources to which the tags have been applied must exist in the target portal.
3. You can update existing ratings, but not existing tags by using the XML configuration interface. The XML configuration interface `action="update"` works only for implicitly creating a new tag.

tag Use the following attributes with the XML resource tag `tag` :

resourceref = "*object_ID*"
This attribute specifies the reference to the resource that is being tagged.

domain = "comm | cust"
This attribute specifies the database domain for the tagged resource. Possible values are:

cust Specify this value for private tags.

comm Specify this value for public tags.

locale This attribute specifies the locale of the tag. This attribute is optional. The default is `null`.

owner = "*user*"
This attribute specifies the owner of the tag.

rating Use the following attributes with the XML resource tag `rating` :

resourceref = "*object_ID*"
This attribute specifies the reference to the resource that is being rated.

domain = "comm | cust"
This attribute specifies the database domain for the rated resource. Possible values are:

cust Specify this value for private ratings.

comm Specify this value for public ratings.

value = "*integer*"
This attribute specifies the rating value.

owner = "*user*"
This attribute specifies the owner of the rating.

custom-resource
Use this tag to represent custom resources, for example books. Use the following attribute with the XML resource tag `custom-resource` :

uri = "*string*"
This attribute specifies a URI that identifies the custom resource.

Use the following subtag with the XML resource tag `custom-resource` :

category-instance
Use the subtag `category-instance` to assign a category in the format of a string to a custom resource. You can assign several categories to a custom resource. Use the following attribute with the subtag `category-instance` :

name = "*category_instance_name*"
Use this attribute to specify a name for a category instance.

Refer to the following code samples.

Example: Exporting tags and ratings

```
<?xml version="1.0" encoding="UTF-8" ?>
<request
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
    type="export">

    <!-- This sample exports all custom resources, ratings, and tags.
         Related sample files:
             CreateTagsAndRatings.xml
             DeleteTagsAndRatings.xml
     -->
    <portal action="locate">

        <custom-resource action="export" objectid="*"/>
        <rating action="export" objectid="*"/>
        <tag action="export" objectid="*"/>

        <!-- Export all tags with a specific locale in the system -->
        <!-- <tag action="export" objectid="*" locale="SPECIFIC_LOCALE"/> -->

    </portal>
</request>
```

Example: Creating tags and ratings

```
<?xml version="1.0" encoding="UTF-8"?>
<request
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
    type="update">

    <!-- This sample creates ratings and tags.

         Related sample files:
             ExportTagsAndRatings.xml
             DeleteTagsAndRatings.xml


         NOTE: This sample file needs to be modified before execution.
               Update the value of the 'owner' attributes of the 'access-control',
               'rating', and 'tag' tags, and specify an existing user.
    -->

    <portal action="locate">

        <!-- Parent element under which a new page for this sample is inserted -->
        <content-node action="locate" objectid="parentPage" uniquename="ibm.portal.Home"/>

        <!-- A new empty page to which a tag and rating are assigned.-->
        <content-node action="update" objectid="samplePageOID"
                      uniquename="ibm.portal.SamplePage.TagsAndRatings"
                      ordinal="last" content-parentref="parentPage"
                      active="true" create-type="explicit" type="page">
            <supported-markup markup="html" update="set"/>
            <localedata locale="en">
                <title>Sample page for tag and rating creation</title>
            </localedata>
        </content-node>


        <!-- A custom resource can be used to assign tags and ratings to resources
             that are not managed by XMLAccess, but can be identified by an URI -->
        <custom-resource action="update" objectid="CH_B1L68B1A00DO80IG7PCV0I1000"
                         uri="book:mySampleBookURI">
            <category-instance action="update" name="cookbook"/>
            <category-instance action="update" name="hardcover"/>
        </custom-resource>

        <!-- Assignment of a rating value of 5 by user wpsadmin to the sample page -->
        <rating action="update" objectid="CJ_B1L68B1A00DO80IG7PCV0I2000"
                resourceref="samplePageOID" domain="comm" value="5"
                owner="uid=wpsadmin,o=defaultwimfilebasedrealm" />
        <!-- Assignment of a rating value of 5 to the custom resource -->
        <rating action="update" objectid="CJ_B1L68B1A00DO80IG7PCV0I3000"
                resourceref="CH_B1L68B1A00DO80IG7PCV0I1000" domain="comm" value="5"
                owner="uid=wpsadmin,o=defaultwimfilebasedrealm"/>

        <!-- Assignment of the tag 'sample' to the sample page -->
        <tag action="update" objectid="CI_B1L68B1A00DO80IG7PCV0I4000"
             resourceref="samplePageOID" domain="comm"
             owner="uid=wpsadmin,o=defaultwimfilebasedrealm" locale="en">sample</tag>
        <!-- Assignment of the tag 'sample' to the custom resource -->
        <tag action="update" objectid="CI_B1L68B1A00DO80IG7PCV0I5000"
             resourceref="CH_B1L68B1A00DO80IG7PCV0I1000" domain="comm"
             owner="uid=wpsadmin,o=defaultwimfilebasedrealm" locale="en">sample</tag>

    </portal>
</request>
```

Example: Deleting tags and ratings

```
<?xml version="1.0" encoding="UTF-8"?>
<request
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
    type="update">

    <!-- This sample deletes ratings and tags.

         Related sample files:
             CreateTagsAndRatings.xml
             ExportTagsAndRatings.xml
```

```
        NOTE: This sample assumes that the CreateTagsAndRatings.xml sample was
        executed before.
     -->
    <portal action="locate">

        <!-- Delete the custom resource created by sample CreateTagsAndRatings.xml -->
        <custom-resource action="delete" objectid="CH_B1L68B1A00DO80IG7PCV0I1000"/>

        <!-- Delete all custom resources in the system -->
        <!-- <custom-resource action="delete" objectid="*"/> -->

        <!-- Delete the ratings created by sample CreateTagsAndRatings.xml -->
        <rating action="delete" objectid="CJ_B1L68B1A00DO80IG7PCV0I2000"/>
        <rating action="delete" objectid="CJ_B1L68B1A00DO80IG7PCV0I3000"/>

        <!-- Delete all ratings in the system -->
        <!--  <rating action="delete" objectid="*"/> -->

        <!-- Delete the tags created by sample CreateTagsAndRatings.xml -->
        <tag action="delete" objectid="CI_B1L68B1A00DO80IG7PCV0I4000"/>
        <tag action="delete" objectid="CI_B1L68B1A00DO80IG7PCV0I5000"/>

        <!-- Delete all tags in the system -->
        <!-- <tag action="delete" objectid="*"/> -->

        <!-- Delete all tags with a specific locale in the system -->
        <!-- <tag action="delete" objectid="*" locale="SPECIFIC_LOCALE"/> -->

    </portal>
</request>
```

Moving tags and ratings between portals by using the XML configuration interface

To move tags and ratings between portals, for example for staging purposes, proceed as follows:

1. Make sure that all users who have applied tags and ratings on the source portal also exist on the target portal.
2. Make sure that all tagged and rated resources on the source portal also exist on the target portal.
3. Use the provided XML sample script to export all tags and ratings from the source portal.
4. Import the result file from the previous export step to the target portal.

Related reference:

"Types of portal resources" on page 1085
The portal resources are represented by the following XML tags.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Hints and tips for tagging and rating

Learn about some hints and tips that apply to tagging and rating. Some hints and tips might help developers and portal administrators, others might help portal users.

"Hints and tips for developers and portal administrators" on page 1347
Learn about some hints and tips for administrators who work with tagging and rating.

"Hints and tips for portal users" on page 1350
Learn about some hints and tips for portal users who work with tagging and rating.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Related reference:

"CP Configuration Service for tagging and rating" on page 312
The CP Configuration Service provides the properties for tagging and rating.

"Parameter reference for the tag and rating widgets" on page 1330
You can configure each of the portal tagging and rating features to determine the look and functionality of these features. To do so, you configure the tag and rating widgets.

Hints and tips for developers and portal administrators

Learn about some hints and tips for administrators who work with tagging and rating.

Using tagging and rating on a virtual portal

To use tagging and rating with your virtual portal, ensure that the **web resource v7.0** and **web content templates 3.0** libraries exist on the virtual portal.

Administrators can assign access roles to users for tagging and rating content.

The portal user roles give users the following rights:

USER The user can view tags and ratings that other users applied.

PRIVILEGED USER
The user can also apply private tags and ratings.

CONTRIBUTOR
The user can also apply public tags and ratings.

MANAGER
The user can also modify tags and ratings that other users applied.

Users can work only with tagging and rating according to their access rights on portal resources.

Examples:

- Users can tag and rate only portal resources that they can access.
- When users click a tag in the tag cloud, they can view only resources that they can access.

Limitation: There is not security handling for tag clouds. When a user clicks a tag, the resources are filtered by the user's access rights and then listed for the user to view. Users might see tags even if they have no access rights on the tagged resources. When they click the tag, the list is empty.

Tags in tag clouds do not reflect access rights.

The tags that are shown in tag clouds do not reflect the access rights that users have on the tagged resources. Resources are filtered by access rights only after the user clicks a tag. This action has the following consequences:

Users might see tags without underlying resources.
Users might see tags that are applied to resources to which the users have no access. If they click a tag that is applied only to resources to which they have no access, they view an empty resource list.

Tag size does not represent frequency of the tag on resources that a user can access.
When a user views a tag cloud, tag sizes in the tag cloud represent how often the tag is applied. It does not represent how often the tag is applied to resources that the user can access.

Only portal content can be pre-tagged.
You can pre-tag only portal resources, such as pages and portlets. You cannot pre-tag custom content or Web Content Manager content.

Tags in a virtual portal can be seen only in that virtual portal.
If your portal contains virtual portals, the tags and ratings are limited to the virtual portal in which they were created. It is not possible to share tags across several virtual portals. Tags that are created in a virtual portal cannot be seen anywhere else.

Tagspace cleanliness.
The portal implementation of tagging prevents tag space littering that is the inclusion of tags that do not contribute to categorizing content. It manifests mostly in the following two issues:

- Redundant tags that result from similar names or spelling variants, for example, `web20` and `Web 2.0`. The portal reduces such duplicate tags by a syntactical type-ahead feature. This feature helps reduce tag space littering by suggesting tag name variants that other users already use. For example, if a user types `web2` and other users already use `Web 2.0` to tag portal resources, the type-ahead feature suggests `Web 2.0`, although `web2` is not an exact partial string of the suggested tag name. It supports users to not use too many different variants of the same tag name when you tag portal resources.

  Note: The type-ahead feature works only with the dialog tag widget of the default tagging user interface of portal versions earlier than V 8.5. With WebSphere Portal Express V 8.5, the tag and rating widgets of earlier portal versions are deprecated.
- Tags pointing to resources that are deleted. You can use the portal administrative cleanup tool, SLChecker. Use it to check for invalid links that do not exist any more and for tags that someone applied who is no longer a portal user. The SLChecker tool can identify and delete tags for deleted resources and tags that are applied by deleted users. For information, read *Deleting orphaned data*.

Staging and migrating tagspaces.
You can transfer the tag space from one portal to another, for example for staging or for migrating from one portal version to another. To do so, use the XML configuration interface (XML Access). For more information, read *Using the XML configuration interface to administer tags and ratings*.

Note: Users can create private and public tags. Private tags are stored in the customization database, whereas public tags are stored in the community database. Only content of the community database is staged and migrated.

Tagging custom content requires UI development.
If you want your users to tag and rate custom content, you must write code. The code must allow customers to find this content with the public APIs. You must also add the resources that you want your users to tag to the portal. Custom content is anything apart from portlets, portal pages, and Web Content Manager resources. For information, read *Enabling your own custom content to be tagged and rated*.

Tag filtering.
You can apply tag filters to suppress words that you do not want to be used as tags in your portal. As you can have more multiple active filters, a tag must pass all filters to be applied. A simple filter is shipped with the portal. If you need more advanced filtering, you must provide your own filters. For more information about creating custom filters, read *Filtering content for tagging*.

SQL Server URI length limit for custom resources.
For SQL Server , tags for custom resources with URIs of which the scheme-specific part in UTF-8 is larger than 850 bytes cannot be stored. As a workaround you can either use URIs with shorter scheme-specific parts or drop the respective index in the database table. Because of this limitation, a warning about the index key length is written when you transfer the database for SQL Server .

Tag Cloud, Tag Center, and Results List portlets do not support WSRP.
The Tag Cloud and Tag Center portlets, including the Result List portlet, do not support WSRP. It is not possible for a Producer portal to provide these portlets as remote web services. It is not possible for a Consumer portal to consume them so that its users can use them.

Maximum number of available tags.
By default the tag widget can show up to 50 different tag names. This limit applies separately to each kind of tags, **community** tags, **personal public**, and **personal private** tags. You can configure these maximum values by using the following two properties:

- For community tags: `com.ibm.wps.cp.tagging.dialog.maxCommunityTags`
- For personal tags: `com.ibm.wps.cp.tagging.dialog.maxPersonalTags`.

Increasing the 50 maximum figures can lead to slower responses of the user interface of the tag widget. The number of tags per resource averages 5 - 8. Users tend to reuse tag names that other users already used and that the tag widget then suggests by the type-ahead feature.

You configure these two properties globally in the CP Configuration Service for tagging and rating or for individual widgets in the widget properties. For details, read *CP Configuration Service for tagging and rating* and *Setting service configuration properties*.

Note: The type-ahead feature works only with the dialog tag widget of the default tagging user interface of portal versions earlier than V 8.5. With WebSphere Portal Express V 8.5, the tag and rating widgets of earlier portal versions are deprecated.

Angled brackets are not allowed in tag names.
Angled brackets (< and >) are not allowed within tag names. Therefore, no matter how you customize the regular expression, angled brackets are not accepted.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

"Deleting orphaned data" on page 280
You use the SLCheckerTool to delete orphaned data in the database.

Related reference:

"CP Configuration Service for tagging and rating" on page 312
The CP Configuration Service provides the properties for tagging and rating.

"Using the XML configuration interface to administer tags and ratings" on page 1343
You can use the XML configuration interface to manage tagging and rating in the portal. For example, you can move tagspaces and ratings between portal versions or for staging purposes.

"Enabling your own custom content for tagging and rating" on page 1324
Enabling your own custom content for tagging and rating works only with the dialog widgets of earlier portal versions.

"Filtering content for tagging" on page 1307
You can use filtering mechanisms to control which terms users can use and which terms they cannot use as tags. The portal provides both a blacklist and a whitelist filter.

Hints and tips for portal users

Learn about some hints and tips for portal users who work with tagging and rating.

Tag names must contain at least one alphanumeric character.
Tag names that consist only of non-alphanumeric characters are not allowed. Example: `"!^#"` .

Tags in tag clouds do not reflect access rights.
The tags that are shown in tag clouds do not reflect the access rights that users have on the tagged resources. Resources are filtered by access rights only after the user clicks a tag. This action has the following consequences:

Users might see tags without underlying resources.
Users might see tags that are applied to resources to which the users have no access. If they click a tag that is applied only to resources to which they have no access, they view an empty resource list.

Tag size does not represent frequency of the tag on resources that a user can access.
When a user views a tag cloud, tag sizes in the tag cloud represent how often the tag is applied. It does not represent how often the tag is applied to resources that the user can access.

How can I avoid closing the widget when clicking on listed resource links?
Rightclick the link that you want to view and select **Open in New Tab**. The widget will remain open, and you can proceed with the other links in the same way.

Type-ahead support requires at least 3 characters
When a user types a tag, the user needs to type at least 3 characters for the type-ahead support feature to work.

Note: The type-ahead feature works only with the dialog tag widget of the default tagging user interface of portal versions earlier than V 8.5. With WebSphere Portal Express V 8.5, the tag and rating widgets of earlier portal versions are deprecated.

Using WebDAV with WebSphere Portal Express

WebSphere Portal Express provides a Web-based Distributed Authoring and Versioning (WebDAV) implementation that individual services can use by plugging into. WebDAV is a set of extensions to the HTTP protocol that allows you to collaborate on editing and managing files on remote Web servers. Caching Proxy supports WebDAV methods used by Microsoft Exchange Server, and user-defined (customized) methods. These methods are hard coded and managed by the Enable and Disable directives. Administrators can also use the corresponding method-mask defined in the PROTECT directive to authorize the use of these methods.

About this task

Examples of such services are:

- WebDAV for managing pages and static content.
- WebDAV filestore. For example, this is used by portal themes.
- WebDAV for Web Content Manager.

Note: WebDAV does not support secure connections, such as, HTTPS. For specific WebDAV client version compatability, refer to the detailed systems requirements documentation and the Tech Note about *WebDAV clients for accessing IBM WebSphere Portal Express*.

"Configuring the WebDAV file store" on page 1352
By default only administrative users can perform write operations on specific folders of the WebDAV file store. This affects public and user owned folders. You can enable write access for all authenticated users on WebDAV file stores folders.

"Using WebDAV file store" on page 1352
You can use WebDAV to work with the portal themes.

"Serving HTTP OPTIONS requests to the server context root by WebDAV clients" on page 1358
Some WebDAV clients send an HTTP OPTIONS request to the server context root (`/`) to check whether the server supports WebDAV. To support these clients, the portal provides a web application called `wp.webdav.options.war` that you can enable. This application responds to such requests with a confirmation that the portal supports WebDAV.

"Working with WebDAV clients" on page 1358
WebDAV is an HTTP extension framework with a plug point for the access and management of hierarchical data, for example, in content management systems. WebDAV stores the data in collections and allows you to work with the data in a user interface view that is similar to that of a file system. Various tools are available for integrating WebDAV resources into the client file system, known as WebDAV clients. To use WebDAV for WebSphere Portal Express, you must first set up your WebDAV clients.

"Task `webdav-deploy-zip-file`" on page 1359
Use this configuration task to manage theme artifacts and to deploy iWidgets. This task uploads archive or compressed files to portal WebDAV folders.

Related concepts:

"Virtual portals" on page 1361
View information on how you can scope your WebSphere Portal Express to have multiple virtual portals.

Related tasks:

"Using WebDAV to manage pages and static content" on page 1877
WebDAV for IBM WebSphere Portal Express provides a simple and easy way to administer portal resources. Both administrators and users can use it.

"WebDAV" on page 1986
With WebDAV for IBM WebSphere Portal Express, you can use standard operating system tools to create, modify, and delete web content rather than the standard authoring portlet.

Related reference:

"Using WebDAV file store"
You can use WebDAV to work with the portal themes.

Related information:

Detailed system requirements

WebDAV clients for accessing IBM WebSphere Portal

WebDAV with web content

Configuring the WebDAV file store

By default only administrative users can perform write operations on specific folders of the WebDAV file store. This affects public and user owned folders. You can enable write access for all authenticated users on WebDAV file stores folders.

Procedure

1. Add the following property to the WP ConfigService resource environment provider in the WebSphere Integrated Solutions Console: `filestore.writeaccess.allowed`.
2. Set the value for the property to `true` .
3. Restart the portal server for the new setting to take effect.

What to do next

Notes:

- There are several WebDAV entry points. However, the property **`filestore.writeaccess.allowed`** applies to the **`filestore`** entry points for home directories for each user located at: `http://server_name:WC_default_host/wps/mycontenthandler/dav/fs-type1/users/user_name`
- Users can modify only their own directories.

Using WebDAV file store

You can use WebDAV to work with the portal themes.

WebDAV overview and entry point URL

WebDAV is defined by RFC2518 as an HTTP extension framework with a plug point for the access and management of hierarchical data. For example, in content management systems. WebDAV stores the data in collections. You can work with the data in a user interface view that is similar to that of a file system. A folder

represents a WebDAV collection. Various tools are available for integrating WebDAV resources into the client file system. Users can use these tools to view and modify resources that they can access with WebDAV. For WebDAV specification information, see the RFC2518 document in the related links. For more information about WebDAV in the portal, see the topics about Using WebDAV with WebSphere Portal Express.

Note: The HTTP Basic Authentication Trust Association Interceptor (TAI) must be enabled to use WebDAV in WebSphere Portal Express. This TAI is enabled by default. See the related links for information.

You can obtain the entry point URL to the WebDAV file store from the service document under the URL `/wps/mycontenthandler/!ut/p/model/service-document`. The service document contains the top-level access point as follows:

```
<app:collection href="/webdav/!ut/p/dav/fs-type1/">
     <atom:title>fs-type1</atom:title>
     <app:categories fixed="yes">
          . . . . .
          <atom:category term="webdav"/>
          <atom:category term="filestore"/>
          . . . . .
     </app:categories>
</app:collection>
```

The entry point URL for themes is as follows:

`http://`*`server`*`:`*`port`*`/`*`PortalServer_root`*`/mycontenthandler/dav/fs-type1/`

Examples of URLs for themes are as follows:

`http://www.my_company.com:10027/wps/mycontenthandler/dav/fs-type1/`

- For theme-related resources:

  `http://my_company.com:10027/wps/mycontenthandler/dav/fs-type1/themes/`
- For skin-related resources:

  `http://my_company.com:10027/wps/mycontenthandler/dav/fs-type1/skins/`

If you want to authenticate against a specific virtual portal, you can identify the target virtual portal either by its host name or its URL context. Examples:

- To authenticate to the virtual portal identified by the host name `vp.mycompany.com` and then connect to the themes folder, use the following URL entry point:

  `http://vp.mycompany.com:10027/wps/mycontenthandler/dav/fs-type1/themes/`
- To authenticate to the virtual portal identified by the URL context `vp1` and then connect to the themes folder, use the following URL entry point:

  `http://localhost:10027/wps/mycontenthandler/vp1/!ut/p/dav/fs-type1/themes/`

For details about the WebDAV specification, see the RFC2518 document in the related links.

Folder structure and reserved folder names

The fs-type1 WebDAV entry points provide the following set of predefined root folders that are used by themes:

- `themes`
- `skins`
- `layout-templates`
- `common-resources`

- `iwidgets`

The fs-type1 WebDAV entry points also provide the following set of predefined root folders. You can access them by using the Remote Model function that is provided by the Enabler API:

- `public`
- `users`

The fs-type1 WebDAV entry points provide the following internal folder:

- `system`

None of the folders that are listed here can be deleted, not even by an administrator.

The two sets of folders differ in the access control policy that guards access to the resources contained in those folders. See the following sections for details.

Theme folders

The following list shows the folder structure for the themes. Each folder represents a WebDAV collection. You administer write access to the theme folders with the virtual resource THEME MANAGEMENT provided by portal access control.

/themes
: Use this folder to store resources that are associated to themes, such as theme templates. Typically, each subfolder represents one theme.

/skins
: Use this folder to store global skins. Typically, each subfolder of this folder represents one global skin.

/layout-templates
: Use this folder to store templates for layouts that can be used by individual themes. Typically, each subfolder represents one layout template.

/common-resources
: Use this folder to share common resources between different themes, so that they can be managed in a single place.

/iwidgets
: Use this folder to place widgets into it.

WebDAV prevents the deletion of these folders. Even users with administrator rights cannot delete these folders and the data in them.

Managing access control for Page Builder theme folders

- All users have view access to all resources in these folders. It includes both anonymous users and authenticated users.

  To give users write access to resources contained in these folders, assign the users MANAGER role on the virtual resource THEME MANAGEMENT in portal access control. Managers can create, modify, or delete such resources

Assigning access control to users and managers for theme resources in WebDAV

To allow non-administrator users to update or modify existing files do the following steps.

1. Open a command prompt and change to the *wp_profile_root*/ConfigEngine directory.
2. Run the following **ConfigEngine** task.
   - Linux :
   - IBM i: `ConfigEngine.sh export-nodes -DWasPassword=wpsadmin -DPortalAdminPwd=wpsadmin -Dquery="/filestore/fs-type1/themes" -Dwp.content.repository.output.dir="c:\temp\jcr"`
   - Windows: `ConfigEngine.bat export-nodes -DWasPassword=wpsadmin -DPortalAdminPwd=wpsadmin -Dquery="/filestore/fs-type1/themes" -Dwp.content.repository.output.dir="c:\temp\jcr"`
3. Edit the file that was exported in the `c:\temp\jcr` directory. Add the manager and user role by adding the following code.

   ```
   <icm:role icm:actions="actionset:Manager,actions:Traverse,View,Edit,Add_Child,Delete,Join,">
       <icm:principal icm:name="uid=testuser,o=defaultwimfilebasedrealm" icm:type="USER" />
     </icm:role>
   ```

   Add it after the `<icm:owner>` element. See the following complete code snippet for reference.

   ```
   <icm:node>
   <icm:access>
     <icm:wps>
      <icm:owner>
       <icm:principal icm:name="uid=wpsadmin,o=defaultwimfilebasedrealm" icm:type="USER" />
      </icm:owner>
      <icm:role icm:actions="actionset:Manager,actions:Traverse,View,Edit,Add_Child,Delete,Join,"
       <icm:principal icm:name="uid=testuser,o=defaultwimfilebasedrealm" icm:type="USER" />
      </icm:role>
     </icm:wps>
   </icm:access>
   </icm:node>
   ```

4. Import the file with the following **ConfigEngine** task.
   - Linux :
   - IBM i: `ConfigEngine.sh import-nodes -DWasPassword=wpsadmin -DPortalAdminPwd=wpsadmin -Dwp.content.repository.input.dir="c:\temp\jcr"`
   - Windows: `ConfigEngine.bat import-nodes -DWasPassword=wpsadmin -DPortalAdminPwd=wpsadmin -Dwp.content.repository.input.dir="c:\temp\jcr"`

Other folders

The following list shows extra folders. Each of these folders represents a WebDAV collection. The access control policy for them is hardcoded as described for each folder.

/public
: All authenticated users have read and write access to this folder.

  Anonymous users have read access only.

/users
: All authenticated users have read access only to this folder.

Anonymous users have read access only.

/users/*user_name*

Only the user *user_name* has access to these files. This folder is created for the individual user *user_name* when the user accesses the WebDAV file store for the first time.

Note: To have human readable folder names, the portal uses the user IDs of the individual users as the names for the users' folders *user_name*. Internally, the portal uses the VMM ID of the user, so data does not need to be moved when the user name is changed.
If you want to programmatically find the URL entry point to a folder for the current user, you can look into the services document. The access point for user-specific data is provided as follows:

```
<app:collection href="/webdav/!ut/p/dav/fs-type1/users/<username>">
     <atom:title>fs-type1-user</atom:title>
     <app:categories fixed="yes">
          . . . . .
          <atom:category term="webdav"/>
          <atom:category term="filestore"/>
          <atom:category term="user"/>
          . . . . .
     </app:categories>
</app:collection>
```

/users/*user_name*/public

The user *user_name* has read and write access to this folder. This folder contains content that the user *user_name* shared with other users. Portal access control mapping: inherited.

All authenticated users have read access to this folder.

Anonymous users have read access to this folder.

All other subfolders of `/users/`*user_name* can only the accessed by the user *user_name*.

/system

The `system` folder is reserved for system internal information. Administrators can view this folder in WebDAV. Other portal users cannot view this folder. Portal access control mapping: None.

File store cache control

The WebDAV file store supports serving timeout values for HTTP Cache Header entries.

You can use regular expressions to specify the timeout value for elements in the file store folder structure that match the regular expression. You need to add the following two custom properties to the WP Config Service resource environment provider with the following key = value properties:

```
filestore.cache.expiration.id.re=regular expression
filestore.cache.expiration.id.seconds=value
```

The id value can consist of an arbitrary string. It is used only to establish the mapping between a regular expression and its associated timeout value. If there are multiple regular expressions that match any file store resource, the maximum of the associated timeout values are used. Examples:

1. All items under the `/themes` folder have an expiration time of 1800 seconds:

```
filestore.cache.expiration.0.re=themes/.*
filestore.cache.expiration.0.seconds=1800
```

2. All items of a certain resource type, such as jpg or gif have an expiration time of 6000 seconds:

```
filestore.cache.expiration.1.re=.*\.jpg|.*\.gif
filestore.cache.expiration.1.seconds=6000
```

   All css files in the themes folder have an expiration time of 8000 seconds:

```
filestore.cache.expiration.2.re=themes/.*\.css
filestore.cache.expiration.2.seconds=8000
```

Supported HTTP methods

WebDAV file store supports the following HTTP methods:

PROPFIND
: This method allows portal users to find out details about the resource hierarchy, such as the WebDAV collection structure. Users can also find details about resources, such as their names, sizes, and dates of last modification.

MKCOL
: This method allows users to create new folders, that are WebDAV collections.

GET This method allows users to retrieve resources on which they have at least View role access rights.

HEAD
: This method allows users to retrieve HTTP headers of resources on which they have at least View role access rights.

POST
: This method allows users to upload new resources.

DELETE
: This method allows users to delete resources or folders, that is WebDAV collections.

PUT This method allows users to update resources, such as documents or images in a folder.

COPY
: This method allows users to copy resources or folders, that is WebDAV collections.

MOVE
: This method is used to move or rename resources or folders, that is WebDAV collections.

Related tasks:

"Enabling HTTP Basic Authentication for simple clients" on page 1610
IBM WebSphere Portal Express provides an HTTP Basic Authentication Trust Association Interceptor that can be enabled to allow specific clients to log into the portal by using HTTP Basic Authentication instead of HTTP Form Based Authentication.

Serving HTTP OPTIONS requests to the server context root by WebDAV clients

Some WebDAV clients send an HTTP OPTIONS request to the server context root (/) to check whether the server supports WebDAV. To support these clients, the portal provides a web application called wp.webdav.options.war that you can enable. This application responds to such requests with a confirmation that the portal supports WebDAV.

About this task

To enable this WAR file, adapt the application.xml files of the deployed Enterprise Application (EAR), wps.ear, so that wp.webdav.options.war is mapped to the context root (/). Proceed as follows:

Procedure

1. Export the EAR, wps.ear, through the WebSphere Integrated Solutions Console.
2. Open a command prompt.
3. Run the following command to expand the EAR:

   ```
   ./EARExpander.sh|bat -ear directory/WebDAV_for_WebSphere_Portal.ear
   -operationDir directory webdav_expanded -operation expand
   ```

4. Locate the file application.xml in the expanded EAR file directory.
5. Edit the file application.xml of the exported EAR, and uncomment or add the following section:

   ```
   <module>
        <web>
             <web-uri>wp.webdav.options.war</web-uri>
             <context-root>/</context-root>
        </web>
   </module>
   ```

6. Run the following command to collapse the EAR:

   ```
   EARExpander.sh|bat -ear directory/wps.ear
                      -operationDir directory/wps_expanded
                      -operation collapse
   ```

7. Update the enterprise application with these changes by using the WebSphere Integrated Solutions Console.
8. Save your changes.
9. Restart the portal for your changes to take effect.

What to do next

Note: Depending on how your web server is set up, this change might cause all requests to be routed to the web server plug-in. In this case refer to your web server documentation for information about how to route only OPTIONS requests to the plug-in.

Working with WebDAV clients

WebDAV is an HTTP extension framework with a plug point for the access and management of hierarchical data, for example, in content management systems. WebDAV stores the data in collections and allows you to work with the data in a user interface view that is similar to that of a file system. Various tools are available for integrating WebDAV resources into the client file system, known as WebDAV clients. To use WebDAV for WebSphere Portal Express, you must first set up your WebDAV clients.

About this task

For specific WebDAV client version compatability, refer to the detailed systems requirements documentation and the Tech Note about *WebDAV clients for accessing IBM WebSphere Portal*.

Notes:

- Numerous other WebDAV clients are available that you can use for WebDAV access. IBM supports the use of these WebDAV clients; however, IBM does not provide fixes or give support for issues found to be specific to a particular WebDAV client.
- Some WebDAV clients have specific restrictions, for example a limit to the size or number of files that you can handle when using WebDAV, or a read only restriction. These restrictions usually have security reasons. If you encounter issues when working with WebDAV, consult the documentation and forums for your WebDAV client.

When you use a Web server to work with WebDAV, complete the following steps:

Procedure

1. Access the WebSphere Integrated Solutions Console.
2. Select **Web servers** > **webserver name** > **Plug-in properties** > **Request and response**.
3. Set **Accept content for all requests** to `true` for the Web server plug-in.
4. Regenerate the web server plug-in.
5. Copy the file `plugin-cfg.xml` to the `Plugin` directory.
6. Open your `plugin-cfg.xml` file and set **`AcceptAllContent`** to `true`.
7. Restart the web server.

Related tasks:

"WebDAV" on page 1986
With WebDAV for IBM WebSphere Portal Express, you can use standard operating system tools to create, modify, and delete web content rather than the standard authoring portlet.

Related information:

Detailed system requirements

WebDAV clients for accessing IBM WebSphere Portal

Task `webdav-deploy-zip-file`

Use this configuration task to manage theme artifacts and to deploy iWidgets. This task uploads archive or compressed files to portal WebDAV folders.

Address the target folder by using a corresponding DAV URI, for example `dav:fs-type1/iwidgets/myWidget/`.

Note:

- By default, this task replaces the referenced target folder by the extracted contents of the referenced archive or compressed file. As a result, files or folders that are contained in the referenced WebDAV folder are deleted before the new content is added. If you want to avoid this behavior and merge the contents, set

the optional `UpdateMode` parameter to the value `merge`. If you add this setting, the task merges the contents of the archive or compressed file into the content that exists at the target URI.

- If you have a portal cluster installation, start the task on the primary node only. Starting the task on a secondary node has no effect.

If the WebDAV folder referenced by the target URI does not exist yet, the task creates it. In this case, make sure to have a trailing slash (**/**) at the end of the target URI.

Description:
This task uploads archive or compressed files to portal WebDAV folders.

Mandatory parameters to be specified on the command line or in the file `wkplc.properties`:

`WasUserid`
The WebSphere Application Server user ID.

`WasPassword`
The WebSphere Application Server password.

`PortalAdminId`
The WebSphere Portal Express administrator user ID.

`PortalAdminPwd`
The WebSphere Portal Express administrator password.

Mandatory parameters to be specified on the command line only:

`TargetURI`
The URI of the WebDAV folder where you want the archive or compressed file to be extracted.

Path parameter:
use only one of the following two path parameters. They are mutually exclusive:

`ZipFilePath`
The file system path to the archive or compressed file. Do not use this parameter in combination with the parameter **`ZipFileClassPath`**.

`ZipFileClassPath`
The Java class path to the archive or compressed file. Do not use this parameter in combination with the parameter **`ZipFilePath`**.

Optional parameters to be specified on the command line only:

`UpdateMode (=replace)`
The default value for this parameter is **`replace`**. If you want to merge the content of the archive or compressed file with the content that exists at the target URI, set this parameter to the value **`merge`**. In this case files that do not yet exist are created, existing files are updated, and no files are deleted.

`VirtualPortalContext`
`VirtualPortalHost`
Use one of these two parameters to identify the virtual portal. Only pages that are contained in the specified virtual portal are refreshed. If you omit this parameter, by default no virtual portal page layout is refreshed.

Example:

You can upload the file `foo.zip` to the public folder of the file store by using either one of the following options:

- Copy the file `foo.zip` to the portal class path, for example `AppServer/lib/ext`.
- Copy the file `myWidget.zip` to the temporary directory `/tmp` on the portal server node.
- Run the configuration task `webdav-deploy-zip-file` as follows:

```
./ConfigEngine.sh webdav-deploy-zip-file
     -DTargetURI=dav:fs-type1/iwidgets/myWidget/
     -DZipFilePath=/tmp/myWidget.zip
```

Virtual portals

View information on how you can scope your WebSphere Portal Express to have multiple virtual portals.

Virtual portals can be of benefit if you want to serve multiple user groups by separate virtual portals for different purposes, but still want to keep your environment simple and limited to a single WebSphere Portal Express installation. Virtual portals allow a large extent of scoping and separating portal resources, user groups, and administration.

"Deciding about virtual portals" on page 1362
Get an overview of possible business and usage scenarios for virtual portals. Decide whether your business scenario is suitable for virtual portals. Get some ideas and hints about what you need to consider when you plan for your virtual portals. This can help you determine whether your business can work with virtual portals, how many virtual portals it requires, and how, and for which purposes you will use them. Based on your decision, you can then plan how you implement and configure your virtual portals.

"Planning for virtual portals" on page 1366
Before you create your virtual portals, review this information for planning purposes. Determine how many virtual portals your business requires, and how and for which purposes you will use them. Based on your decision, plan how you implement and configure your virtual portals.

"Virtual portals and managed pages" on page 1386
Virtual portals are created and managed through the portal administration interface with the Virtual Portal Manager portlet. When you create a virtual portal, a workspace is created that contains a new Portal Site web content library. Any managed pages that are created in the virtual portal are stored in the Portal Site library.

"Administering virtual portals" on page 1386
View information to help you scope your WebSphere Portal Express to have multiple virtual portals.

"Working with the Virtual Portal Manager portlet" on page 1399
For improved manageability of virtual portals, WebSphere Portal Express provides an administration portlet. It is named **Virtual Portal Manager**. It allows you to create virtual portals on demand. You can also use it to pre-configure and administer virtual portals.

"Virtual portals reference" on page 1405
The virtual portals reference provides information about using commands for configuring virtual portals, usage hints and tips, and known limitations.

Related information:

Technotes for virtual portals

Deciding about virtual portals

Get an overview of possible business and usage scenarios for virtual portals. Decide whether your business scenario is suitable for virtual portals. Get some ideas and hints about what you need to consider when you plan for your virtual portals. This can help you determine whether your business can work with virtual portals, how many virtual portals it requires, and how, and for which purposes you will use them. Based on your decision, you can then plan how you implement and configure your virtual portals.

"Scenarios with multiple portals for your business requirements"
Before you decide on a portal installation with multiple virtual portals, you need to determine your specific business requirements and the purpose of your portal. This can help you decide whether virtual portals are a valid solution for your requirements, or whether it is better for you to use multiple real portals. Consider and answer the questions in the following sections.

"Alternative concepts for virtual portals on WebSphere Portal Express" on page 1363
Besides virtual portals, another possible configuration may be an alternative for you, depending on your business needs. This setup is referred to as true portals.

"Usage scenarios for virtual portals" on page 1364
Learn about three typical usage scenarios for virtual portals.

Related concepts:

"Separating and sharing resources between virtual portals" on page 1367
Separation between virtual portals is achieved by **scoping** the portal resources of the virtual portals. Scoping means making portal resources available uniquely and separately to individual virtual portals and their users.

Related information:

Technotes for virtual portals

Scenarios with multiple portals for your business requirements

Before you decide on a portal installation with multiple virtual portals, you need to determine your specific business requirements and the purpose of your portal. This can help you decide whether virtual portals are a valid solution for your requirements, or whether it is better for you to use multiple real portals. Consider and answer the questions in the following sections.

Using virtual portals versus multiple real portals

The benefit of virtual portals consists of sharing several resources between the virtual portals. Rather than having one of these resources for each portal, all virtual portals use the same single instance of such a resource. This reduces the administrative cost and optimizes the resource usage. Typically, virtual portals share the following resources:

- The JVM
- Portlets and other code fragments
- The database

Additional to the benefit of sharing these resources, the virtual portals can be scaled to a large extent, and you can host many virtual portals on a single portal installation. For a complete list of the resources that virtual portals share, see *Separating and sharing resources between virtual portals*. Sharing resources, however, can create dependencies between virtual

portals. For example, if one of the virtual portals requires maintenance, all virtual portals are affected by the outage and undergo the same maintenance updates. If you can accept such dependencies in your business environment, virtual portals are a simple and cheap solution for you. Otherwise, you have the alternative of using multiple real portal installations. For more information, see *Alternative concepts for virtual portals on WebSphere Portal Express*.

Sharing or separating virtual portal administration

Do you plan to have each virtual portal administered by its own group of administrators, or will you have a central administration group for the entire portal installation and all virtual portals?

You can select a specific group of subadministrators who can manage the resources and users of a particular virtual portal. The master administrator of the portal installation can set up the privileges of the individual subadministrators for each virtual portal.
If you do not require a specific subadministrator group for each virtual portal, the portal administrators can share the administrative work for all virtual portals.

Sharing or separating user populations

Does each virtual portal need its own separate user population, or can all virtual portals share the single user population?

To ensure that only members of a dedicated user population can access a virtual portal, use the realm concept that is provided by the Virtual Member Manager (VMM). VMM is available as a built-in user registry in WebSphere Application Server. This security concept is known as federated security.
If all your virtual portals can use the same user population, you can configure federated security with a single realm. This realm can contain users and groups from one or more repositories.

Related concepts:

"Separating and sharing resources between virtual portals" on page 1367
Separation between virtual portals is achieved by **scoping** the portal resources of the virtual portals. Scoping means making portal resources available uniquely and separately to individual virtual portals and their users.

Related information:

Technotes for virtual portals

Alternative concepts for virtual portals on WebSphere Portal Express

Besides virtual portals, another possible configuration may be an alternative for you, depending on your business needs. This setup is referred to as true portals.

This setup allows the re-use of a single hardware, with multiple complete portal installations, that is, one dedicated software profile for each portal. Each portal installation requires its own complete WebSphere Application Server installation. These are the main advantages of true portals:

- The strong isolation of the configuration data due to separate configuration databases
- The full isolation of applications, due to a separate JVM for each true portal. This allows better quality of service.

If you want to implement this solution, be aware of the following limitations:

- You can run only a limited number of true portals on a single hardware machine. This is due to the memory volume required by the JVM.
- You cannot share applications or data between true portals.

Related information:

Technotes for virtual portals

Usage scenarios for virtual portals

Learn about three typical usage scenarios for virtual portals.

Scenario 1: Multi-Portal Enterprise

In this scenario a single enterprise operates multiple different virtual portals on a single portal installation. The virtual portals are used for different parts of the organization, such as the following:

- Development, production, and marketing
- Organizations or branches in different locations and foreign countries
- Affiliate or franchise business models
- Different branding.

These are some of the typical business requirements:

- The portal installation as a whole is operated by the company, as the different parts of the organization are too small to have their own IT staff.
- A common group of administrators is responsible for the administration of all the virtual portals in the installation.
- Each part of the organization wants their own individually customized virtual portal.
- Many applications are commonly used by the different parts of the company, and they are shared between the virtual portals.
- All portal users are contained in the company user directory.
- The portal installation might typically have between ten and thirty virtual portals.

As many resources are commonly used by all sections of the company, sharing these resources has large benefits, and the resulting dependencies are acceptable for the enterprise. In this case, using virtual portals is the appropriate option, as it reduces the amount of resources required for the administration of the portal. For the requirements of this scenario, you can select one or both of the following options:

- As the sections of the company are too small to each have their own administrative staff, you can use a shared group of administrative users.
- To allow all members of the company to access all virtual portals, a shared user population is a suitable approach. Nevertheless, you can reduce the availability of specific resources to specific user groups by assigning access rights accordingly.

Scenario 2: Workgroup Service Provider

In this scenario one central organization provides virtual portals for a large number of small, decentralized, and independent teams. For example, this can be teamrooms for project management in small work units. This scenario supports virtual portals for different parts of the organization as follows:

- It supports a large number of individual virtual portals on a single portal installation. This can be more than a hundred virtual portals.
- The individual logical Portals are intended for small user groups, projects or departments.
- The owning enterprise operates this installation like an IT service provider.
- It is important that virtual portal administrators can create additional virtual portals with predefined default content fast, easily, and on demand. These can be based on a customized virtual portal template.
- Sharing content and applications is a very important aspect in this scenario.
- Administration of each virtual portal and its users and resources is independent and self-contained.

Scenario 3: Hosted Enterprises

In this scenario a service provider hosts and operates independent enterprises on the same portal installation. For example, this scenario can support virtual portals for different tenants or service customers, such as:

- A service provider who supports services for small businesses of the same type.
- A provider who offers services for medical doctor practices.
- A central banking service provider who offers services to different branches of banks.

The business requirements for this scenario include the following:

- Most applications are shared between the tenants.
- The tenants need to be able to administer their virtual portals themselves. If the critical business data of the tenant is stored in the back end of the tenant system, and not in the shared database of the virtual portals, sharing of other portal resources, such as JVM and database is acceptable for the tenants.
- The portal installation might typically have between ten and thirty virtual portals.
- Each tenant portal has its own user directory.
- The content shown to users in the portlets are maintained in the back ends of the tenant companies themselves. For example, this is the case in a service portal with separate virtual portals for individual banks.

Based on these business requirements, you can select one or more of the following options:

- Each virtual portal has its own groups of administrators. The administrators of one virtual portal manage the rights of the users in that virtual portal.
- As the tenants each have their own user repository, they can set up their virtual portal with their own realm. This provides strong separation between the users in the different virtual portals. If individual users need to access more than one of these virtual portals, they need a user ID for each virtual portal that they access.
- You can circumvent the dependency on a common maintenance window by setting up two sets of virtual portals: while one is used for production and carries the user load, the other one is free for maintenance. This also gives the required high availability.

You need to set up the configuration required for such a scenario carefully. As an alternative for this scenario, you can consider using separate portal installations rather than virtual portals. This can be the easier solution especially if you provide

service for a small number of large tenants.

Related concepts:

"Separating and sharing resources between virtual portals" on page 1367
Separation between virtual portals is achieved by **scoping** the portal resources of the virtual portals. Scoping means making portal resources available uniquely and separately to individual virtual portals and their users.

"Managing the user population for virtual portals" on page 1370
You have two basic options for the management of user populations for your virtual portals: Virtual Member Manager (VMM) or Lightweight Directory Access Protocol (LDAP).

Related information:

 Technotes for virtual portals

Planning for virtual portals

Before you create your virtual portals, review this information for planning purposes. Determine how many virtual portals your business requires, and how and for which purposes you will use them. Based on your decision, plan how you implement and configure your virtual portals.

IBM has tested an installation with 300 virtual portals successfully. The limiting factor is not the absolute number of virtual portals, but the overall number of pages and URL mappings. For more details see the topic about *Known limitations for virtual portals*.

IBM WebSphere Portal Express provides two pre-configured virtual portals, one each for use as internet and intranet portals. For more information see the topic about *Scope of virtual portals*.

"Separating and sharing resources between virtual portals" on page 1367
Separation between virtual portals is achieved by **scoping** the portal resources of the virtual portals. Scoping means making portal resources available uniquely and separately to individual virtual portals and their users.

"Managing the user population for virtual portals" on page 1370
You have two basic options for the management of user populations for your virtual portals: Virtual Member Manager (VMM) or Lightweight Directory Access Protocol (LDAP).

"Virtual portal roles and their capabilities" on page 1375
A typical virtual portal scenario works with a master administrator and sub-administrators. Assigning access permissions to the users of virtual portals also requires special considerations.

"Content of a virtual portal" on page 1382
The content of a newly created virtual portal can vary, depending on the method by which you create the virtual portal. You can change the pre-configured content for virtual portals.

"Shaping the user experience" on page 1384
You can improve the user experience that users have with your virtual portals by using human readable URLs, or by using custom themes and skins for your virtual portals.

Related concepts:

"WSRP services" on page 1433
IBM WebSphere Portal Express supports the Web Services for Remote Portlets (WSRP) standard. By using this standard, portals can provide portlets, applications, and content as WSRP services. Other portals can then integrate the WSRP services

as remote portlets for their users.

"Controlling access" on page 1524
After creating users and groups, you can assign them different levels of access to specific resources, roles, and policies. This access controls what actions they can perform on various pages, portlets, and applications.

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related reference:

"Scope of virtual portals" on page 1418
IBM has tested an installation with 300 virtual portals and a total of more than 200,000 pages successfully. More detailed results of these tests are listed here.

Related information:

 Technotes for virtual portals

"Tasks for administering virtual portals" on page 1389
Administering virtual portals and their content comprises the tasks described in the following topics.

Separating and sharing resources between virtual portals

Separation between virtual portals is achieved by **scoping** the portal resources of the virtual portals. Scoping means making portal resources available uniquely and separately to individual virtual portals and their users.

Scoping of resources works as follows:

- A portal resource that is scoped for virtual portals exists individually for each virtual portal. It has an identification that is unique within the entire portal installation. The resource is available only in one particular virtual portal. Consequently, you can customize such resources for each virtual portal independently. Example: The resource `resource_A` is scoped for the virtual portals `VP_1`, `VP_2`, and `VP_3` as `resource_A_VP_1`, `resource_A_VP_2`, and `resource_A_VP_3`. Customizing `resource_A_VP_1` does not affect `resource_A_VP_2` or `resource_A_VP_3`.
- A portal resource that is not scoped for virtual portals is shared between all virtual portals. Consequently, if you customize this resource, this will affect that resource in all virtual portals equally.

Scoping works for some portal resources, but not for others:

- IBM WebSphere Portal Express scopes some portal resources for virtual portals. This means that these resources exist separately for each virtual portal.
- Other resources are common for all virtual portals in a portal installation. However, you can scope some of these resources:
  - You can scope some resources by using portal administration and Portal Access Control.
  - There are some portal resources that cannot be scoped at all.
- The user population can be scoped to one or more specific virtual portals.

The differences in scoping portal resources are described in the following sections.

Portal resources that are scoped for virtual portals

WebSphere Portal Express has the following portal resources scoped internally for virtual portals:

- Portal pages

- Portlet instances
- Portal Search Engine search services and search collections. This includes the search content sources.
- IBM Web Content Manager web content libraries. See the note later in this section.

Scoping of these resources is managed by internal portal mechanisms. Scoped resources are only available for the virtual portal for which they are defined. They are well isolated from other virtual portals. Scoped resources cannot be shared with other virtual portals. They are not visible or accessible outside of the virtual portal for which they have been created. This behavior cannot be changed by any portal access control settings.

The following rules apply:

- Within each virtual portal you or a sub-administrator can use Portal Access Control to grant individual users of that virtual portal specific access permissions to the scoped portal resources. This works just like under a single portal installation.
- An administrator can give access permissions to users who are members of the user population of a virtual portal only on the scoped resources of that same virtual portal. This implies that, vice versa, you can give access permissions on the resources of a virtual portal only to those users who are members of the user population of that virtual portal.
- Users can only use these access permissions when they access the specific virtual portal under which they have the access permissions on the scoped resources. The same users cannot access the resources when logging in to a different virtual portal.

Note for IBM Web Content Manager web content libraries: IBM Web Content Manager web content libraries are scoped to virtual portals if Managed Pages are enabled as by the default WebSphere Portal Express installation. If you want to make IBM Web Content Manager web content libraries available between your virtual portals, you can do so by disabling Managed Pages and restarting your portal. IBM Web Content Manager web content libraries of the base portal are then also available to the virtual portals.

Portal resources that you can separate for virtual portals by using Portal Access Control

There are some portal resources that are not scoped internally for a particular virtual portal. These resources are shared among all virtual portals of the entire installation. However, as a master administrator you can yourself separate such portal resources for the virtual portals. To do this, use Portal Access Control and the access permissions portlets to set up the appropriate access permissions for users on the resources of each virtual portal as required.

You can separate the following portal resources by using Portal Access Control to give users of an individual virtual portal access permission to the resources:

- Portlets
- Portlet applications
- Web modules
- URL mapping contexts
- Users and groups.

You can separate these resources for individual virtual portals by using Portal Access Control. When you do this, apply special care. It can be of benefit to document the relationships between the users and the virtual portals.

Portal resources that cannot be separated for virtual portals

There are some types of portal resources that are not scoped to a particular virtual portal, and you cannot separate them yourself by using Portal Access Control. The following list shows portal resources that you cannot separate for virtual portals:

Themes and skins
If you do not want sub-administrators to be able to manage themes and skins, restrict their access permissions on them.

Vault segments and vault slots
To avoid security problems, use private credentials only. They can be used by only one specific user.

Supported clients and markups
The settings for these are configured in the corresponding portlets; therefore they apply to the entire portal installation.

Policies
Policy resources are not scoped to virtual portals. Users see the policy resources to which they have access, regardless of the virtual portal assignments.

Personalization
Personalization is not aware of virtual portals. A document library that is available in the initial portal installation is also available in each virtual portal, if Personalization is available in that virtual portal and is configured to use that document library. Searching for a document in a document library will produce a document reference (URL) that is different in each virtual portal, but points to the same document in the document library. To provide separation of content within virtual portals, use separate document libraries for each virtual portal. To provide content collaboration between virtual portals, use the same document libraries between virtual portals.

Example: Themes and skins can be accessed by all sub-administrators who have the access permission to apply themes and skins to the pages that they can administer, regardless of which virtual portal the sub-administrators are responsible for.

Separating portlets, portlet applications, and portlet instances

Portlet applications are not scoped for virtual portals. Therefore, the configuration settings that you set for a portlet application by using the Manage Applications portlet apply to that portlet application in all virtual portals. If you need different configurations for a portlet application between virtual portals, create a copy of the portlet application, and configure the copied portlet application as required.

Portlets are separate portal resources, but they are not scoped for each separate virtual portal. However, each portlet in a virtual portal shares its portlet application on the initial portal installation with its siblings on the other virtual portals. Therefore the following configuration settings set for a portlet apply to that portlet in all virtual portals:

- The configuration settings that you set for a portlet by using the Manage Portlets portlet

- The configuration settings that you set for a portlet by using the **Configure** mode of the portlet.

Portlet instances are scoped to the virtual portals. If you need different configurations for a portlet between virtual portals, create a copy of the portlet, and configure the copied portlet as required. The configuration settings that you set for a portlet by using the **Personalize** or **Edit shared settings** mode of the portlet apply only to that individual portlet instance on that individual page.

Special case: Scoping unique names

Unique names that you apply to portal resources represent a special case with regards to scoping. Unique names are attributes to portal resources. Therefore, whether a unique name is scoped to a virtual portal or not is determined by whether the portal resource to which the unique name applies is scoped or not:

- Unique names for scoped portal resources are themselves also scoped.
- Unique names for resources that are not scoped are themselves not scoped.

Example for a scoped unique name: Each virtual portal has its own separate login page. Therefore you can assign the same identical unique name to all login pages for all virtual portals. The unique name that you give to the login page of a specific virtual portal applies only within that portal. It cannot be administered in a different virtual portal that has the same unique name for its login page.

Example for a unique name that is not scoped: Portlet applications are not scoped but shared between all virtual portals. You can assign a unique name to the portlet application. You can reference that portlet application by that unique name throughout the portal installation with all virtual portals.

Related information:

Technotes for virtual portals

Managing the user population for virtual portals

You have two basic options for the management of user populations for your virtual portals: Virtual Member Manager (VMM) or Lightweight Directory Access Protocol (LDAP).

This depends on whether you want separate user populations for your virtual portals or a simple solution with one user population for all virtual portals:

- Using the **Virtual Member Manager** that is integrated with WebSphere Application Server; it is also known as the Federated Repository. You can use the Federated Repository to set up both types of configuration:
  - Configuring separate user populations for each of your virtual portals. This option offers a high flexibility for the user management of your virtual portals. With this configuration, you can define an individual user population for each virtual portal.
  - Using a common user population with the Virtual Member Manager for all your virtual portals. In this case, all users of that user population can access all virtual portals, unless their access permissions are explicitly restricted by portal access control settings. To achieve this restriction, you must define the access permissions manually by using the Portal Access Control portlets.

A user registry can be based on a Lightweight Directory Access Protocol (LDAP) or on a database. For more information about configuring your virtual portals with Virtual Member Manager and the different configuration options see the following sections.

- Using the **Lightweight Directory Access Protocol (LDAP)**. If a single common user repository is sufficient for all virtual portals within your installation, you can continue to use an LDAP in your virtual portal setup. This is the simpler one of the two options.

  With this configuration the entire portal installation and all virtual portals share a common user population, which is defined in a single user repository. In this case all users of that user population can access all virtual portals, unless their access permissions are explicitly restricted by portal access control settings. In order to achieve access restrictions for specific virtual portals, you can use the Portal Access Control portlets. You define user groups and assign to them the access permissions to the resources of each virtual portal.

For WebSphere Portal Express installations, the Federated Repository option offers you more flexibility for the user management of virtual portals. By using the Virtual Member Manager, you can limit the usage of a particular virtual portal to a specific user population. This is achieved by introducing the concept of **realms**.

The following sections give overview information of how to use the Virtual Member Manager and realms in the context of virtual portals. For a wider overview of portal security see the topics about *Securing* and *Configuring* the portal and about access permissions, users and groups. For more details about how to configure the Virtual Member Manager and realms see the topics about adding realm support for your environment.

A virtual portal can only be accessed by members of its associated user population. By using Portal Access Control that you can assign and restrict access permissions within the user population of a virtual portal to the resources of that virtual portal. However, Portal Access Control cannot overwrite the predefined assignment of a particular user population to their virtual portal. You cannot use Portal Access Control to assign access permissions that cross the separation between virtual portals. For example, you cannot use the Portal Access Control of a virtual portal VP_A to give a user User_A_1 of that portal access to resources of another virtual portal VP_B. The following conditions apply:

- A realm contains the entire user population of one virtual portal.
- Each virtual portal can have its own realm of users that are associated. However, it is also possible that multiple virtual portals can share their user population by using the same realm in parallel.
- To be able to log in to a particular virtual portal, a user must be a member of the realm that is associated with that virtual portal.

For more details about preparing the Virtual Member Manager and realms for your virtual portals, read the next section.

Preparing the user populations for your virtual portals

If you plan to use realms for your virtual portals, you need to configure Virtual Member Manager and the realms before creating your virtual portals. Each realm must specify the repository nodes (base entries) that belong to the user population represented by this realm.

In addition to the realms that you create to define the user populations of the individual virtual portals, you must create a super realm. This super realm spans all other realms and contains all the users of those other realms; it is also known as the default realm.

By default WebSphere Portal Express is configured with Federated Repositories as User Registry provider. By default only the super realm, or default realm, is configured. After you have configured your portal instance against your user backend repositories, you can use tasks that are provided by the portal to configure the realms that the Virtual Member Manager provides. For the task that describes how to add a realm and modify the base entries or nodes inside that realm, read the topics about adding realm support for your portal environment.

Using a non-default realm: If you assign a non-default realm to the default virtual portal, ensure that all administrative accounts are available within the non-default realm. If you have Web Content Manager, do not use a non-default realm, as Web Content Manager is not scoped to virtual portals.

The following sections give an overview of example configurations of the Virtual Member Manager for virtual portals. For more information about configuring realms for your virtual portals, read *Virtual Member Manager integration*.

Configuring a common user population for all virtual portals

In a simple setup, you can use the Virtual Member Manager together with a common user repository. This user repository is represented by a single realm, and used by all virtual portals. In this case, all virtual portals use a common realm and a common user repository. This configuration provides no separation between the users of the different virtual portals.

WebSphere Portal Express still supports the WebSphere Application Server Lightweight Directory Access Protocol (LDAP) custom user registry that previous versions of WebSphere Portal Express used. You can configure it as alternative. Again, this configuration uses a common user repository for all virtual portals without separation between the users of the different virtual portals.

Configuring separate user populations for the individual virtual portals

If you want to have the users of your virtual portals that are separated, you must apply the more advanced setup by using Federated Repositories. Then, configure separate realms for your virtual portals. When users access a virtual portal, the portal installation selects the appropriate realm that is based on the current virtual portal context. Within a virtual portal, only users of that corresponding realm are "visible". The administrator of a particular virtual portal can give access permissions only to users and groups in the population of that virtual portal. Therefore, when you create a virtual portal, the realm that represents the population of the new virtual portal must be a subset of the realm that is used by your portal installation.

Note: This separation of user populations between virtual portals works only with Federated Repositories. The portal supports separate realms and user repositories for virtual portals only when you use the Federated Repositories.

When you use the Federated Repositories, you can separate user groups and administrative users by configuring your virtual portals according to your business requirements. You do this based on the following relationships between user repositories, realms, and virtual portals:

- You can aggregate users and groups from one user registry in one realm, and expose them as one coherent user population to the portal installation. You can separate the user population of each virtual portal by assigning different LDAP suffixes to the different realms. The LDAP suffixes are called base entries. This way the concept of realms allows you various flexible configuration options.
- A realm can aggregate one or more base entries in a user registry.
- A realm can combine multiple base entries of one user repository. A suffix of a user repository can belong to one or more realms. The LDAP suffixes of the individual users must match the suffixes of the groups to which they belong.
- A virtual portal is associated with one realm. Each virtual portal uses exactly one realm, but a realm can be used by multiple virtual portals.
- A virtual portal can also be associated with no realm. If no realm is assigned for a virtual portal, the user population that was defined for the super realm can log on to the virtual portal.
- When you use Federated Repositories, the initial portal installation has no realm that is associated by default. The user population of the initial portal installation spans the entire user registry that you configured in the Virtual Member Manager.
- The individual user IDs must be unique across all realms.
- To log in to a virtual portal, the virtual portal administrator and all users must be a member of the realm for that virtual portal. To allow a user access to more than one virtual portal, that user (and the Virtual Member Manager node to which the user belongs in the hierarchy of the user directory) must be a member of all the realms that are associated with these virtual portals. For example, this information applies to a super administrator who is responsible for all virtual portals within an entire Portal installation.
- To administer a virtual portal, the base portal administrator must be a member of the realm that is associated with the virtual portal. The base portal administrator is required to be part of all realms.

  If the administrator is not a part of the virtual portal realm, then the administrator does not have access to that virtual portal. This also applies to the base portal administrator group. If the portal administrator is not part of all realms, then you might encounter issues or the inability to complete certain tasks. For example, a portal installation with virtual portals cannot be migrated, if the portal administrator cannot access the virtual portals for the migration.
- User populations of realms can overlap. In other words, users can be members of multiple realms. If realms overlap, then these users can work in all the virtual portals that are associated with these realms.

Important: The administrator unique ID for the Java Content Repository (JCR) must be a distinguished name (DN) for a super administrator. You specify the administrator unique ID as the value defined in the **jcr.admin.uniqueName** property. To view this property, log in to WebSphere Integrated Solutions Console. Go to **Resources** > **Resource Environment** > **Resource Environment Providers** > **JCR ConfigService PortalContent** > **Custom properties** > **jcr.admin.uniqueName**. For example, you can set up the following configurations:

You can configure one LDAP suffix with all administrative users, for example `dc=administrators,dc=ibm,dc=com` and a separate LDAP suffix with the users, for example `dc=users,dc=ibm,dc=com`.

You can configure separate LDAP suffixes that contain different user populations, for example `dc=bank1,dc=com` for Bank_1 and `dc=bank2,dc=com` for Bank_2.

Notes:

Considerations for deleting resources in virtual portals:
The Portal Access Control administration in the Resource Permissions portlet shows users from different realms who have role mappings on shared resources by their object IDs. Therefore, apply special care and consideration when you delete such portal resources: Do not delete resources on which users from other realms have role mappings, if they are required in other virtual portals. This information applies to members of roles on portal resources that cannot be scoped and are therefore shared between the virtual portals. Role members who belong to the realm of your local virtual portal are displayed as usual, but role members who belong to different realms are displayed in a different manner:

- Role members for shared resources who belong to the realm of the virtual portal where you are currently working are listed by their actual names.
- Role members for shared resources that do not belong to the realm of the current portal are listed by their portal object IDs. For example, a role member from a different realm might be represented as `8_0_B`.

Find the list of role members. Click the **Administration menu** icon. Then, click **Access** > **Resource Permissions**. From the list of Resource Types, select a Resource Type by clicking it. On the Resource Permissions page, click the **Assign Access** icon. The members are listed in the **Roles** column.

How to grant virtual portal administrators access to web content libraries:
Virtual portal administrators do not automatically have access to work with web content libraries when you use the administration portlet. To enable a virtual portal administrator to work with web content libraries, you need to assign them access to either the JCR content root node or individual web content libraries:

- You can assign virtual portal administrators access to the JCR content root node with `Set access on root` in the Web Content Library view of the Administration portlet. For more information, go to the portlet online help.
  - Assign virtual portal administrators administrator access to the JCR content root node to enable them to create new libraries and view, edit, and delete all existing libraries.
  - Assign virtual portal administrators contributor access to the JCR content root node to enable them to create new libraries and view, edit, and delete libraries that they created.
- You can also assign virtual portal administrators access to libraries they did not create by editing the access settings of individual libraries.

Related concepts:

Chapter 12, “Securing,” on page 1517
Security tasks include setting up property extension databases and custom user repositories, configuring and activating SSL, and configuring authentication. In addition, tasks such as activating Federal Information Processing Standards (FIPS) and NIST SP800-131a security modules and configuring external security managers such as Security Access Manager might be required to secure your portal environment.

"Realm support" on page 134
A realm is a collection of users or groups from one or more branches of your repository tree. Those branches can be part of a single repository, for example an LDAP user registry, or it can be a combination of multiple user registries. A realm is then mapped to a Virtual Portal to allow the realm's user population to log in to the Virtual Portal. This functionality allows you to define areas within WebSphere Portal Express that only a limited set of users can access.

"Controlling access" on page 1524
After creating users and groups, you can assign them different levels of access to specific resources, roles, and policies. This access controls what actions they can perform on various pages, portlets, and applications.

"Virtual Member Manager integration" on page 132
IBM WebSphere Application Server includes the Virtual Member Manager (VMM), which IBM WebSphere Portal Express uses to access user and group information. VMM provides an interface that enables communication between WebSphere Portal Express and any repository, whether federated repositories or your own custom user registry.

Related tasks:

Chapter 5, "Configuring," on page 231
Run the following tasks after you install and deploy IBM WebSphere Portal Express. They address tasks that are typically run one time and have a global effect. Some configuration changes are made more frequently or do not have a global effect. These tasks are addressed in the Administering section.

"Adding realm support" on page 582
A realm is a group of users from one or more user registries that form a coherent group within IBM WebSphere Portal Express. Realms allow flexible user management with various configuration options. A realm must be mapped to a Virtual Portal to allow the defined users to log in to the Virtual Portal. When you configure realm support, complete these steps for each base entry that exists in your LDAP and database user registry to create multiple realm support.

"Managing the users of virtual portals" on page 1389
You manage the users of virtual portals by adding and configuring the user repository and later administering the users for virtual portals.

Related information:

 Technotes for virtual portals

Virtual portal roles and their capabilities

A typical virtual portal scenario works with a master administrator and sub-administrators. Assigning access permissions to the users of virtual portals also requires special considerations.

"The master administrator" on page 1376
A key role for the administration of virtual portals is the master administrator. This user ID is created during the initial installation of WebSphere Portal Express with the role administrator on the portal (`admin@portal`). This administrator is also the master administrator of the initial portal installation and all virtual portals that are created. This master administrator is created with all necessary access permissions for administering tasks that are related to the initial portal and the virtual portals.

"Portal Access Control with virtual portals" on page 1378
You can scope some portal resources for your virtual portals by using portal administration and Portal Access Control. For example, you can scope portlet applications. These resources are available to all virtual portals. You can scope these resources to specific virtual portals by limiting their accessibility to the

user populations of the required virtual portals. To make this limit, you use Portal Access Control. Resources that you scoped this way for one virtual portal cannot be accessed from other virtual portals.

"Subadministrators of a virtual portal and their access roles and permissions" on page 1379
When you create a virtual portal by using the Virtual Portal Manager portlet, you select a user group of subadministrators who you want to be responsible for the administration of the new virtual portal.

"Users of a virtual portal and their access roles and permissions" on page 1382
When you create a virtual portal, you need to be aware of the implications listed here.

Related information:

Technotes for virtual portals

The master administrator:

A key role for the administration of virtual portals is the master administrator. This user ID is created during the initial installation of WebSphere Portal Express with the role administrator on the portal (`admin@portal`). This administrator is also the master administrator of the initial portal installation and all virtual portals that are created. This master administrator is created with all necessary access permissions for administering tasks that are related to the initial portal and the virtual portals.

The master administrator has the necessary privileges to run the tasks that are related to managing virtual portals. Use either the **Virtual Portal Manager** portlet or the provided configuration tasks to complete these tasks. For information about these tools, go to *Administering virtual portals*.

The Virtual Portal Manager portlet is installed as part of the initial portal installation. You can use this portlet to create, modify, and delete virtual portals.

The master administrator defines the user population of each virtual portal. To separate the user populations of the virtual portals, the master administrator can either use the **User and Group Permissions** portlet or they can define realms in the Virtual Member Manager configuration files.

Before you create a virtual portal, you define a group of subadministrators. When you create the virtual portal, a default set of roles and access permissions is assigned to this group. As the master administrator, you can change these default assignments and delegate administration of individual virtual portals to subadministrators. Use the **Resource Permissions** portlet that is part of the Portal Access Control.

When you create a virtual portal, it is filled with a default set of portal pages and resources. You can further enhance the content of a virtual portal by either of the following ways:

- By the master administrator of the portal installation. For example, use the XML configuration interface.
- By the subadministrators or other users of the virtual portal. Use the **Manage Pages** portlet.

For information about the content of a virtual portal, go to *Content of a virtual portal*.

Typically, only the master administrator should have the access permissions for the following tasks:

- Using the Virtual Portal Manager portlet
- Using the XML configuration interface to run tasks that are related to one of the virtual portals
- Installing portlets, themes, and skins.

Note: Do not grant the subadministrators of virtual portals the access permissions to run any installation-related tasks, such as installation of portlets or themes. All virtual portals share a common Java virtual machine (JVM). Therefore, it is important to restrict the administration privileges of the virtual portal subadministrators and prevent them from installing their own code artifacts, such as themes or portlets. Unstable or malicious code that is introduced on one virtual portal can destabilize the entire portal installation and all other virtual portals. A flexible way to introduce virtual portal-specific portlets without impacting any other virtual portal is to use web services for remote portlets (WSRP). By using WSRP, you can provide portlets on a remote server and then have the virtual portals consume those portlets so that users can access them remotely. For more information about using WSRP with your portal, go to *Using WSRP services*.

For more information about Portal Access Control, go to *Controlling access*. For more information about virtual portal security, go to *Portal Access Control with virtual portals*.

Related concepts:

"Content of a virtual portal" on page 1382
The content of a newly created virtual portal can vary, depending on the method by which you create the virtual portal. You can change the pre-configured content for virtual portals.

"Controlling access" on page 1524
After creating users and groups, you can assign them different levels of access to specific resources, roles, and policies. This access controls what actions they can perform on various pages, portlets, and applications.

"Setting resource permissions" on page 1568
Assign and control access for different types of resources.

"WSRP services" on page 1433
IBM WebSphere Portal Express supports the Web Services for Remote Portlets (WSRP) standard. By using this standard, portals can provide portlets, applications, and content as WSRP services. Other portals can then integrate the WSRP services as remote portlets for their users.

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related tasks:

"Setting user and group permissions" on page 1567
The User and Group Permissions portlet lets you view and modify the roles that users and groups have on resources.

Related information:

Technotes for virtual portals

"Administering virtual portals" on page 1386
View information to help you scope your WebSphere Portal Express to have multiple virtual portals.

Portal Access Control with virtual portals:

You can scope some portal resources for your virtual portals by using portal administration and Portal Access Control. For example, you can scope portlet applications. These resources are available to all virtual portals. You can scope these resources to specific virtual portals by limiting their accessibility to the user populations of the required virtual portals. To make this limit, you use Portal Access Control. Resources that you scoped this way for one virtual portal cannot be accessed from other virtual portals.

Portal Access Control provides a flexible concept to grant certain users or user groups access privileges to specific pages and other resources of a portal. A super administrator can delegate a subset of the administration privileges to other administrative users. You can use this flexibility to enable separation between different virtual portals in the following ways:

- Use the delegated administration model to set up individual partitions in your portal for the virtual portals.
- Define separate subadministrator users who administer the individual virtual portals and give each of the subadministrators the access permissions for their virtual portals.
- Define separate user populations who can access the individual virtual portals. For more detail about how this setting is supported see *Managing the user population for virtual portals*.

The inheritance concept of Portal Access Control allows this setup. The combination of access permissions that a subadministrator has on portal resources and on users and groups defines the scope of the virtual portal of that subadministrator:

- By inheritance, subadministrators of virtual portals implicitly have the administrative access permissions for all the child pages of their respective root content nodes, and of the content of their virtual portals. The subadministrator of a virtual portal cannot assign any access permission on resources that are scoped for other virtual portals.
- Depending on the access permissions to users and groups that the master administrator gives the subadministrators, they can grant access to users who belong to the user population of their virtual portals. The subadministrator of a virtual portal cannot assign any access permissions to users or groups of other virtual portals.

This way, each virtual portal represents a certain sub area of the main portal and can be managed individually.

Related concepts:

"Managing the user population for virtual portals" on page 1370
You have two basic options for the management of user populations for your virtual portals: Virtual Member Manager (VMM) or Lightweight Directory Access Protocol (LDAP).

Related information:

Technotes for virtual portals

Subadministrators of a virtual portal and their access roles and permissions:

When you create a virtual portal by using the Virtual Portal Manager portlet, you select a user group of subadministrators who you want to be responsible for the administration of the new virtual portal.

During the creation of the new virtual portal the Virtual Portal Manager portlet assigns the following default that is set of necessary access permissions on the virtual portal to the subadministrator group that you specified:

- Administrator role access permission on the root label of the virtual portal. You cannot change this access permission.
- Administrator role access permission on the virtual portal URL context. You cannot change this access permission.
- Editor role access permission on the administration portlets that are part of the virtual portal. You can change this access permission.

As the subadministrators have the Editor role access permissions on the administration portlets of their virtual portal, they can use these administration portlets to run administrative tasks on the virtual portal. For example, they can add portlets to pages. The default access permissions that are given to subadministrators of virtual portals are limited to managing the pages and documents in the virtual portal. Depending on your specific use case scenario, you might want to give the subadministrators extra permissions to manage more resources, or possibly users. Some of the permissions are not scoped to the virtual portal, but are global to the whole portal installation. This is particular critical for resources available in all virtual portals. For example, if you use a single realm for all virtual portals, the users are available in all virtual portals. A subadministrator who has the permissions to manage a user can manage that user in all virtual portals.

If you want to change the default access permissions for the subadministrators, use one of the following actions:

- If you want to change the default Editor access permission for the subadministrators on the administrative portlets or the list of portlets *globally* and *before* you create virtual portals, configure the Virtual Portal Manager portlet accordingly. For details about how to do this see *Pre-configuring the subadministrators for virtual portals*.
- If you want to assign extra access permissions to the subadministrators *specifically* and *after* creating a virtual portal, use the master administrator user ID of your initial portal installation and modify those access permissions for them manually in Portal Access Control. To do this, you can use the User and Group Permissions portlet, the Resource Permissions portlet, the XML configuration interface, or the Portal Scripting Interface. The consequences differ, depending on where you make the updates:
  - If you do this in the initial portal installation, you can change the access permissions for the subadministrators on the virtual portal as a whole.
  - If you do this in the virtual portal itself, you can change the access permissions for the subadministrators on the individual resources of the virtual portal.

The following list shows the tasks for which you can assign extra access permissions to subadministrators of virtual portals. It also specifies whether an access permission is scoped to the virtual portal or if it is global to the entire portal

installation, including all virtual portals. You can assign the permissions for these tasks to subadministrators only by using use the master administrator user ID of your initial portal installation.

Granting access permissions to users and groups of virtual portals
This task requires one of the following access permissions:

- Delegator on the group that defines the users of the virtual portal. This way is the preferred option, as the access permission is limited to the virtual portal.
- Delegator@Groups or Delegator@Users. Both of these access permissions apply globally to the entire portal installation, including all virtual portals.

Cloning portlet applications, for example, the web clipping portlet
This task requires Editor@Portlet Application. This access permission applies globally to the entire portal installation, including all virtual portals.

Access permissions for policies.
To manage policies, subadministrators need different access permissions, depending on the task that you want the subadministrative user to be able to complete. For example, to delete policies, a subadministrator needs Manager@Policy and User@Business Rules. This access permission is the highest permission. These access permissions apply globally to the entire portal installation, including all virtual portals.

Using the XML configuration interface
This task requires Security Administrator@Portal and Editor@XML access. These access permissions apply globally to the entire portal installation, including all virtual portals.

Managing portal search collections
This task requires Editor@Virtual Resource PSE_SOURCES. This access permission applies globally to the entire portal installation, including all virtual portals.

Managing URL mappings
This task requires Editor@parent context for parent mappings and Manager@context for URL mappings. These access permissions apply globally to the entire portal installation, including all virtual portals.

Managing tags and ratings
This task requires Manager@Tags and Manager@Ratings. These access permissions apply globally to the entire portal installation, including all virtual portals.

Managing personalization rules
This task requires the following access permissions:

- Privileged User on the following portlet applications:
  - Personalization Editors
  - Personalization Navigator
  - Personalization Picker
- Manager@the Personalization Rule

These access permissions apply globally to the entire portal installation, including all virtual portals.

Granting virtual portal administrators access to web content libraries
Virtual portal administrators do not automatically have access to work

with web content libraries when they are using the administration portlet. To enable a virtual portal administrator to work with web content libraries, you must assign them access to either the JCR content root node or individual web content libraries:

- You can assign virtual portal administrators access to the JCR content root node by using the `Set access on root` button in the Web Content Library view of the Administration portlet. For more information, see *Setting root access for all web content libraries* in the Portal Content help.
  - Assign virtual portal administrators administrator access to the JCR content root node to enable them to create new libraries and view, edit, and delete all existing libraries.
  - Assign virtual portal administrators contributor access to the JCR content root node to enable them to create new libraries and view, edit, and delete libraries that they created.
- You can also assign virtual portal administrators access to libraries they did not create by editing the access settings of individual libraries.

Templating sample content is provided by default with WebSphere Portal Express. This sample content is available from the Create Content tab of the site toolbar. If you want to use the sample content with a specific virtual portal, you must syndicate the following web content libraries to the virtual portal:

- `Template Page Content 3.0`
- `Web Content Templates 3.0`

If you fail to syndicate these libraries, the portal shows an error when you add the sample content to a page.

Related concepts:

"Controlling access" on page 1524
After creating users and groups, you can assign them different levels of access to specific resources, roles, and policies. This access controls what actions they can perform on various pages, portlets, and applications.

"Setting resource permissions" on page 1568
Assign and control access for different types of resources.

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related tasks:

"Pre-configuring the subadministrators for virtual portals" on page 1402
You can configure the roles and access rights that are assigned to subadministrators on portlets of a virtual portal *globally* and *before* you create a virtual portal. The following steps must be completed on your initial portal installation.

"Setting user and group permissions" on page 1567
The User and Group Permissions portlet lets you view and modify the roles that users and groups have on resources.

Related reference:

"Portal Scripting Interface" on page 1113
You can use the Portal Scripting Interface to configure your portal by running scripts from a command line.

Related information:


Technotes for virtual portals

Users of a virtual portal and their access roles and permissions:

When you create a virtual portal, you need to be aware of the implications listed here.

- The All Authenticated Users group is common across all virtual portals that share the same realm. When you create virtual portals, this group is given the Privileged User role on resources of all those virtual portals, independent of the role assignments that its users have on the initial portal installation. Restrict role assignments and thereby access permissions for the All Authenticated Users group, and assign access to user groups or users as required.

  Note that role assignments that you configured for users on the initial portal installation are not passed on to the virtual portal. For example, if you restricted access permissions to some virtual resources for users on the initial portal installation, these restrictions do not apply to the users in the context of the virtual portal.
- The All Authenticated Users group is valid over all virtual portals that share the same realm. This means that users who are in a realm that belongs to more than one virtual portal, these users have the assigned roles on all virtual portals to which they have access by membership to that realm.

If you want to change these default roles and the access permissions for the users, you can do this by one of the following ways:

- To configure the scope of access permissions for users *before* creating a virtual portal, configure your realms and user groups accordingly.
- To change the access permissions of users of a virtual portal *after* creating a virtual portal, use the Portal Access Control portlets in that virtual portal. You can have the sub-administrators of the virtual portal perform this task.

Related information:

Technotes for virtual portals

Content of a virtual portal

The content of a newly created virtual portal can vary, depending on the method by which you create the virtual portal. You can change the pre-configured content for virtual portals.

When you create the virtual portal by using the Virtual Portal Manager portlet, the portlet starts an XML configuration interface script that creates the initial content of the new virtual portal. The content of a virtual portal is similar to that of a full portal installation, but some administration portlets that manage global portal settings are not included in the default content of virtual portals. For example, the administration portlet Virtual Portal Manager is installed as part of the initial portal installation only. It is not part of the default content of virtual portals that you create. You can use it only in the initial portal installation.

When the content is created, the Virtual Portal Manager portlet grants the following set of default roles and access permissions to the subadministrators of the virtual portal:

- Administrator to the content root (Administrator@content root) of the virtual portal
- Editor to portlet instances (Editor@portlet entities) that are created for the new virtual portal.

You can modify the roles and access permissions for the subadministrators of a virtual portal manually according to your business needs:

- To change the roles and access permissions for subadministrators on the portlets *globally* and *before* you create a virtual portal, configure the Virtual Portal Manager portlet accordingly. For details about how to do this see *Pre-configuring the subadministrators for virtual portals*.
- To change the roles and access permissions *specifically* and *after* creating a virtual portal, use the Portal Access Control portlets. If you do this change in the initial portal installation, you can change the access permissions on the virtual portal as a whole. If you do this change in the virtual portal itself, you can change the access permissions on the individual resources of the virtual portal.

If you want to change the content of virtual portals, it can be done by one of the following ways:

- To change the content *globally* and *before* creating a virtual portal, advanced master administrators can reconfigure the XML script that specifies the initial content for virtual portals. For details about how to do this see *Pre-configuring the default content for virtual portals.*

  Note: When you modify or replace this XML script, plan ahead and apply special care. You can add or remove some content to enhance or reduce the functionality of a virtual portal to a certain extent. The following portal resources are mandatory content of a virtual portal and must be included in a customized XML initialization script for virtual portals:
  - Content Root (`wps.content.root`)
  - Login (`wps.Login`)
  - Administration (`ibm.portal.Administration`).

  Depending on the functionality that you want to make available, more content is required. For example, to allow templating. Include Application Root (`wps.application.root`) and Templates (`ibm.portal.Templates`).
- To change the content *specifically* and *after* creating a virtual portal, use either of the following portal tools:
  - Use the Manage Pages portlet of the virtual portal. The subadministrator of the virtual portal can do this change.
  - Use the XML configuration interface to import content into the virtual portal. This procedure can be done only from the initial portal installation.

If you use the configuration task `create-virtual-portal` to create a virtual portal, the new virtual portal that you create is empty. You must create the content for the virtual portal. For example, you can do this by using the XML configuration interface. For more information, see *The XML configuration interface*.

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related tasks:

"Pre-configuring the subadministrators for virtual portals" on page 1402
You can configure the roles and access rights that are assigned to subadministrators on portlets of a virtual portal *globally* and *before* you create a virtual portal. The following steps must be completed on your initial portal installation.

"Pre-configuring the default content for virtual portals" on page 1400
When you create the virtual portal by using the Virtual Portal Manager portlet, the virtual portal is pre-filled with default content. This default content is determined

by the default XML script file for initializing virtual portals.

Related information:

Technotes for virtual portals

Shaping the user experience

You can improve the user experience that users have with your virtual portals by using human readable URLs, or by using custom themes and skins for your virtual portals.

"Human readable URL mappings for virtual portals"
You can provide human readable URLs for your users to access their virtual portals. For example, you can give each virtual portal a human readable URL, such as `http://www.ibm.com:10039/wps/portal/tivoli.` You can pass the human readable URL of a virtual portal to its users. They can then use it to access their virtual portal.

"Individual themes and skins for each virtual portal" on page 1385
If you expose multiple virtual portals on a single portal installation, you can give each virtual portal its own look and feel for the user experience.

Related information:

Technotes for virtual portals

Human readable URL mappings for virtual portals:

You can provide human readable URLs for your users to access their virtual portals. For example, you can give each virtual portal a human readable URL, such as `http://www.ibm.com:10039/wps/portal/tivoli.` You can pass the human readable URL of a virtual portal to its users. They can then use it to access their virtual portal.

When you create a virtual portal, you specify the human readable URL as required by your business environment. The URL mapping that you specify is assigned to the virtual portal during its initialization. The URL mapping points to the content root of the virtual portal.

Internally, this URL mapping corresponds to a unique name `wps.vp.`*`internal_ID_of_the_virtual_portal`*. The portal installation uses this unique name to identify and access the virtual portal unambiguously. The XML configuration interface and the Portal Scripting Interface also use this URL mapping to identify the virtual portal.

You can also specify extra URL mappings for a virtual portal, both for the content root or for other content of the virtual portal, for example, a page in the navigation of the virtual portal.

All URL mappings use the same context root and servlet name in the URL. This setting applies to both the initial URL mapping of a virtual portal and any additional URL mappings that you might create for it.

For more information, see *URL mappings*.

Notes:

1. There is a 1:1 relation between a virtual portal and its initial URL Mapping. Each mapped URL points to the root content node of one virtual portal. You cannot use the same URL Mapping for two different virtual portals.

2. You must not delete or modify the initial URL Mapping for a virtual portal or modify its unique name. Deleting this URL Mapping or modifying its unique name makes the virtual portal unusable. This setting is independent of whether you use the administration portlets URL Mapping or Custom Unique Names or the XML configuration interface to change the URL Mapping.
3. If you use an external security manager, such as Tivoli Access Manager, you can restrict the usage of virtual portals by using the URL Mappings. To restrict, you base the URL filtering rules of a security proxy on the URL Mappings that you defined. Block all URLs by default and explicitly enable the defined URL Mappings only.
4. A URL mapping that is defined for a resource in a particular virtual portal must use the same URL context as the human readable URL context for that virtual portal itself. Example: In a virtual portal that uses the human readable URL mapping `wps/portal/vp_1`, all URL mappings for portal resources must start with `wps/portal/vp_1`, for example `wps/portal/vp_1/url_1` and `wps/portal/vp_1/url_2`. Within this virtual portal, a URL mapping such as `wps/portal/url_1` is not valid, as the portion `vp_1` of the URL Context is missing.
5. There are some strings that you cannot use as URL mappings for virtual portals, for example `vp`. These strings are reserved names and correspond with URL codec names. They are listed in the following:

   ```
   a0,    a0_1,   a0_2,   a0_3,  a1,    a2, a3,
   base64xml
   b0,    b0_1,   b0_2,   b0_3,  b1,    b2, b3,
   c0,    c0_1,   c0_2,   c1,    c2,    c3, c4,  c4_1, c4_2, c4_3, c5, c6, c7,
   cxml,  cxml_1, cxml_2, cxmld, kcxml,
   d0,    d1,     d2,     d3,    d4,    d5, dl2, dl3,  dl4,  delta
   kcxml, nm1,    nm2,    nm3,   nm4,   p0, pw,
   resource,      sel,    s0,    t0,    vp, wml,
   z0,    z0_1,   z0_2,   z0_3,  z1,    z2, z3
   ```

   For more information see the topic about *Restrictions to names for URL mapping contexts*.

Related tasks:

"URL mapping" on page 277
URL mappings were deprecated starting with WebSphere Portal Express Version 8.5. Instead, you can now use friendly URLs or Vanity URLs as an alternative to URL mapping.

Related information:

Technotes for virtual portals

Individual themes and skins for each virtual portal:

If you expose multiple virtual portals on a single portal installation, you can give each virtual portal its own look and feel for the user experience.

When you create virtual portals, the portal creates parallel root content nodes for each virtual portal. You can apply separate themes and skins for each content root and its child pages without impacting the representation of other content in the parallel trees for the other virtual portals. Each virtual portal will look like its own portal to its users. Users will not be aware that there are two or more different content nodes on the same physical portal installation.

You can apply the specific look and feel of each virtual portal to both the (unauthenticated) Welcome page and the authenticated pages of the virtual portal.

This means that each virtual portal can have its own look and feel even before the user logs in to the portal. Users can switch between the unauthenticated pages of different virtual portals by simply entering the different URL to get to the other portal. You can also provide specific login, and self-enrollment pages for each virtual portal. Once users log out, they are redirected to the specific unauthenticated page of the virtual portal that they had accessed.

Related information:

Technotes for virtual portals

Virtual portals and managed pages

Virtual portals are created and managed through the portal administration interface with the Virtual Portal Manager portlet. When you create a virtual portal, a workspace is created that contains a new Portal Site web content library. Any managed pages that are created in the virtual portal are stored in the Portal Site library.

Because web content libraries are not shared across virtual portals, these pages are visible only within the virtual portal.

Syndication of the Portal Site library for a virtual portal is the same as syndication of any other web content library.

Related information:

Technotes for virtual portals

Administering virtual portals

View information to help you scope your WebSphere Portal Express to have multiple virtual portals.

Note: Before you start creating or administering virtual portals, read the information in *Planning for virtual portals*.

Administering virtual portals and their content comprises the following tasks:

- Administering the portal content and resources for virtual portals
- Administering the users for virtual portals
- Administering content and search with virtual portals

You can use the following tools to administer your virtual portals:

- The Virtual Portal Manager administration portlet
- Command line tools as follows:
  - You can use portal configuration tasks for administering virtual portals.
  - You can use the XML configuration interface to work with virtual portals.

The following table shows how you can use these portal tools to administer virtual portals:

Table 175. Administrative tasks and the portlets and other tools for these tasks

Administrative task	Portlet for this task	Configuration task	XML configuration interface
Configuring the sub-administrators for a virtual portal	Access control portlets	---	X
Creating a virtual portal	Virtual Portal Manager	X	---
Filling a virtual portal with initial content	Virtual Portal Manager	---	X
Listing all virtual portals	Virtual Portal Manager	X	---
Modifying a virtual portal	Virtual Portal Manager	X	---
Deleting a virtual portal	Virtual Portal Manager	X	---

Note: The following two administrative tasks are manual tasks:

- Adding and configuring the user repository for the virtual portal
- Pre-configuring virtual portals

The following sections provide more information about these administrative tasks and how you perform them. The portal configuration tasks for administering virtual portals are documented under *Portal configuration tasks for administering virtual portals*.

"Administering the portal content and resources for virtual portals"
When you create a virtual portal by using the Virtual Portal Manager portlet, the portlet also creates default portal content and resources for the virtual portal. This default content is determined by the default XML script file for initializing virtual portals. In general, you can administer portal resources for a virtual portal just like you do for a normal portal installation.

"Tasks for administering virtual portals" on page 1389
Administering virtual portals and their content comprises the tasks described in the following topics.

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related reference:

"Portal configuration tasks for administering virtual portals" on page 1406
You can use configuration tasks for administering virtual portals.

Related information:

Technotes for virtual portals

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Administering the portal content and resources for virtual portals

When you create a virtual portal by using the Virtual Portal Manager portlet, the portlet also creates default portal content and resources for the virtual portal. This

default content is determined by the default XML script file for initializing virtual portals. In general, you can administer portal resources for a virtual portal just like you do for a normal portal installation.

About this task

You must be aware that some resource types are scoped to a particular virtual portal and cannot be accessed from outside of that virtual portal. Such scoped portal resource types are assigned to only that one portal. Sharing of these resource types between virtual portals is not possible. This restriction is imposed by the portal system and provides a secure isolation between virtual portals. You cannot change this behavior.

Other resource types are not scoped. They are shared among all virtual portals of the same installation. If you want to restrict such resource types to particular virtual portals, you can define their visibility by using Portal Access Control. These access restrictions should usually be defined by the master administrator of the portal installation. For more information about scoping of portal resources for virtual portals, see *Planning for virtual portals* and *Separating and sharing resources between virtual portals*.

Procedure

- To change the content *globally* and *before* creating a virtual portal, modify the default XML script that specifies the initial content for virtual portals. For details about how to do this see *Pre-configuring the default content for virtual portals*.
- To change the content *specifically* and *after* creating a virtual portal, use either of the following portal tools:
  1. Use the Manage Pages portlet of the virtual portal. You can have the subadministrator of the virtual portal do this.
  2. Use the XML configuration interface to import content into the virtual portal. You can use this portal tool only from the initial portal installation.

  Note: When you create a virtual portal, the portlets that are associated with IBM Web Content Manager are not included in the virtual portal, even if you deployed these portlets as part of your original portal installation. To use any of these portlets in a virtual portal, you must manually create a page and add the portlets:
  - Authoring portlet: Select **Web Content Authoring** when you are adding the portlet.
  - Web Content Viewer portlet: Select **Web Content Viewer** when you are adding the portlet.

Related concepts:

"Planning for virtual portals" on page 1366
Before you create your virtual portals, review this information for planning purposes. Determine how many virtual portals your business requires, and how and for which purposes you will use them. Based on your decision, plan how you implement and configure your virtual portals.

"Separating and sharing resources between virtual portals" on page 1367
Separation between virtual portals is achieved by **scoping** the portal resources of the virtual portals. Scoping means making portal resources available uniquely and separately to individual virtual portals and their users.

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal

configurations.

Related tasks:

"Pre-configuring the default content for virtual portals" on page 1400
When you create the virtual portal by using the Virtual Portal Manager portlet, the virtual portal is pre-filled with default content. This default content is determined by the default XML script file for initializing virtual portals.

Related information:


Technotes for virtual portals

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Tasks for administering virtual portals

Administering virtual portals and their content comprises the tasks described in the following topics.

"Managing the users of virtual portals"
You manage the users of virtual portals by adding and configuring the user repository and later administering the users for virtual portals.

"Creating a virtual portal" on page 1391
As a master administrator of the portal installation, you can create virtual portals by using the Virtual Portal Manager portlet.

"Filling a virtual portal with content" on page 1393
When you create a virtual portal by using the Virtual Portal Manager portlet, the portlet fills the new virtual portal with default content.

"Configuring the sub administrators for virtual portals" on page 1394
You can administer the sub administrators of a virtual portal as required by using the Portal Access Control of your initial portal installation.

"Modifying a virtual portal" on page 1397
You can modify a virtual portal by using either the Virtual Portal Manager portlet or the appropriate configuration task.

"Deleting a virtual portal" on page 1397
You can delete a virtual portal by using the Virtual Portal Manager portlet.

Related information:


Technotes for virtual portals

Managing the users of virtual portals:

You manage the users of virtual portals by adding and configuring the user repository and later administering the users for virtual portals.

"Adding and configuring the user repository for a virtual portal" on page 1390
For the user repository for your virtual portals, you have a choice of using either federated security or an LDAP.

"Administering the users for virtual portals" on page 1390
As the master administrator of the portal installation you assign administrative users for the virtual portals. These virtual portal sub-administrators can manage the access rights of the user population of the virtual portal for which they are responsible.

Related information:

Technotes for virtual portals

Adding and configuring the user repository for a virtual portal:

For the user repository for your virtual portals, you have a choice of using either federated security or an LDAP.

When you enable security on a portal installation, by default you have Federated Security as WebSphere Security provider. For details, review the section about Configuring the portal.

Related information:

Technotes for virtual portals

Administering the users for virtual portals:

As the master administrator of the portal installation you assign administrative users for the virtual portals. These virtual portal sub-administrators can manage the access rights of the user population of the virtual portal for which they are responsible.

When you use realms to separate the user populations of the virtual portals from each other, you need to configure the realms manually in a Virtual Member Manager configuration file. This is typically done by the master administrator of the portal installation.

When you create a virtual portal, be aware of the following implications:

- For scoped resources: Access rights that you configured for users on scoped resources of the initial portal installation are not passed on to similar resources of a virtual portal. This statement applies only to resources that are scoped for each virtual portal, such as pages or portlet instances, but not for shared resources. For example, if you restricted access rights to some pages or portlet instances for users on the initial portal installation, these restrictions do not apply for the users of the virtual portal. For more detailed information about scoped resources see *Planning for virtual portals* and *Separating and sharing resources between virtual portals*.
- The All Authenticated Portal Users group and the All Portal User Groups are valid over all portals that share the same realm. This means that users who are in a realm or user group that belongs to more than one virtual portal, these users have the assigned access rights on all virtual portals to which they have access.

If you want to change these default access rights for the users, you can do one of the following:

- To configure the scope of access rights for users *before* creating a virtual portal, configure your realms and user groups accordingly.
- To change the access rights of users of a virtual portal *after* creating a virtual portal, use the Portal Access Control portlets in that virtual portal. You can have the sub-administrators of the virtual portal perform this task.

Related concepts:

"Planning for virtual portals" on page 1366
Before you create your virtual portals, review this information for planning purposes. Determine how many virtual portals your business requires, and how and for which purposes you will use them. Based on your decision, plan how you

implement and configure your virtual portals.

"Separating and sharing resources between virtual portals" on page 1367
Separation between virtual portals is achieved by **scoping** the portal resources of the virtual portals. Scoping means making portal resources available uniquely and separately to individual virtual portals and their users.

Related information:

Technotes for virtual portals

Creating a virtual portal:

As a master administrator of the portal installation, you can create virtual portals by using the Virtual Portal Manager portlet.

When you create virtual portals, you specify the following attributes:

- The **title** of the virtual portal. This title is later displayed in the list of virtual portals in the Virtual Portal Manager portlet. The title is not visible for users of the virtual portal.
- A **description** for the virtual portal. This field is optional.
- Either a host name or a context for the virtual portal:
  - The portal **context** of the virtual portal. The context must be unique. The context is used to create the URL of the virtual portal. This URL is mapped to the actual internal URL of the virtual portal. You can give the friendly URL to the users of the virtual portal. They can use it to access the virtual portal without having to remember the internal URL.
  - A host name for the virtual portal. This attribute is optional. Use it to add a host name of your choice for the virtual portal. The portal uses that host name for the friendly URL of the virtual portal as follows: `http://your_host_name_example:port/wps/portal`. You can pass that friendly URL to your portal users for easier access to your portal.

    Notes:

    - This URL is used internally to access the virtual portal instance, even if you specify a context URL that is easy to use. Make sure that the host name that you specify here is accessible.
    - You cannot use the same virtual portal host name twice in the same portal installation.
    - After you create the virtual portal, you cannot change the host name that you specify for the virtual portal. If you must use a different host name for a virtual portal, see the topic about *Using a new host name for an existing virtual portal*.
    - If you use web content libraries, do not specify a context URL for the new virtual portal that matches the name of a library on your server. If the name of a library and the URL context of a virtual portal have the same value, incorrect rendering of web content can result.

  Notes:
  - You must specify either a host name or a context.
  - If you specify both a host name and a context, the host name takes precedence and the context is ignored.

  - There are some strings that you cannot use as URL mappings for virtual portals, for example `vp`. These strings are reserved names and correspond with URL codec names. For a list of these reserved strings, see *Shaping the user experience*.
  - Use only ASCII characters for the URL Context. For example, you cannot use a URL Context such as `språk`. If you use non-ASCII characters, the portal shows an error message such as the following `EJPAH2009E: Invalid characters were found in a context name or label`. Similarly, you cannot use escaped URL encoding either. For example, a URL Context such as `spr%E5k`.
- The **realm** that represents the user population for the virtual portal. This field is only shown if your portal configuration supports realms.

When you use the Virtual Portal Manager administration portlet to create the virtual portal, you can add the following additional parameters as well:

- The user group of **sub-administrators** who are able to administer the virtual portal.
- The **theme** of the virtual portal.

For details see, *Using the Virtual Portal Manager administration portlet*. As an alternative, you can also use the appropriate configuration task to create virtual portals. For details about the configuration tasks for administering virtual portals see, *Portal configuration tasks for administering virtual portals*. When you use the configuration task for creating a virtual portal, you need to deliver this information by using the XML configuration interface in a later step.

Notes:

1. Before you create a virtual portal, read the information in *Planning for virtual portals*.
2. If you use the configuration task `create-virtual-portal` to create a virtual portal, the virtual portal is created without content. For more information about filling a virtual portal with content, see *Filling a virtual portal with content*.
3. If you do not specify a virtual portal title in the language, which is either defined as the user-preferred language or defined in the user's browser, the display fallback uses the unique name if present, or a string version of the object ID. So, to display the virtual portal title and content root correctly, the administrator must select the preferred language for the portal user. Or the administrator must define the display language in the user's browser, according to the language in which the title is set. For information about the language search sequence, see *Selecting and changing the language*.
4. When you create a virtual portal, a workspace is created that contains a new portal site web content library. All managed pages that are created in the virtual portal are stored in the virtual portal site library. As web content libraries are not shared across virtual portals, such managed pages are visible only within the virtual portal. Syndication of the portal site library for a virtual portal is the same as syndication of any other web content library.

You can pre-configure the content and the sub-administrators for virtual portals. For details see, *Pre-configuring virtual portals*.

Related concepts:

"Shaping the user experience" on page 1384
You can improve the user experience that users have with your virtual portals by using human readable URLs, or by using custom themes and skins for your virtual portals.

"Planning for virtual portals" on page 1366
Before you create your virtual portals, review this information for planning purposes. Determine how many virtual portals your business requires, and how and for which purposes you will use them. Based on your decision, plan how you implement and configure your virtual portals.

Related tasks:

"Using the Virtual Portal Manager administration portlet" on page 1402
After you complete a regular portal installation, the portal is ready and enabled for implementing virtual portals. For improved manageability of virtual portals, WebSphere Portal Express provides a portlet for administering virtual portals. It is named Virtual Portal Manager. It enables the creation of extra virtual portals as you need. You can also use it to list, modify, or delete virtual portals in your portal.

Related reference:

"Portal configuration tasks for administering virtual portals" on page 1406
You can use configuration tasks for administering virtual portals.

"Task: create-virtual-portal" on page 1407
Portal ConfigEngine task that creates a new virtual portal.

"Known limitations for virtual portals" on page 1418
The following sections describe known limitations of virtual portals.

Related information:


Technotes for virtual portals

"Selecting and changing the language" on page 1429
You can control the multiple language-specific settings within the portal.

"Pre-configuring virtual portals" on page 1400
When you use the Virtual Portal Manager portlet to create a virtual portal, the portlet creates the new virtual portal with default portal content and resources. It also creates default access rights for the virtual portal sub-administrators on those resources. You can modify both the default portal content and the default access rights for the sub-administrators *globally* and *before* creating a virtual portal.

Filling a virtual portal with content:

When you create a virtual portal by using the Virtual Portal Manager portlet, the portlet fills the new virtual portal with default content.

This default content is determined by the default XML script file for initializing virtual portals. If you want different content in your virtual portal, you can pre-configure your own custom script file. For more information, about the default content for virtual portals and how you configure it before creating a virtual portal, see *Pre-configuring the default content for virtual portals*. For more information, about how you can add more portal content, see *Administering the portal content and resources for virtual portals*. You can also use the portal XML configuration interface to add content to a virtual portal.

For more information, see the *Virtual portals command reference*. For more information, about the XML configuration interface and how to use it, see the *XML configuration interface*.

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related tasks:

"Pre-configuring the default content for virtual portals" on page 1400
When you create the virtual portal by using the Virtual Portal Manager portlet, the virtual portal is pre-filled with default content. This default content is determined by the default XML script file for initializing virtual portals.

"Administering the portal content and resources for virtual portals" on page 1387
When you create a virtual portal by using the Virtual Portal Manager portlet, the portlet also creates default portal content and resources for the virtual portal. This default content is determined by the default XML script file for initializing virtual portals. In general, you can administer portal resources for a virtual portal just like you do for a normal portal installation.

Related reference:

"Virtual portals command reference" on page 1406
You can configure virtual portals by using portal configuration tasks or the XML configuration interface.

Related information:


Technotes for virtual portals

Configuring the sub administrators for virtual portals:

You can administer the sub administrators of a virtual portal as required by using the Portal Access Control of your initial portal installation.

When you create a virtual portal by using the Virtual Portal Manager portlet, you select a user group of sub administrators. The sub administrators who you want to be responsible for the administration of the new virtual portal. During creation of the new virtual portal, the Virtual Portal Manager portlet creates a set of necessary access permissions on the virtual portal for the sub administrator group that you specified. This action includes EDITOR role access permissions on the administration portlets that are part of a virtual portal. As a result, the sub administrators of a virtual portal can do administrative tasks on the virtual portal with these administration portlets.

If you want to change the default access permissions for the subadministrators, use one of the following actions:

- If you want to change the default Editor access permission for the subadministrators on the administrative portlets or the list of portlets *globally* and *before* you create virtual portals, configure the Virtual Portal Manager portlet accordingly. For details about how to do this see *Pre-configuring the subadministrators for virtual portals*.
- If you want to assign extra access permissions to the subadministrators *specifically* and *after* creating a virtual portal, use the master administrator user ID of your initial portal installation and modify those access permissions for them manually in Portal Access Control. To do this, you can use the User and Group Permissions portlet, the Resource Permissions portlet, the XML configuration interface, or the Portal Scripting Interface. The consequences differ, depending on where you make the updates:
  - If you do this in the initial portal installation, you can change the access permissions for the subadministrators on the virtual portal as a whole.
  - If you do this in the virtual portal itself, you can change the access permissions for the subadministrators on the individual resources of the virtual portal.

Assigning additional access permissions to the sub administrators

Depending on the usage of your virtual portals, you might have to give the sub administrators extra access permissions on specific resources.

Note: Do not grant the sub administrators of virtual portals the access permissions to do any installation-related tasks, such as installation of portlets or themes. An unstable or malicious portlet that is installed in one virtual portal can destabilize the entire portal installation, as all virtual portals share Java virtual machine. Typically, only the master administrator of the portal installation can do installation-related tasks.

The following list shows the tasks for which you can assign extra access permissions to subadministrators of virtual portals. It also specifies whether an access permission is scoped to the virtual portal or if it is global to the entire portal installation, including all virtual portals. You can assign the permissions for these tasks to subadministrators only by using use the master administrator user ID of your initial portal installation.

Granting access permissions to users and groups of virtual portals
This task requires one of the following access permissions:

- Delegator on the group that defines the users of the virtual portal. This way is the preferred option, as the access permission is limited to the virtual portal.
- Delegator@Groups or Delegator@Users. Both of these access permissions apply globally to the entire portal installation, including all virtual portals.

Cloning portlet applications, for example, the web clipping portlet
This task requires Editor@Portlet Application. This access permission applies globally to the entire portal installation, including all virtual portals.

Access permissions for policies.
To manage policies, subadministrators need different access permissions, depending on the task that you want the subadministrative user to be able to complete. For example, to delete policies, a subadministrator needs Manager@Policy and User@Business Rules. This access permission is the highest permission. These access permissions apply globally to the entire portal installation, including all virtual portals.

Using the XML configuration interface
This task requires Security Administrator@Portal and Editor@XML access. These access permissions apply globally to the entire portal installation, including all virtual portals.

Managing portal search collections
This task requires Editor@Virtual Resource PSE_SOURCES. This access permission applies globally to the entire portal installation, including all virtual portals.

Managing URL mappings
This task requires Editor@parent context for parent mappings and Manager@context for URL mappings. These access permissions apply globally to the entire portal installation, including all virtual portals.

Managing tags and ratings
This task requires Manager@Tags and Manager@Ratings. These access permissions apply globally to the entire portal installation, including all virtual portals.

Managing personalization rules
This task requires the following access permissions:

- Privileged User on the following portlet applications:
  - Personalization Editors
  - Personalization Navigator
  - Personalization Picker
- Manager@the Personalization Rule

These access permissions apply globally to the entire portal installation, including all virtual portals.

Granting virtual portal administrators access to web content libraries
Virtual portal administrators do not automatically have access to work with web content libraries when they are using the administration portlet. To enable a virtual portal administrator to work with web content libraries, you must assign them access to either the JCR content root node or individual web content libraries:

- You can assign virtual portal administrators access to the JCR content root node by using the `Set access on root` button in the Web Content Library view of the Administration portlet. For more information, see *Setting root access for all web content libraries* in the Portal Content help.
  - Assign virtual portal administrators administrator access to the JCR content root node to enable them to create new libraries and view, edit, and delete all existing libraries.
  - Assign virtual portal administrators contributor access to the JCR content root node to enable them to create new libraries and view, edit, and delete libraries that they created.
- You can also assign virtual portal administrators access to libraries they did not create by editing the access settings of individual libraries.

Templating sample content is provided by default with WebSphere Portal Express. This sample content is available from the Create Content tab of the site toolbar. If you want to use the sample content with a specific virtual portal, you must syndicate the following web content libraries to the virtual portal:

- `Template Page Content 3.0`
- `Web Content Templates 3.0`

If you fail to syndicate these libraries, the portal shows an error when you add the sample content to a page.

The configuration task `create-virtual-portal` does not assign roles to the sub administrators of the virtual portal. In this case, you assign the required roles manually by using the portal access administration portlets or by using the portal XML configuration interface. For more information about the XML configuration interface and how to use it see, *The XML configuration interface*.

Related concepts:

"Virtual portal roles and their capabilities" on page 1375
A typical virtual portal scenario works with a master administrator and sub-administrators. Assigning access permissions to the users of virtual portals also

requires special considerations.

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related tasks:

"Pre-configuring the subadministrators for virtual portals" on page 1402
You can configure the roles and access rights that are assigned to subadministrators on portlets of a virtual portal *globally* and *before* you create a virtual portal. The following steps must be completed on your initial portal installation.

Related reference:

"Task: assign-virtual-portal-admin-group" on page 1409
Portal ConfigEngine task that assigns a group of administrators to a virtual portal.

Related information:

Technotes for virtual portals

Modifying a virtual portal:

You can modify a virtual portal by using either the Virtual Portal Manager portlet or the appropriate configuration task.

You can use the Virtual Portal Manager portlet to change the following settings of an existing virtual portal:

- The title of the virtual portal.
- The description of the virtual portal.
- The realm of the virtual portal.

  Note: Apply special care when you are changing the realm of a virtual portal. Changing the realm of a virtual portal might change the users and groups who have access to the virtual portal and to resources of that virtual portal. This includes the possibility that the sub-administrator of the virtual portal can lose the rights to administer the portal. If this happens, change the realm back to the original realm by using the **initial** portal installation as an administrative interface.
- The theme of the virtual portal.

Related reference:

"Task: modify-virtual-portal" on page 1411
Portal ConfigEngine task that modifies a virtual portal.

Related information:

Technotes for virtual portals

Deleting a virtual portal:

You can delete a virtual portal by using the Virtual Portal Manager portlet.

Note: You cannot delete the initial portal installation.
After the virtual portal resource is deleted, the scoped resources of that particular virtual portal are deleted later by a scheduled cleanup service. The URL mapping that was created when the virtual portal was created is also deleted. The following resources are not deleted:

- The unscoped resources that were available in the virtual portal; they belong to the initial portal installation and are therefore not deleted.
- Extra URL mappings which administrators might have created manually are not deleted.

Note: If you delete a virtual portal and you want to create a new virtual portal immediately after the deletion by using the same URL context, you do not have to wait for the scheduled cleanup service. Run the cleanup task for deleting resources by running the XML script `Task.xml` of the XML configuration interface. Then, you can create the new virtual portal. For details, see *Working with the XML configuration interface*.

Preparing the deletion of virtual portals

Virtual portals can be created in environments where multiple portal installations share database domains, such as Community or Customization. This environment can be in a staging environment or for different lines of production. In this case some portal resources, such as page customization within a virtual portal, can be visible in several of these installations. If you delete a virtual portal, then this portal installation cannot determine whether the corresponding resources can be deleted or if they are still valid in the context of other portal installations and must be preserved. It is the responsibility of the portal administrator to decide whether clean-up of related resources that are in shareable database domains is required before a virtual portal is deleted. The administrator must decide whether these resources are obsolete or still in use. You can complete the cleanup in two ways:

- Manual cleanup of virtual portal resources in shared database domains. To clean up manually, proceed as follows:
  1. Connect to the virtual portal that you want to delete.
  2. Clean up the two database domains Community and Customization. Run the XML configuration scripts `DeleteSharedCommunityContent.xml` and `DeleteSharedCustomizationContent.xml`.
  3. Delete the virtual portal.
- Automated cleanup of virtual portal resources in shared database domains. Run the configuration task `delete-virtual-portal` and specify either the URL context or the host name of the virtual portal that you want to delete. Example syntax for both options:
  - For IBM i:

```
- ConfigEngine.sh delete-virtual-portal
                  -DremoveResourcesInSharedDomains=true
                  -DVirtualPortalContext=URL_context_of_the_VP
- ConfigEngine.sh delete-virtual-portal
                 -DremoveResourcesInSharedDomains=true
                 -DVirtualPortalHostName=host_name_of_the_VP
```

  - For Linux:

```
- ConfigEngine.sh delete-virtual-portal
                  -DremoveResourcesInSharedDomains=true
                  -DVirtualPortalContext=URL_context_of_the_VP
- ConfigEngine.sh delete-virtual-portal
                 -DremoveResourcesInSharedDomains=true
                 -DVirtualPortalHostName=host_name_of_the_VP
```

  - For Windows:

```
- ConfigEngine.bat delete-virtual-portal
                   -DremoveResourcesInSharedDomains=true
                   -DVirtualPortalContext=URL_context_of_the_VP
```

```
- ConfigEngine.bat delete-virtual-portal
                  -DremoveResourcesInSharedDomains=true
                  -DVirtualPortalHostName=host_name_of_the_VP
```

For a list of database domains that can be shared, read the topic about Shared database domains and sharing database domains for your environment.

Cleaning up remaining resources if a virtual portal has been deleted already

If a virtual portal was deleted already without prior cleanup of resources in shareable database domains, and if these resources cannot be accessed by any other portal installation that shares the database domains, proceed by the following steps to remove the remaining resources:

1. Run the cleanup task for deleting resources by running the XML script `Task.xml` of the XML configuration interface.
2. Re-create the virtual portal that was deleted by using the identical URL mapping.
3. Follow the instructions to delete a virtual portal, including cleanup of resources in shareable database domains. This cleanup was described in the preceding section about Preparing for the deletion of virtual portals.

Related reference:

"Task: delete-virtual-portal" on page 1412
Portal ConfigEngine task that deletes a virtual portal.

Related information:

Technotes for virtual portals

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Working with the Virtual Portal Manager portlet

For improved manageability of virtual portals, WebSphere Portal Express provides an administration portlet. It is named **Virtual Portal Manager**. It allows you to create virtual portals on demand. You can also use it to pre-configure and administer virtual portals.

"Pre-configuring virtual portals" on page 1400
When you use the Virtual Portal Manager portlet to create a virtual portal, the portlet creates the new virtual portal with default portal content and resources. It also creates default access rights for the virtual portal sub-administrators on those resources. You can modify both the default portal content and the default access rights for the sub-administrators *globally* and *before* creating a virtual portal.

"Using the Virtual Portal Manager administration portlet" on page 1402
After you complete a regular portal installation, the portal is ready and enabled for implementing virtual portals. For improved manageability of virtual portals, WebSphere Portal Express provides a portlet for administering virtual portals. It is named Virtual Portal Manager. It enables the creation of extra virtual portals as you need. You can also use it to list, modify, or delete virtual portals in your portal.

Related information:

Technotes for virtual portals

Pre-configuring virtual portals

When you use the Virtual Portal Manager portlet to create a virtual portal, the portlet creates the new virtual portal with default portal content and resources. It also creates default access rights for the virtual portal sub-administrators on those resources. You can modify both the default portal content and the default access rights for the sub-administrators *globally* and *before* creating a virtual portal.

"Pre-configuring the default content for virtual portals"
When you create the virtual portal by using the Virtual Portal Manager portlet, the virtual portal is pre-filled with default content. This default content is determined by the default XML script file for initializing virtual portals.

"Pre-configuring the subadministrators for virtual portals" on page 1402
You can configure the roles and access rights that are assigned to subadministrators on portlets of a virtual portal *globally* and *before* you create a virtual portal. The following steps must be completed on your initial portal installation.

Related information:

 Technotes for virtual portals

Pre-configuring the default content for virtual portals:

When you create the virtual portal by using the Virtual Portal Manager portlet, the virtual portal is pre-filled with default content. This default content is determined by the default XML script file for initializing virtual portals.

About this task

This file can have different names, for example, `InitVirtualPortal.xml`, or `InitVirtualContentPortal.xml`, or `InitAdminVirtualPortal.xml`, depending on your portal installation. It is in a WebSphere Application Server asset named `VirtualPortal.zip`. To get to this file, access the WebSphere Integrated Solutions Console, select **Applications** > **Application Types** > **Assets**, and locate the file in the list.

To find out which of these xml files is used for creating virtual portals in your installation, select **Manage Virtual Portals** to open **Virtual Portal Manager** portlet and select the option **Edit Shared Settings** from the portlet menu. This shows the WebSphere Application Server asset name and the XML file name inside that asset. To find out which content virtual portals have that you create, review the XML script file under the location that was given earlier.

Advanced master administrators can customize the default content for virtual portals as required by modifying or replacing the XML script that specifies the initial content for virtual portals.

Note: When you modify or replace this XML script, plan ahead and apply special care. You can add or remove some content to enhance or reduce the functionality of a virtual portal to a certain extent. The following portal resources are mandatory content of a virtual portal and must be included in a customized XML initialization script for virtual portals:

- Content Root (`wps.content.root`)
- Login (`wps.Login`)
- Administration (`ibm.portal.Administration`).

Depending on the functionality that you want to make available, more content is required. For example, to allow templating. Include Application Root (`wps.application.root`) and Templates (`ibm.portal.Templates`).

Procedure

1. Add your custom XML script to a WebSphere Application Server asset. To make this addition, proceed as follows:
   a. Export the file `VirtualPortal.zip` from your portal server.
   b. Log in to the WebSphere Integrated Solutions Console.
   c. Click **Applications** > **Application Types** > **Assets**.
   d. Select `VirtualPortal.zip`.
   e. Click **Export**.
   f. Make a copy of the exported `.zip` file, and name the copy `my_VirtualPortal.zip` or similar.
   g. Add your custom virtual portal script to your copy of the `.zip` file. For example, the script file can be `my_InitVirtualPortal.xml`. To add the script, use a utility program for `.zip` files.
   h. Log in to the WebSphere Integrated Solutions Console.
   i. Click **Applications** > **Application Types** > **Assets**.
   j. Click **Import**.
   k. Select your updated copy of the `.zip` file, for example `my_VirtualPortal.zip` .
   l. Click **Next** > **Next** > **Finish**.
   m. Save your changes to the master configuration.
2. Open the Manage Virtual Portals portlet by clicking the **Administration menu** icon. Then, click **Virtual Portals** > **Manage Virtual Portals**.
3. Open the **Manage Virtual Portals** portlet menu by clicking the dropdown arrow and select the option **Edit Shared Settings**.
4. Edit the SCRIPT_INIT_VP parameter of the portlet. Replace the current value with the name of your custom XML script and custom asset compressed (.zip) file. You can specify this attribute as a file inside a WebSphere Application Server asset by using a syntax such as this:

   `WebSphere:assetname=`*`my_VirtualPortal.zip`*`:`*`my_InitVirtualPortalScript.xml`*

   where `my_VirtualPortal.zip` is the name of your asset and `my_InitVirtualPortalScript.xml` is the name of a file inside the asset. Do not update the default asset `VirtualPortal.zip` installed with WebSphere Portal Express. Instead, create and maintain a separate second asset independent of the default asset `VirtualPortal.zip`.
5. Optional: If you want to create only an empty virtual portal with no content, you can specify the value for this parameter as follows: `WebSphere:assetname=VirtualPortal.zip:InitEmptyVirtualPortal.xml`.
6. Click **OK** twice to save your changes.

Related information:

Technotes for virtual portals

Pre-configuring the subadministrators for virtual portals:

You can configure the roles and access rights that are assigned to subadministrators on portlets of a virtual portal *globally* and *before* you create a virtual portal. The following steps must be completed on your initial portal installation.

Procedure

1. Click the **Administration menu** icon. Then, click **Portlet Management** > **Portlets**. Then, click **Manage Portlets** .
2. In the list of portlets, locate the Virtual Portal Manager portlet.
3. Click the **Configure Portlet** (wrench) icon of the Virtual Portal Manager portlet.
4. Perform the following steps, depending on the requirements for your virtual portals:
   a. If you want to change the list of portlets, to which the subadministrators have access:
      1) Edit the **portletListNeedAccess** parameter of the portlet. Remove those portlets for which you want the subadministrators of your virtual portals to have no access rights.
      2) Add portlets as required by adding the unique names of the portlets to the list. You might need to take a note of the list and remove the parameter, and then enter the parameter with your updated list.

      The default list contains all portlets that are listed under Content of a virtual portal.
   b. If you want to change the access rights that are granted to the subadministrators on the portlets of virtual portals, edit the **actionSetName** parameter of the portlet and change the role that you want to assign to the subadministrators to the role that fits your requirements. The default role is EDITOR. You might need to take a note of the parameter and remove it, and then reenter the parameter with the updated value. You can enter the following values: Administrator, Security Administrator, Delegator, Manager, Privileged User, User.

      Note: Subadministrators have the roles that you assign to them on *all* the portlets that are listed under the **portletListNeedAccess** parameter of the Virtual Portal Manager portlet (see the previous step).
5. Click **OK** to save your changes.
6. Assign the virtual portal subadministrators administrator rights on the JCR root as required.

Related information:

Technotes for virtual portals

Using the Virtual Portal Manager administration portlet

After you complete a regular portal installation, the portal is ready and enabled for implementing virtual portals. For improved manageability of virtual portals, WebSphere Portal Express provides a portlet for administering virtual portals. It is named Virtual Portal Manager. It enables the creation of extra virtual portals as you need. You can also use it to list, modify, or delete virtual portals in your portal.

About this task

Creating a virtual portal

When you create a new virtual portal, you enter or select the following properties for the new virtual portal as required:

- The title for the new virtual portal. The title is later displayed in the list of virtual portals in the Virtual Portal Manager portlet. The title is not visible for users of the virtual portal. This field is limited to 255 characters.
- A description of the new virtual portal. This attribute is optional. This field is limited to 255 characters.
- Either a host name or a context for the virtual portal:
  - A human readable URL context that is used for accessing the virtual portal. You can set a URL that can be easily remembered and is therefore more easy to use than the actual full portal URL. The portal maps the friendly URL to the internal URL of the virtual portal. To do this mapping, it uses the portal URL Mapping feature. The string that you enter is used as the last part of the URL of the virtual portal and is appended to `http://www.example.com/wps/portal/`.

    Notes:

    - The URL context for each virtual portal must be unique.
    - All virtual portal URL contexts must be built from the root context for the portal server and must be unique. They cannot be subcontexts. For example, this URL is invalid:

      ```
      http://www.example.com/wps/portal/vp1/vp2
      ```

      This is the correct format:

      ```
      http://www.example.com/wps/portal/vp2
      ```
    - Use only ASCII characters for the URL Context of the virtual portal. Non-ASCII characters are not allowed for the URL Context. Examples: `språk` or **Düne**. Using non-ASCII characters results in an error message such as the following one:

      ```
      EJPAH2009E: Invalid characters were found in a context name or label
      ```

      Similarly, do not use escaped URL encoding either. For example, a URL Context of `spr%E5k` is not allowed.
  - A host name for the virtual portal. This attribute is optional. Use it to add a host name of your choice for the virtual portal. The portal uses that host name for the friendly URL of the virtual portal as follows: `http://`*`your_host_name_example`*`:`*`port`*`/wps/portal`. You can pass that friendly URL to your portal users for easier access to your portal.

    Notes:

    - This URL is used internally to access the virtual portal instance, even if you specify a context URL that is easy to use. Make sure that the host name that you specify here is accessible.
    - You cannot use the same virtual portal host name twice in the same portal installation.
    - After you create the virtual portal, you cannot change the host name that you specify for the virtual portal. If you must use a different host name for a virtual portal, see the topic about *Using a new host name for an existing virtual portal*.

- If you use web content libraries, do not specify a context URL for the new virtual portal that matches the name of a library on your server. If the name of a library and the URL context of a virtual portal have the same value, incorrect rendering of web content can result.

Notes:

– You must specify either a host name or a context.
– If you specify both a host name and a context, the host name takes precedence and the context is ignored.
– There are some strings that you cannot use as URL mappings for virtual portals, for example `vp`. These strings are reserved names and correspond with URL codec names. For a list of these reserved strings, see *Shaping the user experience*.
– Use only ASCII characters for the URL Context. For example, you cannot use a URL Context such as `språk`. If you use non-ASCII characters, the portal shows an error message such as the following `EJPAH2009E: Invalid characters were found in a context name or label`. Similarly, you cannot use escaped URL encoding either. For example, a URL Context such as `spr%E5k`.

- The Virtual Member Manager realm that contains the user population of the virtual portal. This entry field is only shown if your portal installation supports realms. If you leave the realm field blank, the user population is the same as for the default realm. If you use a single common user repository for all virtual portals, for example the WebSphere Application Server LDAP custom user registry, this entry field is not shown.
- The initial subadministrator user group for the virtual portal. The portal gives this group administrator access rights on the Content Root node of the new virtual portal and all its child pages. This group must be within the realm that is specified for the virtual portal.
- The Default theme that is applied to the pages of the virtual portal. You select the default theme from a pull-down list. By the default configuration for virtual portals a subadministrator cannot change that default theme, unless you change the roles and access rights give to the subadministrator.

After you enter this information, you create the new virtual portal. With that information the portlet triggers a sequence of processes to establish the new virtual portal. These processes include:

- Creating a new root content node for the virtual portal.
- Creating the new URL mapping to point to the new root content node.
- Assigning the selected theme to the new root content node.
- Granting the specified administrator group the action set for the Administrator role on the new root content node and on the new virtual portal.
- Calling the XML configuration interface script to create the initial content tree. This includes virtual portal specific instances of the following portal resources: Favorites, Administration, Home, Manage Portlets, and Page Customizer with the corresponding concrete portlets. To change the content *globally* and *before* creating a virtual portal, modify

the XML script that specifies the initial content for virtual portals. For details about how to do this see *Pre-configuring the default content for virtual portals*.

- Assigning default roles and access rights to subadministrators and users on the created resources.

Modifying or editing a virtual portal
You can modify the title and description of the virtual portal. You can also set locale-specific titles and descriptions.

Reinitializing a virtual portal
Applies the `InitVirtualPortal.xml` script again and re-creates the default content of a virtual portal. If you replaced the default XML script with your own and configured the Virtual Portal Manager portlet accordingly, your custom script is reapplied. Resources that you removed from the default content are re-created.

Note: Resources that you added to the default content remain in the virtual portal.

Deleting a virtual portal
Deletes the virtual portal, its initial URL mapping, and all the corresponding scoped resources.

Note: This does not delete the unscoped resources from the initial portal installation or extra URL mappings that administrators created.

Related concepts:

"Shaping the user experience" on page 1384
You can improve the user experience that users have with your virtual portals by using human readable URLs, or by using custom themes and skins for your virtual portals.

Related tasks:

"Pre-configuring the default content for virtual portals" on page 1400
When you create the virtual portal by using the Virtual Portal Manager portlet, the virtual portal is pre-filled with default content. This default content is determined by the default XML script file for initializing virtual portals.

Related reference:

"Using a new host name for an existing virtual portal" on page 1419
If you want to use a new host name for an existing virtual portal, you must delete the virtual portal. Then, re-create it with the new host name.

Related information:

Technotes for virtual portals

Virtual portals reference

The virtual portals reference provides information about using commands for configuring virtual portals, usage hints and tips, and known limitations.

"Virtual portals command reference" on page 1406
You can configure virtual portals by using portal configuration tasks or the XML configuration interface.

"Hints and tips for working with virtual portals" on page 1416
View information about configuring virtual portals, hints and tips, and known limitations in WebSphere Portal Express Version 8.5:

"Known limitations for virtual portals" on page 1418
The following sections describe known limitations of virtual portals.

Related information:

Technotes for virtual portals

Virtual portals command reference

You can configure virtual portals by using portal configuration tasks or the XML configuration interface.

"Portal configuration tasks for administering virtual portals"
You can use configuration tasks for administering virtual portals.

"Using the XML configuration interface to work with virtual portals" on page 1414
You can export and import the contents of individual virtual portals by using the XML configuration interface. For example, you can use the XML configuration interface to fill a newly created virtual portal with content. As each virtual portal has its own globally unique portal ID, you can determine all resources that are associated with that virtual portal clearly and individually.

Related information:

Technotes for virtual portals

Portal configuration tasks for administering virtual portals:

You can use configuration tasks for administering virtual portals.

WebSphere Portal Express provides the following configuration tasks that you can use to perform the following work with virtual portals:

- Create virtual portals
- List all virtual portals
- Modify a virtual portal
- Delete a virtual portal.

Before you use these configuration tasks, read the topics about *Planning for virtual portals* and *Administering virtual portals* carefully. For details about the configuration program and about how to use the configuration tasks see *Configuring*.

You pass the parameters in the parameter list for each configuration task as appropriate. You do this by either of the following methods:

- Specifying the parameter and value that is preceded by `-D` on the command line.
- Defining them in the file `wkplc.properties`.

Note: The property file must be encoded in the ISO 8859-1 character encoding format.

"Task: create-virtual-portal" on page 1407
Portal ConfigEngine task that creates a new virtual portal.

"Task: assign-virtual-portal-admin-group" on page 1409
Portal ConfigEngine task that assigns a group of administrators to a virtual portal.

"Task: list-all-virtual-portals" on page 1410
Portal ConfigEngine task that lists all virtual portals.

"Task: modify-virtual-portal" on page 1411
Portal ConfigEngine task that modifies a virtual portal.

"Task: delete-virtual-portal" on page 1412
Portal ConfigEngine task that deletes a virtual portal.

"Using a single configuration task to administer multiple virtual portals" on page 1413
You can administer multiple virtual portals by running a single configuration command. The following configuration tasks support working with multiple virtual portals: `create-virtual-portal`, `delete-virtual-portal`, and `modify-virtual-portal`. Use the `-DvirtualPortalList` parameter with task to create, delete, or modify multiple virtual portals at the same time.

Related concepts:

"Planning for virtual portals" on page 1366
Before you create your virtual portals, review this information for planning purposes. Determine how many virtual portals your business requires, and how and for which purposes you will use them. Based on your decision, plan how you implement and configure your virtual portals.

Related tasks:

Chapter 5, "Configuring," on page 231
Run the following tasks after you install and deploy IBM WebSphere Portal Express. They address tasks that are typically run one time and have a global effect. Some configuration changes are made more frequently or do not have a global effect. These tasks are addressed in the Administering section.

Related information:

 Technotes for virtual portals

"Administering virtual portals" on page 1386
View information to help you scope your WebSphere Portal Express to have multiple virtual portals.

Task: create-virtual-portal:

Portal ConfigEngine task that creates a new virtual portal.

Use the **`create-virtual-portal`** task to create a new virtual portal. This task creates the virtual portal itself, but it does not create any default content for the virtual portal or grant any access permissions to the virtual portal administrators. You need to run these tasks separately after you create the virtual portal. For example, you can use the XML configuration interface create content and grant access permissions. For details, about the access permissions that are required for virtual portal administrators, see the section *Configuring the sub-administrators for virtual portals.*. For details about how to use the XML configuration interface see *The XML configuration interface*.

Parameters

`VirtualPortalTitle`
The title of the virtual portal.

This option is required.

`VirtualPortalRealm`
The realm of the virtual portal.

This input is required if you have realms that are enabled in your portal installation. If you do not have realms that are enabled, do not specify a value for the realm.

Virtual portal context or host name

You can address virtual portals by using either a portal context or a virtual host name. If you specify a host name, the context is ignored:

VirtualPortalContext

The portal context of the virtual portal. The context must be unique.

If you set the host name parameter that is described later, the `VirtualPortalContext` parameter is ignored.

Either the host name or the context option is required.

VirtualPortalHostName

The host name of the virtual portal.

If you specify the host name, the portal uses the host name, and the `VirtualPortalContext` parameter is ignored.

You cannot use the same virtual portal host name twice in the same portal installation.

After you create the virtual portal, you cannot change the host name that you specified for the virtual portal.

Either the host name or the context option is required.

VirtualPortalObjectId

The Object ID is used to reference the virtual portal. If VirtualPortalObjectId is not specified, a new VirtualPortalID is generated.

VirtualPortalNlsFile

The national language support (NLS) file for the virtual portal. Provide the path and file name of your national language support file. This input is optional.

You can create your own national language support file to specify more titles and descriptions in other languages for your virtual portal.

If you specify an NLS file, the value that is given for the virtual portal title in that NLS file overrides the title that you specify by the `VirtualPortalTitle` input parameter.

If you specify an NLS file, do not use prefixes in that NLS file.

If you do not specify an NLS file, the virtual portal is created with the title that you give as the value to the `VirtualPortalTitle` input parameter only. But the virtual portal is created without titles in other languages and without descriptions. If you specify no NLS file, the value that you specify by the `VirtualPortalTitle` input parameter shows to users as the title of the virtual portal in all language environments. This action is independent of the system and browser locale and the user preferences.

If you want to pass a description for the virtual portal to the configuration task, you must specify this action in the NLS file.

For more information about the NLS file and its format, see the *Command reference for the Portal Scripting Interface* and then under the section *Property File Format*.

For a list of the languages that are available with WebSphere Portal Express and their language codes refer to *Language support*.

Syntax

You pass the parameters in the parameter list for each configuration task as appropriate. You do this by either of the following methods:

- Specifying the parameter and value that is preceded by `-D` on the command line.
- Defining them in the file `wkplc.properties`.

Note: The property file must be encoded in the ISO 8859-1 character encoding format.

IBM i

```
ConfigEngine.sh create-virtual-portal -DWasPassword=password
-DPortalAdminPwd=password -DVirtualPortalTitle=MyVirtualPortalTitle
-DVirtualPortalContext=virtual_portal_context_url
```

Linux

```
./ConfigEngine.sh create-virtual-portal -DWasPassword=password
-DPortalAdminPwd=password -DVirtualPortalTitle=MyVirtualPortalTitle
-DVirtualPortalContext=virtual_portal_context_url
```

Windows

```
ConfigEngine.bat create-virtual-portal -DWasPassword=password
-DPortalAdminPwd=password -DVirtualPortalTitle=MyVirtualPortalTitle
-DVirtualPortalContext=virtual_portal_context_url
```

Related concepts:

"Virtual portal roles and their capabilities" on page 1375
A typical virtual portal scenario works with a master administrator and sub-administrators. Assigning access permissions to the users of virtual portals also requires special considerations.

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

"Language support" on page 1419
To reach as many users as possible, WebSphere Portal Express supports different languages for different locations. For instance, a large, international corporation might address users in different countries or regions through multilingual Web sites. In this context the portal can concurrently serve portal views to large numbers of users, each in the user's preferred language.

Related reference:

"Command reference for the Portal Scripting Interface" on page 1126
The IBM WebSphere Portal Express Scripting Interface component provides a scripting interface for the administration functions.

Related information:

Technotes for virtual portals

Task: assign-virtual-portal-admin-group:

Portal ConfigEngine task that assigns a group of administrators to a virtual portal.

Use this task after the creation of a virtual portal to assign a subadministrator group to the virtual portal. This task gives this group permission on the root content node of the virtual portal and all its child pages. The task also gives Editor role access to the administration portlets of the virtual portal. This group must be within the realm that is specified for the virtual portal.

Parameters

VirtualPortalAdminGroup
The full DN (distinguished name) of the administrator group.

Virtual portal context or host name
You address the virtual portal by using either the portal context or the virtual host name, depending on what you specified when you created the virtual portal. If you specified both, specify the host name here.

VirtualPortalContext
The portal context of the virtual portal.

VirtualPortalHostName
The host name of the virtual portal.

Syntax

You pass the parameters in the parameter list for each configuration task as appropriate. You do this by either of the following methods:

- Specifying the parameter and value that is preceded by `-D` on the command line.
- Defining them in the file `wkplc.properties`.

Note: The property file must be encoded in the ISO 8859-1 character encoding format.

IBM i

```
ConfigEngine.sh assign-virtual-portal-admin-group
-DVirtualPortalAdminGroup=cn=vpadmins,o=defaultwimfilebasedrealm
-DVirtualPortalContext=mynewvp -DPortalAdminPwd=password
-DWasPassword=password
```

Linux

```
./ConfigEngine.sh assign-virtual-portal-admin-group
-DVirtualPortalAdminGroup=cn=vpadmins,o=defaultwimfilebasedrealm
-DVirtualPortalContext=mynewvp -DPortalAdminPwd=password
-DWasPassword=password
```

Windows

```
ConfigEngine.bat assign-virtual-portal-admin-group
-DVirtualPortalAdminGroup=cn=vpadmins,o=defaultwimfilebasedrealm
-DVirtualPortalContext=virtual_portal_context_url
-DPortalAdminPwd=password -DWasPassword=password
```

Note: You can create a virtual portal and assign the administrator group in the same command. For example, you can specify it as: `./ConfigEngine.sh create-virtual-portal assign-virtual-portal-admin-group -DVirtualPortalAdminGroup=cn=vpadmins,o=defaultwimfilebasedrealm -DVirtualPortalContext=virtual_portal_context_url -DVirtualPortalTitle=mynewvptitle -DPortalAdminPwd=password -DWasPassword=password`

Related information:

Technotes for virtual portals

Task: list-all-virtual-portals:

Portal ConfigEngine task that lists all virtual portals.

Use this task to list all your virtual portals, together with the following information:

- The title of the virtual portal
- The description of the virtual portal
- The realm of the virtual portal
- The object ID of the virtual portal.

Parameters

This task does not take any parameters.

Syntax

IBM i

```
ConfigEngine.sh list-all-virtual-portals -DPortalAdminPwd=password
-DWasPassword=password
```

Linux

```
./ConfigEngine.sh list-all-virtual-portals -DPortalAdminPwd=password
-DWasPassword=password
```

Windows

```
ConfigEngine.bat list-all-virtual-portals -DPortalAdminPwd=password
-DWasPassword=password
```

Related information:

Technotes for virtual portals

Task: modify-virtual-portal:

Portal ConfigEngine task that modifies a virtual portal.

Parameters

Use this task to modify a virtual portal by using its object ID. To determine the correct object ID of the virtual portal, use the task `list-all-virtual-portals`.

VirtualPortalObjectId
The object ID of the virtual portal. This input is mandatory for identification of the virtual portal that you want to modify.

VirtualPortalRealm
The realm of the virtual portal. You can and must specify a realm only if you have realms that are enabled in your portal installation.

VirtualPortalNlsFile
The national language support (NLS) file for the virtual portal. Provide the path and file name of your national language support file. This input is optional.

You can create your own national language support file to specify more titles and descriptions in other languages for your virtual portal.

If you specify an NLS file, the value that is given for the virtual portal title in that NLS file overrides the title that you specify by the `VirtualPortalTitle` input parameter.

If you specify an NLS file, do not use prefixes in that NLS file.

If you do not specify an NLS file, the virtual portal is created with the title that you give as the value to the `VirtualPortalTitle` input parameter only. But the virtual portal is created without titles in other languages and without descriptions. If you specify no NLS file, the value that you specify by the `VirtualPortalTitle` input parameter shows to users as the title of the virtual portal in all language environments. This action is independent of the system and browser locale and the user preferences.

If you want to pass a description for the virtual portal to the configuration task, you must specify this action in the NLS file.

For more information about the NLS file and its format, see the *Command reference for the Portal Scripting Interface* and then under the section *Property File Format*.

For a list of the languages that are available with WebSphere Portal Express and their language codes refer to *Language support*.

Syntax

You pass the parameters in the parameter list for each configuration task as appropriate. You do this by either of the following methods:

- Specifying the parameter and value that is preceded by `-D` on the command line.
- Defining them in the file `wkplc.properties`.

Note: The property file must be encoded in the ISO 8859-1 character encoding format.

IBM i

```
ConfigEngine.sh modify-virtual-portal -DPortalAdminPwd=password
-DWasPassword=password
```

Linux

```
./ConfigEngine.sh modify-virtual-portal -DPortalAdminPwd=password
-DWasPassword=password
```

Windows

```
ConfigEngine.bat modify-virtual-portal -DPortalAdminPwd=password
-DWasPassword=password
```

Related concepts:

"Language support" on page 1419
To reach as many users as possible, WebSphere Portal Express supports different languages for different locations. For instance, a large, international corporation might address users in different countries or regions through multilingual Web sites. In this context the portal can concurrently serve portal views to large numbers of users, each in the user's preferred language.

Related reference:

"Command reference for the Portal Scripting Interface" on page 1126
The IBM WebSphere Portal Express Scripting Interface component provides a scripting interface for the administration functions.

Related information:

Technotes for virtual portals

Task: delete-virtual-portal:

Portal ConfigEngine task that deletes a virtual portal.

Parameters

Use this task to delete a virtual portal by using its object ID. To determine the correct object ID of the virtual portal, use the task `list-all-virtual-portals`.

`VirtualPortalObjectId`
The object ID of the virtual portal. This input is mandatory for identification of the virtual portal that you want to delete.

Syntax

You pass the parameters in the parameter list for each configuration task as appropriate. You do this by either of the following methods:

- Specifying the parameter and value that is preceded by `-D` on the command line.
- Defining them in the file `wkplc.properties`.

Note: The property file must be encoded in the ISO 8859-1 character encoding format.

IBM i

```
ConfigEngine.sh delete-virtual-portal -DPortalAdminPwd=password
-DWasPassword=password
-DVirtualPortalObjectId=objectID_of_virtual_portal_to_delete
```

Linux

```
./ConfigEngine.sh delete-virtual-portal -DPortalAdminPwd=password
-DWasPassword=password
-DVirtualPortalObjectId=objectID_of_virtual_portal_to_delete
```

Windows

```
ConfigEngine.bat delete-virtual-portal -DPortalAdminPwd=password
-DWasPassword=password
-DVirtualPortalObjectId=objectID_of_virtual_portal_to_delete
```

Related information:

Technotes for virtual portals

Using a single configuration task to administer multiple virtual portals:

You can administer multiple virtual portals by running a single configuration command. The following configuration tasks support working with multiple virtual portals: `create-virtual-portal`, `delete-virtual-portal`, and `modify-virtual-portal`. Use the `-DvirtualPortalList` parameter with task to create, delete, or modify multiple virtual portals at the same time.

About this task

Specify the virtual portal property sets in the file `wkplc.properties` that is used with the configuration tasks. When you run the configuration task, you pass the list of virtual portals by using the parameter `-DvirtualPortalList`.

Here is an example for creating two virtual portals:

Procedure

1. Specify the two virtual portal property sets in the file `wkplc.properties` as follows:

```
vp1.VirtualPortalTitle=vp1
vp1.VirtualPortalRealm=
vp1.VirtualPortalHostName=
vp1.VirtualPortalContext=vp1
vp1.VirtualPortalNlsFile=

vp2.VirtualPortalTitle=vp2
vp2.VirtualPortalRealm=
vp2.VirtualPortalHostName=
vp2.VirtualPortalContext=vp2
vp2.VirtualPortalNlsFile=
```

Specify the realms and host names only if your portal has multiple realms.

2. To create both virtual portals by using a single configuration command, use the parameter `-DvirtualPortalList` to specify the two portals `vp1` and `vp2`:

   ```
   ConfigEngine create-virtual-portal -DvirtualPortalList=vp1,vp2
   ```

   - For IBM i: `ConfigEngine.sh create-virtual-portal -DvirtualPortalList=vp1,vp2`
   - For Linux: `ConfigEngine.sh create-virtual-portal -DvirtualPortalList=vp1,vp2`
   - For Windows: `ConfigEngine.bat create-virtual-portal -DvirtualPortalList=vp1,vp2`

Related information:

Technotes for virtual portals

Using the XML configuration interface to work with virtual portals:

You can export and import the contents of individual virtual portals by using the XML configuration interface. For example, you can use the XML configuration interface to fill a newly created virtual portal with content. As each virtual portal has its own globally unique portal ID, you can determine all resources that are associated with that virtual portal clearly and individually.

About this task

Address the virtual portal by the URL context or the host name that you specified when you created the virtual portal. Specify the unique virtual portal URL or host name with your XML request by either of the following options:

Note: In the following examples, the commands are shown on three lines, but you must enter them as one line.

- If you created the virtual portal by specifying a URL context:

  ```
  xmlaccess -user user -password password -url
  my_host:port_number/wps/config/URL_Context_of _the_Virtual_Portal
  -in XML_file -out result.xml
  ```

  For the variable *URL_Context_of _the_Virtual_Portal* use the URL context that you specified when you created the virtual portal. For more information, see *Creating a virtual portal* and *Task create-virtual-portal*.

- If you created the virtual portal by specifying a host name:

  ```
  xmlaccess -user user -password password -url
  host_name:port_number/wps/config -in XML_file -out result.xml
  ```

For the variable *host_name* use the host name that you specified when you created the virtual portal. For more information, see *Creating a virtual portal* and *Task create-virtual-portal*.

Note: When you use the XML configuration interface to work with virtual portals, be aware of the following rules:

1. You cannot export or import a complete virtual portal by using the XML configuration interface. You can export or import only the contents of a virtual portal. If you want to transfer a virtual portal from a source server to a target server, proceed as follows:
   a. Create the virtual portal on the target server.
   b. Export the contents of the virtual portal from the source server. Specify the URL context or host name of the source virtual portal as described previously.
   c. Import the XML result script from the previous step to the target server. Specify the URL context or host name that you used to create the target virtual portal to which you want to import the content.

   For more information about how to export and import portal configurations by using the XML configuration interface, see *Working with the XML configuration interface*.
2. You can export and import the contents of a single individual virtual portal at a time by using the XML configuration interface. You cannot export or import multiple virtual portals at the same time or an entire portal installation with multiple virtual portals. You must specify a separate XML request for each virtual portal. You can also export content from one virtual portal and import it into a different virtual portal.
3. The access rights for the XML configuration interface are limited to the master administrator of the portal installation as a whole. Subadministrators for the virtual portals cannot use the XML configuration interface to export or import the virtual portal that they administer.
4. Apply special care when you configure unscoped resources by using the XML configuration interface. Unscoped resources are shared between all virtual portals across the entire portal installation. A change of unscoped resources by the XML configuration interface affects all other virtual portals. For example, this feature applies to the following tasks and types of XML processing:
   - Updating URL mappings by using the XML configuration interface: A URL mapping of a URL context in one virtual portal can be unintentionally updated by XML import into another virtual portal to point to a resource in that second virtual portal. Therefore, if you export the content of one virtual portal and import it into a different virtual portal, make sure that you do not include the URL mappings of virtual portal URL contexts in the XML script. Otherwise, you might make the virtual portal unusable in the following two circumstances:
     – If the source virtual portal and the target virtual portal are on the same portal server, the URL mappings of the source virtual portal are updated to point to resources in the target virtual portal into which you imported the content. You can no longer use such a URL context to access the resource in the source virtual portal.
     – If the source virtual portal and the target virtual portal are not on the same portal server, but there is another virtual portal on the target portal server that has the same URL context as the source virtual portal. The URL mappings of this virtual portal are updated to point to resources in

the target virtual portal into which you imported the content. And you can no longer use such a URL context to access the resource in this virtual portal.

Note: This step is critical for the URL mapping of a URL context that is created for a virtual portal during its creation. Updating this initial URL mapping of a virtual portal URL context makes that virtual portal unusable.

- Deploying portlet applications into a virtual portal by using the XML configuration interface: If you deploy a portlet, that portlet is available to all virtual portals in the portal installation, unless you restrict this deployment by using Portal Access Control. If that portlet was already deployed in other virtual portals, errors can occur during the execution of the XML request.

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related reference:

"Task: create-virtual-portal" on page 1407
Portal ConfigEngine task that creates a new virtual portal.

Related information:

Technotes for virtual portals

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

"Creating a virtual portal" on page 1391
As a master administrator of the portal installation, you can create virtual portals by using the Virtual Portal Manager portlet.

"Configuring the sub administrators for virtual portals" on page 1394
You can administer the sub administrators of a virtual portal as required by using the Portal Access Control of your initial portal installation.

Hints and tips for working with virtual portals

View information about configuring virtual portals, hints and tips, and known limitations in WebSphere Portal Express Version 8.5:

- Do not grant the subadministrators of virtual portals the access permissions to run any installation-related tasks, such as installation of portlets or themes. The isolation between the different virtual portals as provided by WebSphere Portal Express Version 8.5 is limited to some extent. All virtual portals share a common Java virtual machine (JVM). Therefore, it is important to restrict the administration privileges of the virtual portal subadministrators and prevent them from installing their own code artifacts, such as themes or portlets. Unstable or malicious code that is introduced on one virtual portal can destabilize the entire portal installation and all other virtual portals. A flexible way to introduce virtual portal-specific portlets without impacting any other virtual portals is to use Web Services for Remote Portlets (WSRP). For more information about WSRP, go to *Using WSRP services*.
- Not all resources can be scoped to individual virtual portals. For example, all themes and skins are available to all virtual portals without restrictions. Credential vault, portlet services, and portal services are also common for an entire portal installation. They cannot be scoped to an individual virtual portal.

- The settings that are defined in the portal property files apply for the entire portal installation. You cannot specify separate settings for individual virtual portals.
- If you want to use the single signon feature that is provided by WebSphere Application Server, you must use the same common domain suffix for all virtual portals.
- Portal search, personalization, and templates, are not aware of virtual portals.
- There are no virtual portal-specific enhancements to the published portal commands and application programming interfaces.
- A URL mapping that is defined for a resource in a particular virtual portal must use the same URL context as the human readable URL context for that virtual portal itself. Example: In a virtual portal that uses the human readable URL mapping `wps/portal/vp_1`, all URL mappings for portal resources must start with `wps/portal/vp_1`, for example `wps/portal/vp_1/url_1` and `wps/portal/vp_1/url_2`. Within this virtual portal, a URL mapping such as `wps/portal/url_1` is not valid, as the portion `vp_1` of the URL Context is missing.
- The administration portlet Virtual Portal Manager cannot delete all resources that are associated with a virtual portal. For example, it does not delete extra URL mappings that administrators created for the virtual portal. You can use the XML configuration interface to delete the URL mappings.
- All virtual portals on a portal installation share a common logging and tracing.
- When you reinitialize a virtual portal by using the Virtual Portal Manager portlet, this applies the XML script for the default virtual portal content (or your custom script) again and re-creates the default content of the virtual portal. Resources that you removed from the default content are re-created. Resources that you added to the default content remain in the virtual portal.
- You must run the `wp-create-realm` task to create realms for your virtual portal. See the topics about *Realm support* and about adding realm support file for your operating system.
- The Portal Access Control administration in the Resource Permissions portlet shows users from different realms who have role mappings on shared resources by their object IDs. Therefore, apply special care and consideration when you are deleting such portal resources: Do not delete resources on which users from other realms have role mappings, if they are required in other virtual portals. This applies to members of roles on portal resources that cannot be scoped and are therefore shared between the virtual portals. Role members who belong to the realm of your local virtual portal are displayed as usual, but role members who belong to different realms are displayed in a different manner:
  - Role members for shared resources who belong to the virtual portal that you are currently working are listed by their actual names under which they were created.
  - Role members for shared resources that do not belong to the realm of the current portal are listed by their portal object IDs. For example, a role member from a different realm might be represented as `8_0_B`.

  Find the list of role members. Click the **Administration menu** icon. Then, click **Access** > **Resource Permissions**. .From the list of Resource Types, select a Resource Type by clicking it. On the Resource Permissions page, click on the **Assign Access** icon. The members are listed in the Roles column.
- You cannot create custom URLs in one virtual portal that address portal resource in another virtual portal. The reason is that both object IDs and unique names relate to resources of the local virtual portal.

Related concepts:

"WSRP services" on page 1433
IBM WebSphere Portal Express supports the Web Services for Remote Portlets (WSRP) standard. By using this standard, portals can provide portlets, applications, and content as WSRP services. Other portals can then integrate the WSRP services as remote portlets for their users.

"Realm support" on page 134
A realm is a collection of users or groups from one or more branches of your repository tree. Those branches can be part of a single repository, for example an LDAP user registry, or it can be a combination of multiple user registries. A realm is then mapped to a Virtual Portal to allow the realm's user population to log in to the Virtual Portal. This functionality allows you to define areas within WebSphere Portal Express that only a limited set of users can access.

Related information:

Technotes for virtual portals

Known limitations for virtual portals

The following sections describe known limitations of virtual portals.

"Scope of virtual portals"
IBM has tested an installation with 300 virtual portals and a total of more than 200,000 pages successfully. More detailed results of these tests are listed here.

"Using a new host name for an existing virtual portal" on page 1419
If you want to use a new host name for an existing virtual portal, you must delete the virtual portal. Then, re-create it with the new host name.

"Change of theme for virtual portal might not take effect" on page 1419
If you change the theme for a virtual portal by editing the virtual portal in the Virtual Portals portlet, this change affects only the root page of the virtual portal.

Related information:

Technotes for virtual portals

Scope of virtual portals:

IBM has tested an installation with 300 virtual portals and a total of more than 200,000 pages successfully. More detailed results of these tests are listed here.

- A IBM WebSphere Application Server 64-bit environment is suited best for a large number of virtual portals.
- For good results, you need to tune the portal caches and JVM.
- The limiting factor is not the absolute number of virtual portals, but the overall number of pages and URL mappings.
- In a limit and boundary test, the processor of the portal database machine was found to be the bottleneck.
- A large number of virtual portals can have an overall performance impact on administrative activities.
- Setting up and administering a large number of virtual portals by using the portal administration user interface can be time consuming.

Related information:

Technotes for virtual portals

Using a new host name for an existing virtual portal:

If you want to use a new host name for an existing virtual portal, you must delete the virtual portal. Then, re-create it with the new host name.

1. Export the contents of the virtual portal by using the XML configuration interface.
2. Delete the virtual portal.
3. Clean up references to the deleted virtual portal by using the `Task.xml` script of the XML configuration interface.
4. Create a new empty virtual portal by using the configuration task `create-virtual-portal`. Use the context of the deleted virtual portal, and specify the new host name as required.
5. Import the contents to the new virtual portal by using the XML configuration interface.

Related information:

Technotes for virtual portals

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Change of theme for virtual portal might not take effect:

If you change the theme for a virtual portal by editing the virtual portal in the Virtual Portals portlet, this change affects only the root page of the virtual portal.

If pages in your virtual portal have explicitly assigned themes, updating the theme for the virtual portal by using the Virtual portals portlet does not alter these pages. To change the theme for such pages of a virtual portal, edit the pages by using the user interface, or use the XML configuration interface (XML Access).

Related information:

Technotes for virtual portals

Language support

To reach as many users as possible, WebSphere Portal Express supports different languages for different locations. For instance, a large, international corporation might address users in different countries or regions through multilingual Web sites. In this context the portal can concurrently serve portal views to large numbers of users, each in the user's preferred language.

If necessary, the portal can support portlets that are shown in different languages. If portlets do not support a requested language, the portal tries to match the user's language preference as well as possible. For example, if a page for a Japanese user shows several portlets in Japanese and a portlet is added that only supports English, then the page shows the new portlet in English but still shows the other portlets in Japanese.

Note: Some back-level versions of browsers and browsers that are not commonly used might have difficulty representing specific languages, depending on the

defined character set. In these cases, in order for the language to be rendered correctly, it might be necessary to define the same preferred language for the browser and the portal it accesses.

WebSphere Portal Express has already been translated into a number of languages. They are shown in the following list, together with the ISO 639 language codes. These codes are used for the languages in WebSphere Portal Express:

WebSphere Portal Express language codes:

- `ar` = Arabic
- `ca` = Catalan
- `cs` = Czech
- `da` = Danish
- `nl` = Dutch
- `en` = English
- `fi` = Finnish
- `fr` = French
- `de` = German
- `el` = Greek
- `iw` = Hebrew
- `hr` = Croatian
- `hu` = Hungarian
- `it` = Italian
- `ja` = Japanese
- `kk` = Kazakh
- `ko` = Korean
- `no` = Norwegian
- `pl` = Polish
- `pt` = Portuguese
- `pt_BR` = Brazilian Portuguese
- `ro` = Romanian
- `ru` = Russian
- `sk` = Slovak
- `sl` = Slovenian
- `es` = Spanish
- `zh` = Simplified Chinese
- `zh_TW` = Traditional Chinese
- `sv` = Swedish
- `th` = Thai
- `tr` = Turkish
- `uk` = Ukrainian

WebSphere Portal Express uses the *ISO 639 Codes for the Representation of Names of Languages* to represent localized resources. The names for directories containing language-dependent resources follow the ISO 639 naming convention (see "Directories for languages" on page 3621).

This section has the following topics:

"Supporting a new language"
To support a new language to IBM WebSphere Portal Express you add resource bundles and, where applicable, JSPs for the new language.

"Changing the character set for a language" on page 1427
The character set is stored in the database. This is the character set used for the response to the user. You can change the character set for a language.

"Dynamically changing the language during the user session" on page 1428
Allow users to change the language while they are logged in to the portal.

"Selecting and changing the language" on page 1429
You can control the multiple language-specific settings within the portal.

"How to control the behavior of the language fallback filter" on page 1431
You can manage the language fallback behavior of IBM WebSphere Portal Express by a built-in servlet filter. This way, you control the way by which the portal determines the language for rendering portlets.

Related reference:

"Directory structure" on page 3618
The topic shows the naming conventions that are used to denote the location of files on the servers and the types of resources you can find in those directories.

Supporting a new language

To support a new language to IBM WebSphere Portal Express you add resource bundles and, where applicable, JSPs for the new language.

About this task

Some JSPs use resource bundles; others, such as help JSPs, are translated directly. Then you update the list of available languages by adding the new language to the portal. You do this by using the XML configuration interface. Use the example XML script `CreateLanguage.xml` to add the new language for the portal. You can also use it to remove an existing language from the portal. The new language is then listed in the language selection menu boxes that are available in administration portlets, or for example, in **Edit My Profile**, **Preferred language**.

Note: The new language will be available only to portlets that you add to your portal, if these portlets support the newly added language and if you make the required language files available. None of the WebSphere Portal Express user interface or messages will be translated to the new language.

"Adding resource bundles for a new language" on page 1422
To allow your portal users to work in an extra language, you add resource bundles for that language. Resource bundles are used to store text that is displayed in JSPs or text that is used in Java code.

CF06 "Adding a new language to render localized content" on page 1425
You can add new languages to the portlet to render localized content in different languages and reach a larger audience.

"Adding JSPs for a new language" on page 1425
For the new language in your portal, you also need to add JSPs. Some JSPs that contain mostly text, such as help JSPs, are translated directly which means that the text is contained in the JSP and not in a resource bundle. For JSPs that do not use resource bundles, you need to copy and translate an existing JSP and store it in the appropriate location.

"Globalization for People Finder fields" on page 1426
All fields of the People Finder portlet must have associated language strings that are defined in each language property file,

PeopleFinderUI_*language_code*.properties. The field names are the names of the corresponding attributes in Member Manager.

"XML samples for creating or removing language definitions" on page 1427
You can modify these XML samples and use them to create or remove language definitions from your portal.

Adding resource bundles for a new language

To allow your portal users to work in an extra language, you add resource bundles for that language. Resource bundles are used to store text that is displayed in JSPs or text that is used in Java code.

About this task

In WebSphere Portal Express, resource bundles are in the JAR files wp.ui.jar and wp.theme.customizer.ext.jar in the nls directory inside the file.

The JAR file wp.ui.jar is in the following directory:

- IBM i: *PortalServer_root*/ui/wp.ui/shared/app
- Linux: *PortalServer_root*/ui/wp.ui/shared/app
- Windows: *PortalServer_root*\ui\wp.ui\shared\app

The JAR file wp.theme.customizer.ext.jar is in the following directory:

- IBM i: *PortalServer_root*/theme/wp.theme.customizer.ext/shared/app
- Linux: *PortalServer_root*/theme/wp.theme.customizer.ext/shared/app
- Windows: *PortalServer_root*\theme\wp.theme.customizer.ext\shared\app

If you want to add new resource bundles for extra languages, place them in the following directory:

- IBM i: *wp_profile_root*/PortalServer/config/nls
- Linux: *wp_profile_root*/PortalServer/config/nls
- Windows: *wp_profile_root*\PortalServer\config\nls

The naming convention for resource bundles is [bundle]_[language]_[country]_[variant].properties. The ISO standard ISO-639 is used for the language codes of most languages. For Hebrew, the old language code iw is used. The ISO standard ISO-3166 is used for the country/region codes. WebSphere Portal Express supports the use of [variant], although resource bundles supplied with the portal do not use it.

Note: If your portal configuration includes Lotus Collaborative Services, add a CSRes_language.properties file for each additional language to the following directory:

- IBM i: *wp_profile_root*/PortalServer/config/nls
- Linux: *wp_profile_root*/PortalServer/config/nls
- Windows: *wp_profile_root*\PortalServer\config\nls

WebSphere Portal Express uses properties files that are called by the Java class **java.util.ResourceBundle** to store text that is rendered in JSPs. The Java mechanism searches for the resource bundles in the following order:

1. [bundle]_[language]_[country]_[variant].properties
2. [bundle]_[language]_[country].properties
3. [bundle]_[language].properties

4. [bundle].properties

In WebSphere Portal Express, the default bundles [bundle].properties are in English.

All languages that are defined for WebSphere Portal Express need to have resource bundles that are defined as well for the themes to function correctly. After you install a new language, add and process the resource bundles located within wp.ui.jar and wp.theme.customizer.ext.jar by using the procedure that is outlined in the following.

Procedure

1. Copy all existing resource bundles into *wp_profile_root*/PortalServer/config/nls directory.
2. Name the resource bundles according to the naming convention for resource bundles with locale code for the languages installed.
3. Translate the resource bundle files.
4. Convert them into Unicode with the Native-to-ASCII converter native2ascii that comes as part of JDK. For more detail about native2ascii, go to the Java documentation *native2ascii - Native-to-ASCII Converter*.
5. Restart your WebSphere Portal Express so that it recognizes the new resource bundles.

"Resource bundles to support a Portal based custom theme"
You can add supported locale to the system. You must provide resource bundles for the new language to the Portal based custom theme to enable them.

Related information:

native2ascii - Native-to-ASCII Converter

Resource bundles to support a Portal based custom theme:

You can add supported locale to the system. You must provide resource bundles for the new language to the Portal based custom theme to enable them.

About this task

Vietnamese is used as an example language in the following steps.

Procedure

1. Run the following XML to create the "Vietnamese" language. Click the **Administration menu** icon. Then, click **Portal Settings** > **Import XML**.

```
<?xml version="1.0" encoding="UTF-8"?>
<request xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 type="update"
 xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd">
 <portal action="locate">
  <language action="update" bidi="false" domain="rel" locale="vi_vn">
   <localedata locale="en_us">
    <title>Vietnamese</title>
   </localedata>
  </language>
 </portal>
</request>
```

2. Copy the contents of the following files from your Portal server into a new file called language_vi_vn.js:

- *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/modules/pagebuilder/js/nls/pb_ui_layer_en_us.js
- *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/modules/portalclient/js/nls/rest_utils_en_us.js
- If you use Active Site Analytics *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/modules/asa/js/nls/asa_layer_en_us.js
- If you use Tagging and Rating *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/modules/portalclient/js/nls/tagging_rating_en_us.js

3. In the language_*vi_vn*.js file, replace all instances of:

   ```
   en_us")
   ```

   with:

   ```
   vi_vn")
   ```

   and all instances of:

   ```
   .en_us=
   ```

   with:

   ```
   .vi_vn=
   ```

4. Place the language_*vi_vn*.js file into the js folder of your custom theme. For example, dav:fs-type1/themes/*custom_theme*/js.
5. Create a theme module for the language_*vi_vn*.js file by creating a languages.json file with the following contents:

   ```
   {
    "modules":[{
     "id":"custom_languages",
     "prereqs":[{
      "id":"dojo"
     }],
     "contributions":[{
      "type":"head",
      "sub-contributions":[{
       "type":"js",
       "uris":[{
        "value":"/js/language_vi_vn.js",
        "lang":"vi_vn"
       }]
      }]
     }]
    }]
   }
   ```

   For more information about theme modules, see Registering theme modules.
6. Place languages.json into the contributions folder of your custom theme. For example, dav:fs-type1/themes/*custom_theme*/contributions.
7. Add custom_languages as the first module in the section that includes Dojo for each profile in your custom theme. For example, in the Portal theme, the custom_languages module would be added to the **moduleIDs** section of the Lightweight profile and the **deferredModuleIDs** section of the Deferred profile. For more information, see *Adding or removing a ready-to-use module to a theme*.
8. Restart your portal to pick up the new language and module contribution.

Related tasks:

"Defining theme modules" on page 2547
You can define theme modules in XML or JSON.

"Adding or removing a module from a profile" on page 2656
To add or remove a module, update the profile that is used to render a page for the theme.

Adding a new language to render localized content

You can add new languages to the portlet to render localized content in different languages and reach a larger audience.

About this task

The Web Content Viewer portlet supports the same set of default languages as IBM WebSphere Portal Express To render localized web content in a new language, you need to extend the supported languages of the Web Content Viewer portlet.

Procedure

1. Click the **Administration menu** icon. Then, click **Portlet Management** > **Portlets**.
2. Search for **Web Content Viewer**.
3. Click **Configure portlet** icon.
4. In the **New Preference** field, enter `ibm.portal.wcm.locales.extension`.
5. In the **New value** field, enter a list of new locales that are separated by commas. For example, to add Australian, British, and American English, enter `en_AU,en_GB,en_US`.
6. Click **Add**.
7. Click **OK**.

Related concepts:

"Portal administration portlets" on page 1047
Administration portlets are supplied with IBM WebSphere Portal Express. Use them to perform administration tasks and actions on portal resources, give other users limited access rights on selected resources, and deploy custom portlets, themes, or skins.

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

"Localized rendering" on page 2459
Localized rendering is provided by either automatically matching the user locale with the content locale, or by providing a navigation option in the site to allow the user to choose the locale for themselves. These two strategies might also be used together in some cases.

Adding JSPs for a new language

For the new language in your portal, you also need to add JSPs. Some JSPs that contain mostly text, such as help JSPs, are translated directly which means that the text is contained in the JSP and not in a resource bundle. For JSPs that do not use resource bundles, you need to copy and translate an existing JSP and store it in the appropriate location.

About this task

The location of JSPs can be, for example, jsp/[mime-type]/[language]/[country]/[variant]/files.jsp. For instance, existing help JSPs are already translated in WebSphere Portal Express and placed in the relevant [language]and [country] subdirectories. When deciding where to store new JSPs, you need to consider how the portal locates a JSP for rendering its content.

The following is an example of the order in which directories are searched, where path1 is a user-defined path, ie5 is the markup version (here: Internet Explorer 5), and the locale is en_US:

1. /html/path1/ie5/en_US/mytemplate.jsp
2. /html/path1/ie5/en/mytemplate.jsp
3. /html/path1/ie5/mytemplate.jsp
4. /html/path1/en_US/mytemplate.jsp
5. /html/path1/en/mytemplate.jsp
6. /html/path1/mytemplate.jsp
7. /html/en_US/mytemplate.jsp
8. /html/en/mytemplate.jsp
9. /html/mytemplate.jsp
10. /mytemplate.jsp

This search order means that if the user language is not supported, the portal will choose the file in the locale independent location, which in the example is the English file.

Globalization for People Finder fields

All fields of the People Finder portlet must have associated language strings that are defined in each language property file, PeopleFinderUI_*language_code*.properties. The field names are the names of the corresponding attributes in Member Manager.

For each People Finder field, the following three strings are required:

- One string for the field name, which is the attribute name
- One string for the field long name
- One string for the field description as it appears in the tasks of People Finder configuration mode

Example:

```
displayName = Name
displayName_Long = Preferred Name
displayName_Desc = Display name
```

The resource key for each field refers to the appropriate language property file, /nls/PeopleFinderUI_*language_code*.properties. Language translations of People Finder fields must be stored in the appropriate language property file.

Language properties files are in the following location on the WebSphere Portal Express server:

PortalServer_root/people/people.impl/peoplefinder/portlet/*nnnnnnn*.ear/lwp_peoplefinder_war.ear/lwp.peoplefinder.jsr168.war/WEB-INF/lib

The variable *nnnnnnn* represents a random number that is generated by the installation process when the `lwppeoplefinder.jar` file is installed.

XML samples for creating or removing language definitions

You can modify these XML samples and use them to create or remove language definitions from your portal.

The following XML sample shows how you use the XML configuration interface to create Japanese language in your portal:

```
<request xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" type="update"
         xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd">
     <portal action="locate">
          <language action="update" domain="rel" locale="ja">
          <localedata locale="ja">
           <title>Japanese</title>
          </localedata>
          </language>
     </portal>
</request>
```

The following XML sample shows how you use the XML configuration interface to remove Japanese language in your portal:

```
<request xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" type="update"
         xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd">
    <portal action="locate">
         <language action="delete" domain="rel" locale="ja">
         <localedata locale="ja">
           <title>Japanese</title>
         </localedata>
         </language>
    </portal>
</request>
```

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Changing the character set for a language

The character set is stored in the database. This is the character set used for the response to the user. You can change the character set for a language.

To change the character set for a language, use the **Supported Markups** portlet. Proceed as follows:

1. Click the **Administration menu** icon..
2. Click **Portal settings** and then click **Supported markups**.
3. Select the markup for which you want to change the character set.
4. Click **Edit selected markup**.
5. Click **Set locale-specific settings**.
6. Select the language for which you want to make the change.
7. Click **Edit setting for selected language**.
8. Change the character set for the selected language in the selected markup.
9. Click **OK** to save your changes, or click **Cancel** to return without saving.

10. Again, on the panel with the list of languages and on the panel for editing the markup, click **OK** to save your changes, or click **Cancel** to return without saving.

For details about how to perform these tasks refer to the Supported Markups portlet help.

Note: For a portlet to be rendered correctly, the language of the portlet must be supported by the character set of the portal.

To help the user's browser to render content correctly, the used character set is written to the HTTP header of the response stream. The default encoding is UTF-8. If it set to another encoding, you can force the default encoding by setting the JVM parameter as follows `default.client.encoding=UTF-8` .

Dynamically changing the language during the user session

Allow users to change the language while they are logged in to the portal.

About this task

If you want your users to be able to change the language during the session, use the following command provided by IBM WebSphere Portal Express:

```
<portal-navigation:url command="ChangeLanguage">
   <portal-navigation:urlParam name="locale" value="language"/>
</portal-navigation:url>
```

where `language` is the two character code for the required language, such as `en`, `de`, or `fr`. For a list of the available languages and their two character codes read *Language support*.

For users to be able to dynamically change the language for the session, add a link to the portal theme with the following text and link reference:

- The text displayed with the link specifies the language to which the user can change.
- The link reference calls the command described previously with the locale parameter corresponding to the specified language.

Users can then click this link to change to the language specified by the locale parameter with the command. If you want to make more than one language available to users, you create a separate link for each language.

Example: To create links for English and German, add the lines shown in the following example to the banner area of your theme:

```
<!-- add these lines -->
<a href="<portal-navigation:url command="ChangeLanguage"><portal-navigation:urlParam name="locale"
     value="en"/></portal-navigation:url>">English</a>
<a href="<portal-navigation:url command="ChangeLanguage"><portal-navigation:urlParam name="locale"
     value="de"/></portal-navigation:url>">Deutsch</a>

<%-- logout button --%>
```

The banner area can be defined in different files, depending on the different themes. Themes in recent portal versions commonly define the banner area within the `Default.jsp`, whereas older themes can include a separate JSP, such as `banner_toolbar.jsp`. For more information about locating the files for your themes refer to the topic about the Location of theme resources.

Notes:

- The changed setting applies only for the duration of the current session. When the user logs out and back in again, the portal applies the default language as determined by the steps described under Selecting and changing the language.
- The previous examples use the `portal:` prefix to designate JSP tags from the portal tag library in `portal.tld`. Your custom JSPs might use a different tag prefix. Refer to Tags used by the portal JSPs for more information.
- Important for every operating system: Touch the *Default.jsp* file after editing any JSP files and before any restart. This updates the timestamp on the file to the current time and will signal a recompile of *Default.jsp* to incorporate the edit changes from other JSP files. Enter:

  ```
  touch Default.jsp
  ```

  An alternative is to edit (open and save) *Default.jsp*, which has the same effect as the touch command. After updating theme JSPs, you must restart WebSphere Portal Express unless JSP reloading is enabled.

Related concepts:

"Understanding the Portal Version 8.5 modularized theme" on page 2521
Modern websites and browsers enable incredible new capabilities that can greatly enhance your user's web experiences. However, these capabilities are not without cost in terms of large page sizes and more processing in the browser when each page is rendered. These capabilities are worth it when you need them, but removing them for an entire site or including them only on pages that take advantage of these capabilities provides for more flexibility.

Related information:

"Tags used by the portal JSPs" on page 2790
Learn about the most commonly used tags in the portal JSPs. Use these tags to modify the appearance and layout of the portal page.

Selecting and changing the language

You can control the multiple language-specific settings within the portal.

Changing the default portal language

After the installation, you can change the default language of the portal. To change the language, use the **Global Settings** portlet. Click the **Administration menu** icon. Then, click **Portal Settings** > **Global Settings**. From the drop-down list, select the required default language for the portal. This option can be, for example, `en` for English. For a list of languages that are supported by the portal, refer to the topic about Language support.

Note: The Global Settings portlet does not work in portal cluster configurations. For portal clusters, set the portal default language in the portal Localizer service by using the WebSphere Integrated Solutions Console. For details about how to set the language refer to the topic about Setting service configuration properties.

Portlets

A portlet can support one or more locales. All portlets must have their own default language that is specified in the deployment descriptor. Otherwise, the portlet cannot be installed.

Changing titles for pages

You can edit titles for pages in the portlet for Managing Pages by using the configure option for the locale-specific settings. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**. Click the **Edit Page Properties** icon, expand the Advanced options, and select the option for setting page titles and descriptions. You can enter a different title for each available language.

Language selection by the user

The user can select the preferred language for rendering portal content during the enrollment process. The user can select from a list of available languages. If required, the user can later change the selected language in the self-care portlet by selecting **Edit My Profile**. The selection list that is shown to the user for choosing a language shows all available portal languages.

Note: The user's language selection does not become effective until after the user logs in.

Language determined by the portal

The portal determines the language for rendering the portal content by a search process along the following sequence at login time:

1. The language that is encoded in the URL by the value for the `locale` parameter takes highest priority. The portal does not encode a locale into the URL by default. However, you can add code to the JSPs to support dynamic language selection. For more information, read *Dynamically changing the language during the user session*.
2. The language that is encoded in the navigational state by the locale tag: `<locale> . . . </locale>`.
3. The language that is stored in the `com.ibm.wps.state.preprocessors.locale.CookieSupportedLanguagePreProcessor` in the portal State Manager Service. For more information, read *State Manager Service*.
4. If the user logged in, the portal displays in the preferred language that is selected and stored in the user repository by the user.
5. If no preferred user language can be found, the portal looks for the language that is defined in the user's browser. If the portal supports that language, it displays the content in that language. If the browser has more than one language that is defined, the portal uses the first language in the list to display the content.
6. If no browser language can be found, for example if the browser used does not send a language or if the portal does not support the language that is set in the browser, the portal resorts to its own default language.
7. If the user has a portlet that does not support the language that was determined by the previous steps, that portlet is shown in its own default language.

This sequence describes the language selection process that is applied *for each user* at logon time. For pages viewed by anonymous users only, the last three steps for determining the language apply. This step applies, for example, before login and after logoff.

The language that is determined by this selection process is applied to the complete portal. If the portal or one component does not find the appropriate resources for the language as selected by this sequence, it tries to find the resources in a similar language. For example, if the determined language was US English (`en_US`), the next closest option is English (`en`).

This search sequence applies to all portal components individually including portlets. For example, if a portlet does not support the language that is selected by the portal, it is shown in default language of the portlet. This way, the portal can show individual portlets in different languages.

If a page does not support any of the languages that are determined by the steps that are given in the preceding list, then the navigation displays the object ID of the page rather than the page title. Such an object ID can be, for example, `7_0_5T`.

Preserving the language of the browser session

A users's choice of language is lost when the navigational state is cleared. For example, the language information is lost if users use bookmarks to friendly URLs for navigation or if the navigational state is cleared intentionally. You can preserve the language choice of the user's browser session. To do so, use the `com.ibm.wps.state.preprocessors.locale.CookieSupportedLanguagePreProcessor` preprocessor in the portal State Manager Service. It stores the language information to a cookie.

You can also determine the maximum lifetime of the cookie that holds the language information. To do so, use the `com.ibm.wps.state.preprocessors.locale.CookieSupportedLanguagePreProcessor.cookie.maxage` property in the portal State Manager Service.

For more detailed information, read *State Manager Service*

Related tasks:

"Dynamically changing the language during the user session" on page 1428
Allow users to change the language while they are logged in to the portal.

Related reference:

"State Manager Service" on page 347
The portal State Manager Service is the access point for managing the navigational state of the portal. The navigational state represents the current view of portal resources as displayed to a user.

How to control the behavior of the language fallback filter

You can manage the language fallback behavior of IBM WebSphere Portal Express by a built-in servlet filter. This way, you control the way by which the portal determines the language for rendering portlets.

By default, IBM WebSphere Portal Express Version 8.5 recognizes whether a general language code is secondary or farther down in the browser priority list. It ignores all regional variations that do not directly match a language that is defined by the portal. For example, if the browser locale list specifies `"ja-JP", "de", "ja"`, the portal ignores the entry `"ja-JP"` and falls back to the second entry of `"de"`. It recognizes that `"ja-JP"` does not have a match among the portal defined locales and that the more generic version `"ja"` of this locale is further down in the list after `"de"`. This behavior conforms to the HTTP specification.

The built-in language fallback servlet filter supports a mode that extends the language fallback behavior. If you enable this extended language fallback mode, the portal runs a fallback for all entries and removes duplicates of the fallback locales that are later in the list. In the example browser locale list of "ja-JP", "de", "ja" as given before, the portal recognizes that "ja-JP" does not match any of the portal defined locales. Therefore, it runs a fallback to the portal defined locale "ja". It uses "ja" as the locale of choice and ignores the third entry "ja" from the list. The resulting locale list that the portal uses is "ja", "de". This fallback mechanism bypasses strict adherence to the HTTP specification of accepting and showing languages and uses the fallback option instead. It also makes your WebSphere Portal Express Version 8.5 behave the same way as earlier portal versions.

You set the extended mode in the custom properties of the WP Configuration Service Resource environment provider by adding the property engine.enableExtendedLanguageFallback=true. Proceed by the following steps:

1. Open the WebSphere Integrated Solutions Console.
2. Select the Resource Environment Provider WP ConfigService.
3. In the Custom Properties section, add the property engine.enableExtendedLanguageFallback and set it to the value true.
4. Save your changes.
5. Restart your WebSphere Portal Express for your changes to take effect.

Example: Assume that the browser sends the Accept-Language header "DE_de, en". This combination means that the user prefers German as spoken in Germany as first priority, then English, but not German in general. In a portal scenario, these preferences would result in serving resources in "en", even though "de" would be supported (but is not acceptable according to the header). The default language fallback mode adds all fallback locales to the header, preserving the relative order of the original locales in the header. The result would be "DE_de, en, de", with the fallback to "de" showing after "en" to match the user preference.

In contrast, the extended language fallback mode gives fallbacks precedence over the original order of locales in the header. For "DE_de, en", the filter would generate "DE_de, de, en", giving all fallbacks for "DE_de" the same relative priority as "DE_de".

You can write your own filters and apply your own locale fallback logic. WebSphere Portal Express uses the value of the Accept-Language header for further processing. To replace the Language fallback servlet filter, proceed as follows:

1. Edit the file *wp_profile_root*/config/cells/node-name/applications/wps.ear/deployments/wps/wps.war/WEB-INF/web.xml.
2. Locate the following section:

```
<filter>
    <filter-name>Locale Filter</filter-name>
    <filter-class>com.ibm.wps.engine.ExtendedLocaleFilter</filter-class>
</filter>
```

3. Replace the filter class com.ibm.wps.engine.ExtendedLocaleFilter by the class name of your custom language filter.
4. Save the file.
5. Restart your WebSphere Portal Express for your changes to take effect.

Notes:

- If you remove a language from the portal, both filters rely on the locales that were defined last. For example, if you remove Japanese (`"ja-JP"`, `"ja"`) from the locales that are defined in the portal, then the browser locale list specifies `"ja-JP,de,ja"`. In this case, the portal recognizes that `"ja-JP"` and `"ja"` do not have a match among the portal defined locales. Therefore, it ignores these entries `"ja-JP"` and `"ja"`. Instead, it falls back to the second entry named `"de"`.
- IBM Web Content Manager does not use portlets and therefore cannot apply the filter that the portal uses for portlets. Therefore, if your portal installation includes Web Content Manager, you must set the custom filter in the Web Content Manager `web.xml` under the directory location *wp_profile_root*`/config/cells/`*node-name*`/applications/wcm.ear/deployments/wcm/ilwwcm.war/WEB-INF/`. Replace the extended filter as shown previously and restart the portal.

  If you use servlet delivery instead of portlet delivery, then you must edit `web.xml` under the directory location *wp_profile_root*`/config/cells/`*node-name*`/applications/PA_WCM_Authoring_UI.ear/deployments/PA_WCM_Authoring_UI/ilwwcm-authoring.war/WEB-INF/`.

Related reference:

"XML samples for creating or removing language definitions" on page 1427
You can modify these XML samples and use them to create or remove language definitions from your portal.

WSRP services

IBM WebSphere Portal Express supports the Web Services for Remote Portlets (WSRP) standard. By using this standard, portals can provide portlets, applications, and content as WSRP services. Other portals can then integrate the WSRP services as remote portlets for their users.

The WSRP standard and specification is provided by OASIS. It defines a web service communication interface for interactive presentation-oriented web services. This standard simplifies the integration of remote portlets, applications, and content into portals. Producers and Consumers use this interface for providing and consuming portlets. Users can work with WSRP in the following ways:

- Producers can provide portlets as presentation-oriented WSRP services and make them available to Consumers who want to use these services.
- Consumers can select from a rich choice of available remote portlets and integrate them into their portal.
- Portal site visitors can then access the integrated remote portlets. They can work and interact with them in the same way as they do with local portlets. The integrated remote portlets appear and operate to portal site visitors the same way as local portlets.

The WSRP implementation of WebSphere Portal Express Version 8.5 is built on the JAX-WS based web service stack of IBM WebSphere Application Server. The WSRP implementation in earlier versions of WebSphere Portal Express was based on the web service stack that is based on the JAX-RPC standard.

The information that is given in this WSRP section assumes that you are already familiar with WebSphere Portal Express and with the WSRP specification. If you want more detailed information about the OASIS WSRP specification, refer to the OASIS WSRP Standard website at http://oasis-open.org/committees/wsrp.

Before you start working with WSRP in your portal, read the relevant topics about WSRP carefully:

- In all cases, read the topics about planning, security considerations, and hints and tips for using WSRP with your portal.
- Depending on how you want to use WSRP, read the topics about Producers or Consumers.

Currently, there are two versions of the WSRP standard, WSRP 1.0 and WSRP 2.0. WebSphere Portal Express supports both versions of the WSRP standard.

The following topics provide information about WSRP Producers and Consumers and about how they communicate with each other.

"What is new in WSRP"
WSRP in IBM WebSphere Portal Express Version 8.5 is now based on the JAX-WS standard for Java based web services. It takes advantage of the improvements of the current JAX-WS based web services stack that is part of IBM WebSphere Application Server. The WSRP services are implemented as JAX-WS compliant service providers and service clients. To configure web service security and quality of service, you can manage them in the WebSphere Integrated Solutions Console by using policy sets.

"Learning about WSRP" on page 1435
Web Services for Remote Portlets (WSRP) allows easy integration of remote portlets, applications, and content into portal.

"Planning for WSRP" on page 1437
Before you work with WSRP, plan your configuration based on the information in the following topics.

"Using your portal as a WSRP Producer" on page 1444
Learn about the tasks that you perform when you use your portal to provide WSRP services as a WSRP Producer portal.

"Using your portal as a WSRP Consumer" on page 1457
Learn about the tasks that you perform when you use your portal as a WSRP Consumer portal to consume remote portlets. You can consume portlets from a IBM WebSphere Portal Express or from a different WSRP Producer such as the IBM WSRP Version 2.0 Producer for IBM WebSphere Application Server.

"Using handlers for WSRP web services" on page 1494
You can extend your WSRP Producer or WSRP Consumer portal by handlers that comply with JAX-WS.

"Reference for using WSRP with the portal" on page 1498
Reference information about using WSRP with the portal includes WSRP markup caching and Known limitations.

Related information:

OASIS WSRP Standard

What is new in WSRP

WSRP in IBM WebSphere Portal Express Version 8.5 is now based on the JAX-WS standard for Java based web services. It takes advantage of the improvements of the current JAX-WS based web services stack that is part of IBM WebSphere Application Server. The WSRP services are implemented as JAX-WS compliant service providers and service clients. To configure web service security and quality of service, you can manage them in the WebSphere Integrated Solutions Console by using policy sets.

WebSphere Portal Express now offers an extra security mechanism for WSRP. Besides, web services security, WSRP also supports HTTP-cookie-based Single Sign

On (SSO). This security mechanism does not require web service configuration and allows the WSRP Producer to process both authenticated and unauthenticated requests. The WSRP Producer works with WSRP Consumer portals that are built either on JAX-WS or on other web service stacks, such as JAX-RPC. In particular, WSRP in WebSphere Portal Express Version 8.5 works with WSRP counterparts in earlier versions of WebSphere Portal Express. The WSRP implementation in earlier versions of WebSphere Portal Express was based on the JAX-RPC API and the JAX-RPC web service stack.

New in CF 05

The WSRP Consumer markup caching feature now offers better performance. You can now use WSRP markup caching without enabling the portlet container fragment caching. With a new configuration parameter, you can enable and disable markup caching specifically for selected remote portlets or for all remote portlets of a Consumer portal.

The WSRP Consumer provides multiple new configuration parameters for defining two-phase rendering behavior, WSRP response timeouts, and a limit for the size of file uploads.

New in CF 06

You can now configure the WSRP Consumer to invalidate the remote session when a user explicitly logs out of the Consumer portal.

Learning about WSRP

Web Services for Remote Portlets (WSRP) allows easy integration of remote portlets, applications, and content into portal.

Related information

For information about the WSRP standard refer to the OASIS WSRP Standard website: http://oasis-open.org/committees/wsrp.

For more information about how WSRP works with the portal, read the following topics.

"WSRP Producer"
WSRP Producers provide portlets for Consumers who integrate them into their portal for their users.

"WSRP Consumer" on page 1436
WSRP Consumers consume the portlets that Producers provide.

"How Producer and Consumer portals communicate" on page 1436
The WSRP standard defines the interfaces and the protocol for communication between the Producer portal and the Consumer portal.

"Abbreviations" on page 1437
These topics about WSRP and the portal use some abbreviations.

Related information:

 OASIS WSRP Standard

WSRP Producer

WSRP Producers provide portlets for Consumers who integrate them into their portal for their users.

A WSRP Producer is a portal that provides WSRP services. These WSRP services make it possible for Consumers to access and call portlets on the Producer portal. Consumer portals can then consume portlets from the Producer portal as remote portlets by calling these WSRP services. The Producer works as the "server" of the WSRP communication.

A WSRP Producer provides one or more portlets through WSRP services for invocation by Consumer applications from remote sites. The Producer portal receives the requests from the Consumer portal to the WSRP service. The Producer portal generates the markup as required and sends it to the Consumer portal, where it is displayed for the user who started the request.

To allow communication between the Producer portal and the Consumer portal, the Producer provides a set of web service interfaces to the Consumers. These interfaces are defined by the WSRP standard. The Producer can make some or all of these interfaces available to the Consumers as appropriate. The Producer provides a description for the WSRP service interfaces and WSRP services in a service description document. The format of the service description document is defined in the Web Services Description Language (WSDL) standard. This WSDL service description also provides general technical information, for example, the endpoint addresses of the WSRP services.

WSRP Consumer

WSRP Consumers consume the portlets that Producers provide.

A WSRP Consumer is a portal that calls WSRP services on Producer portals and consumes portlets from Producer portals that provide them as remote portlets.

The Consumer part of the WSRP implementation in the portal enables the portal to consume portlets by calling WSRP services from remote Producer portals. The Consumer works as the "client" of the WSRP communication.

A WSRP Consumer integrates selected portlets into the Consumer portal as remote portlets. The Consumer portal requests and receives the markup from the remote WSRP service at the Producer portal and presents it to its users. This way, the Consumer portal can consume one or more remote portlets through the WSRP services.

How Producer and Consumer portals communicate

The WSRP standard defines the interfaces and the protocol for communication between the Producer portal and the Consumer portal.

To set up communication with a Producer portal, the Consumer portal requires the following information from the Producer portal:

- The WSDL (Web Services Description Language) service description document, which provides the following information:
  - Descriptions of the WSRP interfaces and WSRP services that the Producer provides.
  - Technical information, such as the service endpoint addresses.
- Information about the quality of service and security configuration of the WSRP services on the Producer portal.

When the Consumer has this information, the administrator of the Consumer portal can configure the Consumer portal accordingly. After this configuration, the Consumer can consume remote portlets from the Producer portal.

Depending on the overall WSRP setup, the Consumer might in turn provide information to the Producer. For example, if the Producer portal has security configured, the Consumer can send user IDs of Consumer portal users to the Producer. The Producer can then give these users access to the provided portlets. The Consumer portal users can then work with the portlets that the Consumer portal consumes from the Producer.

Related concepts:

"Exchanging the required information between Producer and Consumer portals" on page 1439
The WSRP standard defines the interfaces and the protocol for communication between the Producer portal and the Consumer portal. The Producer portals provide WSRP services that can be called to invoke provided portlets. The Consumer portals request WSRP services to call the remote portlets.

"Information that the Producer exchanges with the Consumer" on page 1452
As a WSRP Producer, you must provide information to Consumers of your WSRP services so that they can prepare for consuming them as remote portlets. Depending on the configuration, you might also need information from the Consumer.

"Information that the Consumer exchanges with the Producer" on page 1458
Before a Consumer can consume portlets from a Producer, the Consumer needs specific information about the Producer. Depending on your configuration, you might also need to provide information about your Consumer portal to the Producer.

Abbreviations

These topics about WSRP and the portal use some abbreviations.

J2EE Java 2 Platform, Enterprise Edition

JSSE Java Secure Socket Extension

OASIS
Organization for the Advancement of Structured Information Standards

SOAP Simple Object Access Protocol

UDDI Universal Description, Discovery, and Integration

WSRP Web Services for Remote Portlets

WSDL
Web Services Description Language

Planning for WSRP

Before you work with WSRP, plan your configuration based on the information in the following topics.

You can consume portlets from a IBM WebSphere Portal Express or from a different WSRP Producer such as the IBM WSRP Version 2.0 Producer for IBM WebSphere Application Server.

Prerequisites for WSRP in the portal

If you want to use WSRP with your portal, you must have the appropriate level of IBM WebSphere Application Server installed. Refer to WebSphere Portal detailed system requirements to determine the WebSphere Application Server version that is required by your version of WebSphere Portal Express.

"Supported portlet APIs"
Learn about how the WSRP implementation in WebSphere Portal Express Version 8.5 supports different portlet APIs.

"Exchanging the required information between Producer and Consumer portals" on page 1439
The WSRP standard defines the interfaces and the protocol for communication between the Producer portal and the Consumer portal. The Producer portals provide WSRP services that can be called to invoke provided portlets. The Consumer portals request WSRP services to call the remote portlets.

"Security for WSRP services" on page 1441
IBM WebSphere Portal Express supports two security mechanisms for WSRP.

"How you work with WSRP in your portal" on page 1443
To work with WSRP in your portal, you perform different administrative tasks. Some of these tasks depend on whether you use your portal as a Producer or Consumer portal.

Related information:

WebSphere Portal detailed system requirements

Supported portlet APIs

Learn about how the WSRP implementation in WebSphere Portal Express Version 8.5 supports different portlet APIs.

Currently, there are two versions of the WSRP standard, WSRP 1.0 and WSRP 2.0. WebSphere Portal Express supports both versions of the WSRP standard.

You can use the WSRP implementation in WebSphere Portal Express Version 8.5 to do the following with your portal:

For Producer portals:

The WebSphere Portal Express WSRP Producer can provide a portlet through WSRP 1.0, or 2.0, or both, depending on the API version of the portlet:

- The Producer can provide JSR 168 portlets through both WSRP V 1.0 and V 2.0.
- The Producer can provide JSR 286 portlets only through WSRP V 2.0.

The following table shows which provided portlets on a WSRP Producer are available in which WSRP version:

Table 176. . Which API type portlets the Producer can provide through the WSRP versions.

API to which the portlet complies	Provided by WSRP V 1.0	Provided by WSRP V 2.0
JSR 168 standard API	X	X
JSR 286 standard API	JSR 286 portlets cannot be provided by WSRP V 1.0.	X

For Consumer portals:

The WebSphere Portal Express WSRP Consumer supports both WSRP V 1.0 and WSRP V 2.0 Producer portals. It can therefore consume all two types of portlets: JSR 168 and JSR 286.

Note: Currently, the WSRP Producer implementation in the portal does not support the Registration interface of the WSRP specification. This interface is optional. However, the Consumer portal can handle Producers that support WSRP registration interfaces.

Exchanging the required information between Producer and Consumer portals

The WSRP standard defines the interfaces and the protocol for communication between the Producer portal and the Consumer portal. The Producer portals provide WSRP services that can be called to invoke provided portlets. The Consumer portals request WSRP services to call the remote portlets.

Currently, there are two versions of the WSRP standard, WSRP 1.0 and WSRP 2.0. WebSphere Portal Express supports both versions of the WSRP standard.

WSRP builds upon existing web service technology and web service standards. The WSRP implementation of WebSphere Portal Express Version 8.5 is built on the JAX-WS based web service stack of IBM WebSphere Application Server. The WSRP implementation in earlier versions of WebSphere Portal Express was based on the web service stack that is based on the JAX-RPC standard. WSRP in WebSphere Portal Express Version 8.5 can interoperate with WSRP counterparts that are built on other web service stacks, such as JAX-RPC. In particular, WSRP in WebSphere Portal Express Version 8.5 can communicate and interoperate with WSRP counterparts in WebSphere Portal Express Version 8.0 and earlier versions.

WSRP defines a set of four web service interfaces. Two of these web service interfaces are mandatory and two are optional. The following table shows how the WSRP Producer and the WSRP Consumer support the interfaces:

Table 177. The four WSRP web service interfaces and how the WSRP Producer and Consumer support them

Web service interface	Supported by WebSphere Portal Express WSRP Producer	Supported by WebSphere Portal Express WSRP Consumer
Service Description	Yes	Yes
Markup	Yes	Yes
Registration (optional)	No	Yes
Portlet Management (optional)	Yes	Yes

Currently, the WSRP Producer implementation of the portal does not support the Registration interface of the WSRP specification. However, the Consumer implementation of the portal can interoperate with Producers that support WSRP Registration interfaces.

To set up communication with a Producer portal, the Consumer portal requires the following information from the Producer portal:

- The WSDL (Web Services Description Language) service description document, which provides the following information:
  - Descriptions of the WSRP interfaces and WSRP services that the Producer provides.
  - Technical information, such as the service endpoint addresses.

- Information about the quality of service and security configuration of the WSRP services on the Producer portal.

When the Consumer has this information, the administrator of the Consumer portal can configure the Consumer portal accordingly. After this configuration, the Consumer can consume remote portlets from the Producer portal.

Depending on the overall WSRP setup, the Consumer might in turn provide information to the Producer. For example, if the Producer portal has security configured, the Consumer can send user IDs of Consumer portal users to the Producer. The Producer can then give these users access to the provided portlets. The Consumer portal users can then work with the portlets that the Consumer portal consumes from the Producer.

The information that follows here describes details of the WSRP Producer and the WSRP Consumer. These details are relevant for the administrators of their respective portals. For example, an administrator of the WSRP Producer must know which service providers the WSRP Producer supports. An administrator of the WSRP Consumer normally does not need to know about the implementation details of a WSRP Producer.

Service providers of the WSRP Producer

To comply with the JAX-WS standard, the WSRP Producer of WebSphere Portal Express provides a set of service providers that implement the WSRP web service interfaces. The following table lists the supported service providers:

Table 178. WSRP Producer service providers that provide the web service interfaces for the two WSRP versions

Web service interface	WSRP 1.0 service provider	WSRP 2.0 service provider	WSRP 2.0 service provider for portal-internal WSRP communication
Service Description	WSRPServiceDescriptionService	WSRPServiceDescriptionService_v2	WSRPServiceDescriptionService_v2_internal
Markup	WSRPBaseService	WSRPBaseService_v2	WSRPBaseService_v2_internal
Portlet Management	WSRPPortletManagementService	WSRPPortletManagementService_v2	WSRPPortletManagementService_v2_internal

You can administer and configure each of the WSRP 1.0 and 2.0 service providers separately. You administer and configure them by using the administration clients of WebSphere Application Server. For example, you can use the WebSphere Integrated Solutions Console.

Note: Do not change the configuration of the WSRP 2.0 service providers for portal-internal WSRP communication. These service providers are used internally by WebSphere Portal Express during client side aggregation. Remote consumer portals cannot access these service providers.

Service clients and references of the WSRP Consumer

The WSRP Consumer of WebSphere Portal Express provides a set of service clients and default service references. The set includes two service clients, one each to support WSRP 1.0 and WSRP 2.0 and the respective port types or web service interfaces. There is one default service reference per service client. You can configure and administer the service clients and service references by using the WebSphere Integrated Solutions Console. For example, you can use the WebSphere Integrated Solutions Console to configure WSRP service providers and WSRP service clients by attaching policy sets. You can configure each service client and service reference separately.

The following table lists the supported service clients and service references:

Table 179. WSRP Consumer service clients and service references for the two WSRP versions

	WSRP 1.0	**WSRP 2.0**
Service client	`WSRPService`	`WSRPService_v2`
Service reference	`service/wsrp/WSRPService`	`service/wsrp/WSRPService_v2`

The service clients support all WSRP service interfaces: Service Description, Markup, Portlet Management, and Registration.

Note: The configuration of the WSRP service clients is managed outside WebSphere Portal Express. The portal WSRP Consumer supports all service client configurations that are configured in WebSphere Application Server. This support includes message level security, transport level security, and other quality of service configuration. However, the service client configuration of the WSRP Consumer must be compatible with the web service configuration of the WSRP Producer. Example: If the service providers of the Producer portal are configured for WS-Security, the service references of the Consumer portal must also be configured for WS-Security. Otherwise, the WSRP communication fails.

Related concepts:

"Information that the Producer exchanges with the Consumer" on page 1452
As a WSRP Producer, you must provide information to Consumers of your WSRP services so that they can prepare for consuming them as remote portlets. Depending on the configuration, you might also need information from the Consumer.

"Information that the Consumer exchanges with the Producer" on page 1458
Before a Consumer can consume portlets from a Producer, the Consumer needs specific information about the Producer. Depending on your configuration, you might also need to provide information about your Consumer portal to the Producer.

Security for WSRP services

IBM WebSphere Portal Express supports two security mechanisms for WSRP.

You can configure security for WSRP in your WebSphere Portal Express, but you do not need to configure security.

If security is configured for both the WSRP Producer and the WSRP Consumer, the WSRP Consumer sends a security token to the WSRP Producer as part of the WSRP request message. The security token represents the identity of the current user of the Consumer portal. The WSRP Producer uses this security token to authenticate and identify the user. The WSRP Producer processes the WSRP request under this user identity and performs access control for the provided portlets. If the WSRP Producer cannot process the security token or cannot authenticate the user on the WSRP Consumer side, the WSRP Producer rejects the WSRP request.

WebSphere Portal Express supports the following security mechanisms:

HTTP-cookie-based single sign-on

You can configure the WSRP Consumer to forward LTPA V2 cookies to the WSRP Producer as part of the WSRP request messages. The WSRP Producer uses these cookies to authenticate and identify the user and establish the security context for processing the WSRP request. This security option has the following advantages:

- It does not require configuration of the WSRP web services.
- It makes it possible for the WSRP Producer to accept and process both unauthenticated and authenticated requests. The Producer processes unauthenticated requests that do not contain an LTPA V2 cookie without establishing an individual security context. This way, it can serve requests from anonymous users.

Web Services Security (WSS)
You can configure the WSRP Consumer and WSRP Producer for Web Services Security according to the WS-Security standard. With a WS Security configuration, the WSRP Consumer sends a header that complies with the WS-Security standard as part of the WSRP request messages. The header contains credentials that identify and authenticate the user. For this option, both the WSRP Consumer and the WSRP Producer must be configured for Web Services Security.

For a WSRP Producer, security configuration is optional. A WSRP Consumer must use the same security configuration as the WSRP Producer from which it consumes portlets. If the request message that is sent by the WSRP Consumer does not comply to the security configuration of the WSRP Producer, the WSRP Producer does not accept the message. If the request message from the WSRP Consumer contains the security token that the WSRP Producer expects, the Producer processes the request under the appropriate user identity. If the WSRP Producer is not configured for security, it processes WSRP requests under the anonymous user identity.

Considerations for configuring security:

For Producer portals:
For a Producer portal, security for WSRP services is **optional**. You can configure securityt if required, but you do not have to do so.

When you configure security, you must also configure Portal Access Control and assign access rights for the Consumer portal users on the Producer portal. Assign the access rights based on the security configuration information as follows:

- If you use security, assign access rights on the Producer portal to the actual Consumer portal users.
- If you do not use security, assign access rights to the anonymous user, or disable Portal Access Control for the WSRP Producer.

By default, Portal Access Control is enabled for the WSRP Producer. For details about how to disable and enable Portal Access Control for the WSRP Producer, read *Configuring Portal Access Control for a WSRP Producer portal*.

For Consumer portals:
- For a Consumer portal, you must define a security configuration that is compatible with the security configuration of the Producer portal from which you consume WSRP services. This configuration must include all the appropriate security aspects.
- On the Consumer portal, the consumed portlets behave like local portlets. Therefore, you can configure Portal Access Control for the remote portlets on the Consumer portal the in same way as for local portlets. If you use Web Services Security, do not make the affected remote portlets available to anonymous users on the Consumer portal.

Instead, configure Portal Access Control to make the affected remote portlets available to authenticated users only.

For portals that work as both a Producer and a Consumer portal:

- If you use your portal as both a Producer and a Consumer portal, the security configurations for both these roles are independent of each other.

For more detailed information about Portal Access Control, read the sections about *Configuring Portal Access Control for a WSRP Producer portal* and *Managing Access Control*.

Related tasks:

"Configuring Portal Access Control for a WSRP Producer portal" on page 1452
If you configure security for WSRP services, you must also configure Portal Access Control for the Producer.

Related information:

"Managing Access Control" on page 1525
Get familiar with concepts related to administering IBM WebSphere Portal Express access control. To administer access control, use the Resource Permissions portlet, the User and Group Permissions portlet, the Manage Users and Groups portlet, the XML configuration interface, or the Portal Scripting Interface.

 WebSphere Application Server product documentation V 8.5

How you work with WSRP in your portal

To work with WSRP in your portal, you perform different administrative tasks. Some of these tasks depend on whether you use your portal as a Producer or Consumer portal.

Producer tasks

When Producers work with WSRP, they perform the following tasks:

1. Planning for WSRP
2. Using your portal as a WSRP Producer:
   a. Preparing security for a WSRP Producer portal and configuring the WSRP service providers. Preparing security is optional, depending on your setup.
   b. Exchanging the required information with Consumers of your WSRP services. This exchange includes information about the WSRP interfaces and the configuration of the WSRP services.
   c. Working with WSRP:
      1) Providing the local portlets in your Producer portal as WSRP services so that Consumer portals can consume and call them as remote portlets.
      2) Withdrawing a portlet. If you withdraw a portlet, Consumers can no longer consume this portlet remotely.
      3) If you do not use message authentication, assign anonymous users the Privileged User role for the provided portlets.

      To perform these tasks, you use the **Manage Portlets** portlet. Alternatively, you can use the XML configuration interface.
   d. Customizing the WSRP configuration of your Producer portal.

Consumer tasks

When Consumers work with WSRP, they perform the following tasks:

1. Planning for WSRP
2. Using your portal as a WSRP Consumer:
   a. Exchanging the required information for the WSRP communication with the Producer. This exchange includes information about the WSRP interfaces and the configuration of the WSRP services.
   b. Configuring the WSRP service clients, especially the security aspects. Whether you must perform this task depends on the Producer portal security setup.
   c. Creating and configuring one or more Producer definitions for the Producer portals from whom you want to consume WSRP services. To work with Producer definitions, use the portal administration portlet **Web Service Configuration**. You can also use the XML configuration interface.
   d. Consuming WSRP services, that is integrating and using the WSRP services in your portal as remote portlets. To consume WSRP services, use the portal administration portlet **Manage Web Modules**. You can also use the XML configuration interface.

Using your portal as a WSRP Producer

Learn about the tasks that you perform when you use your portal to provide WSRP services as a WSRP Producer portal.

About this task

1. "How to access the Producer WSDL" on page 1445
   As a Producer you must provide the URL for the Producer WSDL service description document to the Consumer.
2. "Securing a WSRP Producer portal" on page 1447
   To secure provided portlets, you can configure the WSRP Producer for web service message security, for example, for message authentication. If you configure message authentication, you must also configure Portal Access Control.
3. "Information that the Producer exchanges with the Consumer" on page 1452
   As a WSRP Producer, you must provide information to Consumers of your WSRP services so that they can prepare for consuming them as remote portlets. Depending on the configuration, you might also need information from the Consumer.
4. "Providing WSRP services as a Producer" on page 1454
   After you prepared your portal as a Producer portal, you can provide your portlets through WSRP. Providing portlets makes them available to Consumers. They can integrate them into their Consumer portals and use them as remote portlets. You can also withdraw portlets from being provided through WSRP. Consumer portals can then no longer use them.
5. "Exporting customized WSRP portlet instances by using the XML configuration interface" on page 1456
   If consumed portlets are customized on the Consumers portal, then the Producer can export the customized instances of those portlets by using the XML configuration interface.
6. "Changing the WSRP Producer context root" on page 1457
   The context root for the WSRP Producer references the context root for the WSRP Producer facade servlet. This context root is the entry point for all WSRP

protocol traffic, and you can change the context root as required to support your environment. This customization is optional.

How to access the Producer WSDL

As a Producer you must provide the URL for the Producer WSDL service description document to the Consumer.

The basic WSDL document is available under the following URL:

`http://producer_portal_host:producer_port/WpsContextRoot/wsdl/wsrp_service.wsdl`

where *WpsContextRoot* is the portal context root that was set at installation time. You can find its value in the file `wkplc.properties`.

This URL refers to a WSDL file with the WSRP 2.0 services. When a Consumer portal administrator configures the Consumer portal to interact with the Producer, the Consumer administrator must specify the WSDL URL. The Consumer accesses this URL and reads the contents of the WSDL document, which provides the endpoint addresses of the WSRP 2.0 services. Consumers use these endpoint addresses to communicate with the Producer through the WSRP protocol.

The URL format of the WSRP service endpoint addresses is as follows:

protocol`://`*host_name*`:`*port_number*`/`*wsrp_context_root*`/`*wsrp_port_name*

wsrp_context_root references the context root for the WSRP Producer facade servlet. The facade servlet is provided with the `wps.ear` enterprise application in the WebSphere Portal Server WSRP Facade web module `wps_facade.war` and controls access to the WSRP web service engine.

You can change the context root for the WSRP Producer with the `modify-servlet-path` configuration task, as described in *Changing the portal URI*

In addition, you can configure and override various settings to manipulate the contents of the WSDL document. To do so, you specify the appropriate parameters in the URL for the WSDL service description document:

- The WSRP version: WSRP Version 1.0, WSRP Version 2.0, or both
- The host name for the endpoint addresses
- The transfer protocol: HTTP, HTTPS, or both
- The port number for HTTP URLs
- The port number for HTTP**S** URLs.

To generate the WSDL service description document, the Producer portal uses the following parameters:

1. Parameters from the URL of the request for the service description document
2. Settings in the portal Configuration Service
3. Settings that result from the request for the service description document.

As a result, the portal provides a fallback mechanism for selecting the parameters from the WSDL service description document, depending on which parameters you specify:

1. If the administrator of the Consumer portal adds parameters as URL parameters to the URL for requesting the service description document, then the Producer portal uses these parameters. For example, it can use the parameters for creating the endpoint addresses of the WSRP web services that are contained in the WSDL document.

2. If you do not define such a parameter as a URL parameter in the service description document request, then the Producer looks for entries in the portal Configuration Service.
3. If you do not specify the parameter in the Configuration Service, then the Producer uses the host name and port of the incoming request to generate the endpoint addresses of the WSRP web services that are contained in the WSDL document.

URL parameters

The administrator of the Consumer portal can request a modified WSDL service description document from the Producer. In this document, the Producer modifies the endpoint addresses of the WSRP web services according to URL parameters that the Consumer provided. These parameters that are listed and described in the following table. Here is an example of the supported URL syntax:

```
http://producer_portal_host:producer_port/wp_contextRoot/wsdl/wsrp_service.wsdl
       ?protocol=<protocolValue>&port=<httpPort>&securePort=<httpsPort>
       &version=<WSRPVersion>&hostname=<hostname>
```

The following table lists possible values for URL parameters:

Table 180. Values for URL parameters:

URL parameter	Possible values	Results in WSDL file
`protocol`	`http`	WSRP endpoint addresses with HTTP protocol
	`https`	WSRP endpoint addresses with HTTP**s** protocol
	mixed	WSRP endpoint addresses with HTTP and with HTTP**s** protocol
`version`	`v1`	WSRP services for WSRP Version 1.0
	`v2`	WSRP services for WSRP Version 2.0
`port`	Integer, for example `80`	Port number for HTTP endpoint addresses
`portSecure`	Integer, for example `443`	Port number for HTTP**s** endpoint addresses
`hostname`	Name of the host, for example `localhost`	The host name that is used for endpoint addresses

Note: The URL parameters do not modify the Producer settings and bindings. They manipulate only the content of the WSDL service description document, for example, for debugging and tracing purposes.

Settings in the portal Config Service

To control the default output for the WSDL document, you set the following parameters in the portal configuration service:

wsrp.hostname = localhost
: Use this property to specify a host name for the endpoint addresses in the WSRP service WSDL document of the Producer.

wsrp.port.http = 80
Use this property to specify the HTTP port that is used for the endpoint addresses in the WSRP service WSDL document of the Producer.

wsrp.port.https = 443
Use this property to specify the HTTP**S** port that is used for the endpoint addresses in the WSRP service WSDL document of the Producer.

Related tasks:

"Changing the portal URI after an installation" on page 368
You can change the default portal Uniform Resource Identifier (URI) any time after you install IBM WebSphere Portal Express. Some applications have a fixed context root that cannot be changed.

Securing a WSRP Producer portal

To secure provided portlets, you can configure the WSRP Producer for web service message security, for example, for message authentication. If you configure message authentication, you must also configure Portal Access Control.

"Configuring security on the Producer portal"
You can configure security for the WSRP Producer portal and the provided portlets. If you enable security, the WSRP Producer processes the WSRP requests from the WSRP Consumer under the user identity that is associated with the WSRP request that the Consumer sent. This user identity is represented by a security credential that is included in the WSRP request message. The security credential is provided by the WSRP Consumer. Normally, it represents the identity of the user who is logged in to the Consumer Portal.

"Configuring Portal Access Control for a WSRP Producer portal" on page 1452
If you configure security for WSRP services, you must also configure Portal Access Control for the Producer.

Configuring security on the Producer portal:

You can configure security for the WSRP Producer portal and the provided portlets. If you enable security, the WSRP Producer processes the WSRP requests from the WSRP Consumer under the user identity that is associated with the WSRP request that the Consumer sent. This user identity is represented by a security credential that is included in the WSRP request message. The security credential is provided by the WSRP Consumer. Normally, it represents the identity of the user who is logged in to the Consumer Portal.

About this task

Note: For the WSRP Producer, security for WSRP services is optional. You can configure it if required, but you do not have to provide security. If you provide security for your WSRP services, the WSRP Consumer must be configured to use the same security mechanism as the WSRP Producer from which the Consumer consumes portlets.
You can configure security for the WSRP Producer by using either of the following two authentication mechanisms:

HTTP-cookie-based single sign-on
This security option is newly available with WebSphere Portal Express Version 8.5. To authenticate and identify the user and establish the security context for processing the WSRP request, the WSRP Producer uses LTPA V2 HTTP cookies that the WSRP Consumer sends as part of the WSRP request messages. The WSRP Producer receives the cookie and establishes

the corresponding security context on the Producer side. This option requires configuration of the WSRP Consumer to forward HTTP cookies. It has the following advantages:

- It does not require configuration of the WSRP web services. It makes it possible for the WSRP Producer to accept and process both unauthenticated and authenticated requests.
- The Producer processes unauthenticated requests that do not contain an LTPA V2 cookie without establishing an individual security context.

Web Service Security

You can configure the WSRP web service providers for Web Service Security according to the WS-Security standard. The WSRP Consumer sends a header that complies with the WS-Security standard as part of the WSRP request messages. The header contains credentials that identify and authenticate the user. For example, you can configure the Consumer portal to include Lightweight Third-Party Authentication (LTPA) version 1 or version 2 tokens or Username tokens in the WS-Security header. For this option, both the WSRP Consumer and the WSRP Producer must be configured for Web Services Security.

The Web Service Security configuration is based on policy sets. IBM WebSphere Portal Express provides a set of default policy sets and provider policy set bindings that can be attached to the WSRP service providers. If you configure your WSRP Producer for WS-Security, the Producer accepts and processes only authenticated requests. It rejects unauthenticated requests that do not contain a WS-Security compliant header.

For both security setup options, the WSRP Producer and the WSRP Consumer must be configured for Single Sign-On (SSO). The requirements for SSO depend on the authentication method that is used. For example, if you use LTPA version 1 or version 2, the WSRP Consumer and the WSRP Producer must use the same user registry or use the same realm. In addition, the WSRP Producer and the WSRP Consumer must exchange shared keys that are used to sign the security credentials.

If you use the Web Services Security option, the WSRP Producer accepts only authenticated request messages and rejects request messages that do not contain a suitable security header. In contrast, if you use the HTTP-cookie-based single sign-on security option, the WSRP Producer accepts both authenticated and unauthenticated request messages. If the message does not contain a security credential, the WSRP Producer does not establish a security context for processing the request. By default, the WSRP Producer performs access control for provided portlets.

You can choose to not set up security for the WSRP Producer and Consumer portals. In this case, the WSRP Producer does not process the WSRP requests from the Consumer under a specific user identity. Instead, the Producer processes the WSRP requests anonymously. In this case, the Consumer must not be configured for Web Service Security.

"Securing the WSRP Producer by HTTP-cookie-based single sign-on" on page 1449

You can provide security for your WSRP Producer by using HTTP-cookie-based single sign-on (SSO). For using this security option, the WSRP Producer

requires no configuration. The WSRP Consumer must be configured to send or forward LTPA V2 single sign-on cookies as part of the WSRP request message to the WSRP Producer.

"Securing the WSRP Producer by WS-Security" on page 1450
You can configure Web Services Security according to the WS-Security standard for your WSRP Producer and the provided web services.

Related concepts:

"Exchanging the required information between Producer and Consumer portals" on page 1439
The WSRP standard defines the interfaces and the protocol for communication between the Producer portal and the Consumer portal. The Producer portals provide WSRP services that can be called to invoke provided portlets. The Consumer portals request WSRP services to call the remote portlets.

"Security for WSRP services" on page 1441
IBM WebSphere Portal Express supports two security mechanisms for WSRP.

Related information:

WebSphere Application Server product documentation V 8.5

Securing the WSRP Producer by HTTP-cookie-based single sign-on:

You can provide security for your WSRP Producer by using HTTP-cookie-based single sign-on (SSO). For using this security option, the WSRP Producer requires no configuration. The WSRP Consumer must be configured to send or forward LTPA V2 single sign-on cookies as part of the WSRP request message to the WSRP Producer.

About this task

The single sign-on cookie represents a security credential that can be understood both by the WSRP Consumer and the WSRP Producer. The WSRP Producer receives the cookie and establishes the corresponding security context for the user on the Producer side.

For using HTTP-cookie-based single sign-on, the WSRP Producer must not be configured for Web Services Security.

Prerequisites for using HTTP-cookie-based single sign-on:

For using HTTP-cookie-based single sign-on (SSO), single sign-on must be configured between the WSRP Consumer and the WSRP Producer. This configuration requires the following two prerequisites:

- The WSRP Consumer and the WSRP Producer must be configured to use a shared user registry.
- The LTPA keys must be exchanged between WSRP Consumer and WSRP Producer.

Required WSRP Consumer configuration:

For using HTTP-cookie-based single sign-on, the WSRP Consumer must be configured to forward single sign-on cookies to the WSRP Producer. For more information, read *Securing the WSRP Consumer by HTTP-cookie-based single sign-on*.

Related tasks:

"Securing the WSRP Consumer by HTTP-cookie-based single sign-on" on page 1463
You can configure your WSRP Consumer for using HTTP-cookie-based single sign-on. For this option, you must configure the WSRP Consumer to send or forward LTPA V2 single sign-on cookies as part of the WSRP request message to the WSRP Producer.

Securing the WSRP Producer by WS-Security:

You can configure Web Services Security according to the WS-Security standard for your WSRP Producer and the provided web services.

About this task

The WSRP Producer in IBM WebSphere Portal Express provides a set of JAX-WS compliant service providers. You can manage the configuration of the WSRP service providers in IBM WebSphere Application Server through the concept of policy sets. You might want to configure the service providers of the WSRP Producer for WS-Security-based authentication and caller identification. You can do so by attaching an appropriate policy set to the service provider, for example by using the WebSphere Integrated Solutions Console.

The WebSphere Application Server ensures message security and quality of service according to the configuration that you defined. The WSRP Producer provides a set of default policy sets and default provider policy set bindings. You can use them for configuring WSRP service providers. You do not have to create your own policy set and provider policy set binding.

The following table describes the provided WSRP application policy sets and the provided WSRP provider policy set bindings:

Table 181. Provided WSRP application policy sets and provided WSRP provider policy set bindings

	WSRP application policy sets	**WSRP provider policy set bindings**
LTPA based	**LTPA-based message authentication policy set** This policy set defines LTPA token-based message authentication. It does not define other security mechanisms such as message confidentiality, or other web service mechanisms such as WS-Addressing.	**LTPA-based message authentication provider binding** You must use this provider policy set binding with the LTPA-based message authentication policy set. It defines the corresponding provider binding, including caller identification.
Username based	**Username-based message authentication policy set** This policy set defines Username token-based message authentication. It does not define other security mechanisms such as message confidentiality, or other web service mechanisms such as WS-Addressing.	**Username-based message authentication provider binding** You must use this provider policy set binding with the Username-based message authentication policy set. It defines the corresponding provider binding, including caller identification.

The WSRP application policy sets and client policy set bindings are contained in

compressed format in the directory `PortalServer/doc/policy-sets-samples` of the portal installation. For instructions about how to import and attach policy sets and provider policy set bindings, read the WebSphere Application Server documentation.

To use the WSRP policy sets and provider policy set bindings for service configuration, use the procedure given later in this topic.

Note: You are not limited to using the default policy sets and provider policy set bindings. Instead, you can also create and use a policy set and provider policy set binding of your choice. The WSRP Producer supports all service configurations that WebSphere Application ServerWebSphere Application Server supports. Therefore, you can use all security tokens that WebSphere Application Server supports. Some token types might require a specific setup. For more detailed information about web service configuration, read the WebSphere Application Server product documentation. Note that it is necessary to define a compatible web service configuration on the WSRP Consumer portals.

Procedure

1. Import the WSRP policy sets and provider policy set bindings. To do so, use a WebSphere Application Server administrative client, such as the WebSphere Integrated Solutions Console:
   a. Open the **Application policy sets** panel.
   b. Select **Import (From Selected Location)**.
   c. Select the LTPA-based message authentication policy `set.zip` file or the username-based message authentication policy `set.zip` file that you want to import.
   d. Open the **General provider policy set bindings** panel.
   e. Select **Import (From Selected Location)**.
   f. Select the LTPA-based message authentication provider `binding.zip` file or username-based message authentication provider `binding.zip` file that you want to import.
2. Attach the policy set and provider policy set binding to a WSRP service provider. Proceed as follows:
   a. Open the **Service providers** panel.
   b. Open the service provider that you want to configure. Do not select one of the internal service providers, such as `WSRPBaseService_v2_internal`, `WSRPPortletManagementService_v2_internal`, or `WSRPServiceDescriptionService_v2_internal`.
   c. Select the service. The service is the first resource listed.
   d. Use the **Attach** option to select and attach the LTPA-based message authentication policy set or the username-based message authentication policy set.
   e. Select the service. The service is the first resource listed.
   f. Use the **Assign Binding** option to select and assign the LTPA-based message authentication provider binding file or username-based message authentication provider binding.
   g. Save your changes to the master configuration.
3. After you have completed this configuration, restart your portal.

Related information:

WebSphere Application Server product documentation V 8.5

Configuring Portal Access Control for a WSRP Producer portal:

If you configure security for WSRP services, you must also configure Portal Access Control for the Producer.

Before you begin

By default Portal Access Control is enabled for the Producer portal. You might want to disable Portal Access Control for using WSRP with WebSphere Portal. You can disable the WSRP security by setting the portal configuration parameter `wsrp.security.enabled` in the portal WP Configuration Service to `false`. You complete this step in the WebSphere Integrated Solutions Console. With this setting all portlets that your Producer portal provides can be accessed through the WSRP protocol without any authentication.

To enable Portal Access Control for the Producer portal again, set the portal configuration parameter `wsrp.security.enabled` in the portal WP Configuration Service to `true`.

Procedure

1. Activate the WSRP security by setting the property `wsrp.security.enabled` to `true` in the portal WP Configuration Service in the WebSphere Integrated Solutions Console.
2. Assign access rights based on the authentication information:
   - If you use security, assign access rights to the actual Consumer portal users. Get the user information from the Consumer.
   - If you do not use security, assign access rights to the anonymous user.

Information that the Producer exchanges with the Consumer

As a WSRP Producer, you must provide information to Consumers of your WSRP services so that they can prepare for consuming them as remote portlets. Depending on the configuration, you might also need information from the Consumer.

The exchange between Producers and Consumers includes the following information:

- "WSRP service description"
- "WSRP services configuration" on page 1453
- "Information about portal users" on page 1453
- "Registration information" on page 1453

WSRP service description

WSRP Consumers need information about how to bind to the WSRP services that the Producer provides. This information is described in the Web Service Description Language (WSDL) document of the Producer. The WSDL document provides general technical information about how the Consumer connects to the Producer and about the related infrastructure. The Consumer can use the information in the WSDL document to bind to the Producer and retrieve the Producer's service description for further details about the Producer.

The WSDL document provides information about various aspects and properties of the Producer:

- The WSRP services that the Producer provides

- The end-point addresses of the WSRP services.

As a Producer you can customize the content of the WSDL service document that you provide to WSRP Consumer portals.

WSRP services configuration

The Producer must provide information about the configuration of the WSRP web services to the Consumer. The Producer web service configuration is not described in the WSDL service description document. For WSRP communication to be successful, the WSRP producer web service configuration and the WSRP Consumer web service configuration must be compatible. The administrator of the WSRP Consumer portal must ensure that the WSRP Consumer portal uses a web services configuration that is compatible with the configuration of the Producer portal.

Example: The Producer configures web service security by using LTPA V2 tokens for WSRP services. If the Consumer sends messages that are secured by a different mechanism or not secured at all, the Producer cannot accept the messages.

Depending on the mechanism that is used for security, the Producer and Consumer portals must be set up specifically. For example, a security mechanism can require a user repository that is shared between the Consumer and Producer portals. Other security mechanisms can also require an exchange of certificates or public keys between the Consumer and Producer portals. For details about the prerequisites and implications of web service configuration and security, read the WebSphere Application Server product documentation.

Information about portal users

If the Producer portal has security for its WSRP services set up, the administrator of the Producer portal must assign access permissions to the users of the Consumer portal by using Portal Access Control. In this case, the Producer must obtain the required user information from the Consumer.

Registration information

If the Producer requires registration by the Consumer portal, the Producer must provide the required registration information to the Consumer.

"Producer checklist for exchanging information with a Consumer"
Use this list to check whether you provided and obtained all required information that you must exchange with the Consumer.

Producer checklist for exchanging information with a Consumer:

Use this list to check whether you provided and obtained all required information that you must exchange with the Consumer.

Information that a Producer provides to a Consumer

Mandatory information:

The WSDL service description document that contains the WSRP service description of the Producer.

The following information can be required.

Whether this information is required depends on the setup of the Producer portal:

The service configuration of the Producer portal.

If the Producer has set up a specific WSRP configuration, the Consumer needs this information. For example, a security mechanism can require a user repository that is shared between the Consumer and Producer portals. Other security mechanisms can also require an exchange of certificates or public keys between the Consumer and Producer portals.

Registration information.

The Consumer needs this information if the Producer requires registration.

Group IDs and handles of portlets.

If the Consumer administrator consumes a remote portlet from a Producer portal by using the XML configuration interface, the Consumer needs the group IDs and the portlet handles of the remote portlets at the Producer portal.

Information that a Producer obtains from a Consumer

The following information can be required.

Whether this information is required depends on the security setup of the Producer portal:

User information about users of the Consumer portal who uses the remote portlets.

If the Producer portal has security configured, the Producer needs this information to give the users of the Consumer portal access permission to the provided WSRP services.

Providing WSRP services as a Producer

After you prepared your portal as a Producer portal, you can provide your portlets through WSRP. Providing portlets makes them available to Consumers. They can integrate them into their Consumer portals and use them as remote portlets. You can also withdraw portlets from being provided through WSRP. Consumer portals can then no longer use them.

About this task

To provide or withdraw a portlet for WSRP services, you can use either of the following options:

- The portal administration portlet **Manage Portlets**.
- The portal XML configuration interface. For more information about the XML configuration interface (XMLAccess), read the topics about the XML configuration interface.

Notes:

1. A Producer provides portlets, not portlet instances. Therefore, only settings that are made in the Configure mode of the portlet on the Producer portal are available at the consumed remote portlet. Adding a remote portlet to a page on the Consumer side might create a new instance of the provided portlet on the Producer side. This portlet instance can be modified only on the Consumer portal. It is not available for use on the Producer portal.

2. Customization of the Producer portlets by Consumer portal users can be exported by using the XML configuration interface.

Proceed by selecting the appropriate topic from the following links:

"Using the Manage Portlets portlet to provide portlets through WSRP"
To provide portlets through WSRP, you use the **Manage Portlets** portlet. You can also use this portlet to withdraw the portlet from being available through WSRP.

"Using the XML configuration interface to provide or withdraw a portlet"
A WSRP Producer can provide or withdraw portlets by using the XML configuration interface.

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Using the Manage Portlets portlet to provide portlets through WSRP:

To provide portlets through WSRP, you use the **Manage Portlets** portlet. You can also use this portlet to withdraw the portlet from being available through WSRP.

About this task

Click the **Administration menu** icon. Then, click **Portlet Management** > **Portlets**. Then, click **Manage Portlets**.

For details about how to work with the portlet, refer to the portlet help.

Using the XML configuration interface to provide or withdraw a portlet:

A WSRP Producer can provide or withdraw portlets by using the XML configuration interface.

Before you begin

To provide or withdraw the portlet by using the XML configuration interface, specify the `provided` attribute to the `portlet` tag:

provided = "true | false"

Use this attribute with the `portlet` tag to specify providing or withdrawing a portlet:

true To provide the portlet as a WSRP service, you set the `provided` attribute to `true`. When you run the XML script, the portlet is provided through WSRP. The portlet can now be consumed as a remote portlet by Consumer portals.

false To withdraw the portlet, set the `provided` attribute to `false`. The portlet is withdrawn. It is no longer available for Consumer portals to consume.

Example

XML script examples:

The following two XML samples show you how to use the XML configuration interface to provide a portlet that complies with the standard portlet API. The examples show the **provided** attribute highlighted. If you want to withdraw a portlet by using the XML configuration interface, specify `false` instead of `true` for the **provided** tag.

Providing a standard API portlet:

The following XML sample shows you how to provide a portlet that complies with the standard portlet API:

```
<?xml version="1.0" encoding="UTF-8" ?>
<request type="update" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="PortalConfig_1.4.xsd">
      <!--
          Sample for providing a standard API compliant portlet as a WSRP producer.
          Be aware that this sample is provided as a sample only.
    It might or might not work, depending on the configuration of your portal.
      -->
   <portal action="locate">
      <!--
          uid must match the uid of the portlet application appended with .webmod
      -->
      <web-app action="locate" active="true"
         uid="stdTestsuite.war.webmod">
            <!--
                uid must match the optional portlet-app id attribute from the portlet.xml.
                If this is not set, the .war file name must be supplied here.
            -->
         <portlet-app action="update" uid="stdTestsuite.war">
            <!--
                Name must match the portlet-name tag in the portlet.xml file.
            -->
            <portlet action="update" name="TestPortlet1" provided="true" />
         </portlet-app>
      </web-app>
   </portal>
</request>
```

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Exporting customized WSRP portlet instances by using the XML configuration interface

If consumed portlets are customized on the Consumers portal, then the Producer can export the customized instances of those portlets by using the XML configuration interface.

Users of the Consumer portal can customize the consumed portlets, if the Producer gave them the required access rights. The customized portlet instances are created on the Producer portal. The Producer portal administrator can export the customized portlet instances by using the XML configuration interface for later import into another portal.

Here is an example XML script for exporting all customized WSRP portlet instances:

```
<request
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
    type="export">

    <!-- Sample for exporting the customized portlet instances of a WSRP Producer -->
    <portal action="locate">

        <wsrp-customized-portletinstance objectid="*" action="export"/>

    </portal>
</request>
```

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Changing the WSRP Producer context root

The context root for the WSRP Producer references the context root for the WSRP Producer facade servlet. This context root is the entry point for all WSRP protocol traffic, and you can change the context root as required to support your environment. This customization is optional.

About this task

The facade servlet is provided with the `wps.ear` enterprise application in the `WebSphere Portal Server WSRP Facade` web module `wps_facade.war` and controls access to the WSRP Web Service engine.

Procedure

To change the context root for the WSRP Producer, run the `modify-servlet-path` configuration task, as described in *Changing the portal URI*.

Related tasks:

"Changing the portal URI after an installation" on page 368
You can change the default portal Uniform Resource Identifier (URI) any time after you install IBM WebSphere Portal Express. Some applications have a fixed context root that cannot be changed.

Using your portal as a WSRP Consumer

Learn about the tasks that you perform when you use your portal as a WSRP Consumer portal to consume remote portlets. You can consume portlets from a IBM WebSphere Portal Express or from a different WSRP Producer such as the IBM WSRP Version 2.0 Producer for IBM WebSphere Application Server.

1. "Information that the Consumer exchanges with the Producer" on page 1458
   Before a Consumer can consume portlets from a Producer, the Consumer needs

specific information about the Producer. Depending on your configuration, you might also need to provide information about your Consumer portal to the Producer.

2. "Working with Producer definitions" on page 1473
   To make a WSRP Producer known to your Consumer portal, you create a Producer definition for that WSRP Producer in your Consumer portal. You can also configure the Producer definition.
3. "Consuming portlets in a Consumer portal" on page 1483
   After you create a Producer definition, you can proceed to consume the portlets that are provided by that Producer. This way, you integrate them into your Consumer portal as remote portlets.
4. "Customizing the WSRP configuration of your Consumer portal" on page 1486
   You can customize some aspects of you WSRP Consumer portal.

Related information:

 IBM WSRP 2.0 Producer for WebSphere Application Server documentation

Information that the Consumer exchanges with the Producer

Before a Consumer can consume portlets from a Producer, the Consumer needs specific information about the Producer. Depending on your configuration, you might also need to provide information about your Consumer portal to the Producer.

The information that Producers and Consumers exchange includes the following items:

- "WSRP service description"
- "WSRP interfaces" on page 1459. They are included in the WSDL service description document.
- "WSRP web service configuration and user information" on page 1459
- "Registration information" on page 1460
- "Group IDs and handles of portlets" on page 1460.

WSRP service description

WSRP Consumers need technical information about the WSRP services that the Producer provides. This information is described in the Web Service Description Language (WSDL) document of the Producer. The Consumer can use the information in the WSDL document to set up communication with the Producer.

The WSDL document provides information about various aspects and properties of the Producer:

- The WSRP services that the Producer provides
- The WSRP interfaces that the Producer supports
- The end-point addresses of the WSRP services.

If the WSRP Producer is an IBM WebSphere Portal Express, then Consumers of WSRP services can access the WSDL document of the Producer at the following URL:

`http://producer_portal_host:producer_port/wp_contextRoot/wsdl/wsrp_service.wsdl`

Notes:

1. The host and port and the *wp_contextRoot* directory must match the host and port of the Producer WebSphere Portal Express installation.

2. If the communication with the Producer is set up to use Secure Socket Layer communication, the Consumer must use **HTTPs** to address this URL:

```
https://producer_portal_host:producer_port/wp_contextRoot/wsdl/wsrp_service.wsdl
```

If the WSRP service Producer is *not* a WebSphere Portal Express, then the owner or administrator of the Producer portal must provide the information to the Consumer.

WSRP interfaces

The WSRP standard defines a set of four web service interfaces. The interfaces are listed here:

Service Description
This interface is mandatory. It provides the self-description of the Producer and a description of the available portlets.

Markup
This interface is mandatory. It is an interface for requesting and interacting with markup fragments.

Portlet Management
This interface is optional. It provides operations for managing the lifecycle of the hosted portlets and their persistent state.

Registration
This interface is optional. It is not supported by the current implementation of the WSRP Producer in WebSphere Portal Express. However, the WebSphere Portal Express Consumer can handle Producers that support WSRP registration interfaces.

The Producer can provide some or all of these interfaces to the Consumers as appropriate. The Producer describes these WSRP interfaces in the Web Services Description Language (WSDL) document as described under "WSRP service description" on page 1458. The WSDL document provides general technical information about the web services that the Producer provides.

WSRP web service configuration and user information

The Producer can use a specific security or web service configuration for the provided WSRP services. This configuration can include a specific web service security configuration. The configuration of the WSRP web services on the Consumer portal must be compatible with the web service configuration of the Producer portal. For example, the Producer might configure message authentication according to the WS-Security standard for the WSRP services. In this case, the WSRP Consumer web services must also be configured for message authentication. Depending on the actual web service configuration, the Producer and Consumer portals might have to use the same user registry.

If the Producer portal has security configured for its WSRP services, the administrator of the Producer portal must assign access permissions for the provided portlets to the users of the Consumer portal. to do so, the Producer uses Portal Access Control. In this case, the Producer must obtain the required user information from the Consumer.

Registration information

If the Producer requires registration by the Consumer portal, the Producer must provide the required registration information to the Consumer.

Group IDs and handles of portlets

A Consumer portal can consume a remote portlet from a Producer portal by using the XML configuration interface. In this case, the Consumer portal administrator must specify the `handle` and `groupid` of the remote portlet. Therefore, the Producer must provide this information to the Consumer.

The portlet handle and group ID are listed in the WSRP service description of the Producer. The portlet handle for each portlet that the Producer provides is listed in a `portletHandle` tag. The group ID for each portlet that the Producer provides is listed in the `groupID` tag of the service description.

"Consumer checklist for exchanging information with a Producer"
Use this list to check whether you have obtained and provided all required information that you must exchange with the Producer.

"Configuring security on the Consumer portal" on page 1461
You can configure security for the WSRP Consumer. If you enable security, the WSRP Consumer sends a security token as part of the WSRP request message to the WSRP producer. The security token represents the identity of the user who is logged in to the Consumer Portal. The WSRP Producer uses the security token to process the WSRP requests under the user identity that is represented by the security token.

Consumer checklist for exchanging information with a Producer:

Use this list to check whether you have obtained and provided all required information that you must exchange with the Producer.

Information that a Consumer obtains from the Producer

Mandatory information:

The WSDL document that contains the service description of the Producer.
The WSDL document must include a description of the **two mandatory WSRP interfaces: Service Description and Markup**. The WSDL document can contain a description of the two optional WSRP interfaces: Portlet Management and Registration.

Information that can be required
Depending on the setup of the Producer portal, the following information can be required:

The web service configuration of the WSRP services of the Producer portal. The Consumer needs the web service configuration if the Producer has set up a specific web service configuration for the WSRP services.

The user registry information.
The Consumer might need this information if the Producer has configured message authentication that is based on token forwarding.

Registration information.
The Consumer needs this information if the Producer requires registration.

Group IDs and handles of portlets.
The Consumer needs these IDs and handles if the Consumer administrator consumes a remote portlet from a Producer portal by using the XML configuration interface.

Information that a Consumer provides to the Producer

Information that can be required.
The information that can be required depends on the security setup of the Producer portal.

User information about users of the Consumer portal who will use the remote portlets.
If the Producer portal has message authentication configured, the Producer must grant the users of the Consumer portal access permission to the provided portlets.

Configuring security on the Consumer portal:

You can configure security for the WSRP Consumer. If you enable security, the WSRP Consumer sends a security token as part of the WSRP request message to the WSRP producer. The security token represents the identity of the user who is logged in to the Consumer Portal. The WSRP Producer uses the security token to process the WSRP requests under the user identity that is represented by the security token.

About this task

For a WSRP Producer, security for WSRP services is optional. If a WSRP Producer requires security, the WSRP Consumer must be configured to use the same security mechanism as the WSRP Producer. Otherwise, the Consumer cannot consume the portlets that the Producer provides.

Example: A Producer might configure message authentication Web Service Security for the WSRP services by using a particular security token type according to the WS-Security standard. In this case, the WSRP Consumer web services must also be configured for web service security, and they must use the same security token type message authentication. You can configure security for the WSRP Consumer by using either of the following two authentication mechanisms:

HTTP-cookie-based single sign-on
The WSRP Consumer forwards LTPA v2 HTTP cookies that it receives from the client to the Producer as part of the WSRP request messages. The WSRP Producer receives the cookie and establishes the corresponding security context on the Producer side. This option requires configuration of the WSRP Consumer to forward HTTP cookies. It has the following advantages:

- It does not require configuration of the WSRP web services. It makes it possible for the WSRP Producer to accept and process both unauthenticated and authenticated requests.
- The Producer processes unauthenticated requests that do not contain an LTPA V2 cookie without establishing an individual security context.

Web Services Security

You can configure the WSRP Consumer to use Web Service Security according to the WS-Security standard. The WSRP Consumer sends a header that complies with the WS-Security standard as part of the WSRP request messages. The header contains credentials that identify and authenticate the user. For example, you can configure the Consumer portal to include Lightweight Third-Party Authentication (LTPA) tokens or Username tokens in the WS-Security header. For this option, both the WSRP Consumer and the WSRP Producer must be configured for Web Services Security.

When you configure the WSRP Consumer for Web Service Security, you can choose the security token type for the WSRP ports of a Producer definition. If you configured the security token type, the WSRP Consumer portal creates a security token of the selected type when it sends a request to the respective WSRP port of the Producer.

By alternative, you can manage the configuration of the WSRP service clients in IBM WebSphere Application Server by using policy sets. This type of management includes the security-related aspects and the quality of service related aspects of the service configuration. You configure the service clients and service references of the WSRP Consumer by attaching an appropriate policy set to the service client. IBM WebSphere Portal Express WebSphere Portal provides a set of default policy sets and client policy set bindings. To configure them, you use the WebSphere Application Server administration functions.

For both setup options, the WSRP Producer and the WSRP Consumer must be configured for Single Sign-On (SSO). The requirements for SSO depend on the authentication method that is used. For example, if you use LTPA V2, the WSRP Consumer and the WSRP Producer must use the same user registry or use the same realm. In addition, the WSRP Producer and the WSRP Consumer must exchange shared keys that they use to sign the security credentials.

"Securing the WSRP Consumer by HTTP-cookie-based single sign-on" on page 1463
You can configure your WSRP Consumer for using HTTP-cookie-based single sign-on. For this option, you must configure the WSRP Consumer to send or forward LTPA V2 single sign-on cookies as part of the WSRP request message to the WSRP Producer.

"Configuring WSRP Producer ports for Web Service Security on the Consumer portal" on page 1464
You can configure each WSRP port of a particular Producer definition for web service security by using LTPA or username tokens.

"Configuring WSRP web service clients" on page 1466
You might want to set up a specific and complex service configuration. In this case, you can configure the WSRP service clients and service references of the WSRP Consumer by using the concept of policy sets. If you intend to configure web service security by using LTPA or username tokens, do not configure the WSRP service clients and service references. In this case, you do not need to read this topic and its subtopics, but follow the procedure described in Configuring WSRP Producer ports for web service security on the Consumer portal.

"Enabling Portal Access Control for a WSRP Consumer portal" on page 1473
You can configure Portal Access Control for the remote portlets that you consume on your Consumer portal.

Securing the WSRP Consumer by HTTP-cookie-based single sign-on:

You can configure your WSRP Consumer for using HTTP-cookie-based single sign-on. For this option, you must configure the WSRP Consumer to send or forward LTPA V2 single sign-on cookies as part of the WSRP request message to the WSRP Producer.

About this task

The single sign-on cookie represents a security credential that both by the WSRP Consumer and the WSRP Producer understand. The WSRP Producer receives the cookie and establishes the corresponding security context for the user on the Producer side. For using HTTP-cookie-based single sign-on, the WSRP Consumer must not use Web Services Security. In particular, if you plan to use this security option, you must not configure Web Services Security for the respective Producer definition.

Prerequisites for using HTTP-cookie-based single sign-on:

For using HTTP-cookie-based single sign-on (SSO), single sign-on must be configured between the WSRP Consumer and the WSRP Producer. This configuration requires the following two prerequisites:

- The WSRP Consumer and the WSRP Producer must be configured to use a shared user registry.
- The LTPA keys must be exchanged between WSRP Consumer and WSRP Producer.

Configuring the WSRP Consumer for HTTP-cookie-based single sign-on:

For cookie forwarding of the LTPA v2 cookie, follow the description given in *Customizing client cookie forwarding*. You need to create a cookie forwarding rule for the cookie named LtpaToken2. To include the Producer host and the hosts of all resources that are linked by the remote portlets, choose the `hostdomainname` parameter.

The following example properties contain cookie forwarding rules for using HTTP-cookie-based single sign-on:

`wsrp.consumer.cookieforward.LtpaToken2 = alpha.domain.com`
`wsrp.consumer.cookieforward.LtpaToken = alpha.domain.com`

With these settings, the WSRP Consumer forwards the LTPA v1 and LTPA v2 cookies that it received from the clients to the Producers and resources on host `alpha.domain.com`.

`wsrp.consumer.cookieforward.LtpaToken2 = .domain1.com,.domain2.com`

With this setting, the WSRP Consumer forwards the LTPA v2 cookie that it received from the clients to all Producers and resources on hosts in the domains `domain1.com` and `domain2.com`

Related tasks:

"Securing the WSRP Producer by HTTP-cookie-based single sign-on" on page 1449
You can provide security for your WSRP Producer by using HTTP-cookie-based single sign-on (SSO). For using this security option, the WSRP Producer requires no configuration. The WSRP Consumer must be configured to send or forward LTPA V2 single sign-on cookies as part of the WSRP request message to the WSRP Producer.

"Customizing Client Cookie Forwarding" on page 1492
A client can send cookies to the WSRP Consumer as part of an HTTP request. You can customize the WSRP Consumer to forward specific client cookies to the Producer ports or to other resources that are served by the WSRP Consumer as a

proxy.

Configuring WSRP Producer ports for Web Service Security on the Consumer portal:

You can configure each WSRP port of a particular Producer definition for web service security by using LTPA or username tokens.

About this task

If you configure web service security for a Producer port, the WSRP Consumer creates a WS-Security-compliant header. The header contains a security token. When the Producer receives a WSRP request message that contains a WS-Security header, it processes the request under the user identity that is represented by the security token and performs access control for provided portlets.

IBM WebSphere Portal Express Version 8.5 provides three security token types for the most common scenarios. The following list describes these scenarios. In a default portal installation, none of the Producer ports is configured for message authentication or a token type. If your setup does not require security, you do not need to configure the Producer ports.

LTPAv2_Token
The Consumer portal provides an LTPA version 2 token in the WS-Security message header. This token type requires that Consumer and Producer portals share their user registry and LTPA configuration.

LTPA_Token
The Consumer portal provides an LTPA version 1 token in the WS-Security message header. This token type requires that the Consumer and Producer portals share their user registry and LTPA configuration.

Note:

IBM WebSphere Application Server Version 8.5 supports the LTPA v2 token by default. Use the LTPA_Token only if a Producer requires an LTPA v1 token and cannot be configured to use LTPA v2 tokens. A WebSphere Portal Express Version 8.5 Producer does not require LTPA v1 tokens. If you use a WebSphere Portal Express V 8.5 Producer, do not use this token type.

As WebSphere Application Server Version 8.5 does not support LTPA v1 by default, you need to enable the single sign-on interoperability mode in WebSphere Application Server to use LTP v1. To do so, use the single sign-on (SSO) panel within the WebSphere Integrated Solutions Console. For more information about this option, read the documentation about single sign-on settings in the WebSphere Application Server product documentation. If you select this token type and did not enable LTPA v1 tokens before, the WSRP Consumer throws an exception when trying to create the security token for a WSRP request message.

Username_Token
The Consumer portal provides a username token in the WS-Security message header. The username token specifies the user name in clear text.

You can set the token types by either of the following two ways:

Procedure

- You can use the portal administration portlet Web Service Configuration. Proceed as follows:
  1. In the portlet, go to the section for the port settings of the specific Producer for which you want to set the token types.
  2. From the list of service references and token types, select the token type for each port. By default, this list offers the three security token types. If you have defined custom service references, the list also offers these services. You can select either of the token types and custom service references from this list:
     - If you select a token type, the WSRP Consumer uses the default WSRP service reference for this port. Additionally, it includes a security token of the specified type in the WS-Security header of the WSRP request messages.
     - If a custom service reference is available and you select it, the WSRP Consumer uses this service reference. It does not generate extra security tokens.
     - If you do not select anything from this list, the WSRP Consumer uses the default WSRP service reference. It does not generate security tokens.
- You can use the portal XML configuration interface (XMLAccess) to set port specific settings, for example token types. For information about the XML configuration interface and how to use it, read the information about the *XML configuration interface*.

Results

The WSRP Consumer provides a token of the selected type in the WS-Security header of WSRP request messages that are sent to the appropriate Producer port. No further security mechanism, such as message integrity or message confidentiality, is used. If you plan a more complex service configuration or if you plan to use another token type, read *Configuring WSRP web service clients*.

The token types correspond to the default WSRP policy sets and provider policy bindings that are available for the configuration of Producers. The tokens are also compatible to a corresponding WebSphere Portal Express Version 7 or 8 Producer security configuration.

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related tasks:

"Configuring WSRP web service clients" on page 1466
You might want to set up a specific and complex service configuration. In this case, you can configure the WSRP service clients and service references of the WSRP Consumer by using the concept of policy sets. If you intend to configure web service security by using LTPA or username tokens, do not configure the WSRP service clients and service references. In this case, you do not need to read this topic and its subtopics, but follow the procedure described in Configuring WSRP Producer ports for web service security on the Consumer portal.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Configuring WSRP web service clients:

You might want to set up a specific and complex service configuration. In this case, you can configure the WSRP service clients and service references of the WSRP Consumer by using the concept of policy sets. If you intend to configure web service security by using LTPA or username tokens, do not configure the WSRP service clients and service references. In this case, you do not need to read this topic and its subtopics, but follow the procedure described in Configuring WSRP Producer ports for web service security on the Consumer portal.

About this task

The WSRP Consumer of WebSphere Portal Express provides a set of service clients and default service references. The set includes two service clients, one each to support WSRP 1.0 and WSRP 2.0 and the respective port types or web service interfaces. There is one default service reference per service client. You can configure and administer the service clients and service references by using the WebSphere Integrated Solutions Console. For example, you can use the WebSphere Integrated Solutions Console to configure WSRP service providers and WSRP service clients by attaching policy sets. You can configure each service client and service reference separately. If your Consumer portal communicates with Producer portals with different web service configurations, you can also use extra service client references.

The service clients realize the communication with the remote WSRP Producers. They comply to the JAX-WS standard and build upon the JAX-WS web service stack.

The following table lists the supported service clients and service references:

Table 182. WSRP Consumer service clients and service references for the two WSRP versions

	WSRP 1.0	**WSRP 2.0**
Service client	`WSRPService`	`WSRPService_v2`
Service reference	`service/wsrp/WSRPService`	`service/wsrp/ WSRPService_v2`

The service clients support all WSRP service interfaces: Service Description, Markup, Portlet Management, and Registration.

WebSphere Portal Express provides a set of default WSRP policy sets and client policy set bindings. You can use them to configure the WSRP service clients and service references. In this case, you do not have to create your own policy sets and client policy set bindings. By alternative, you can create and use a policy set and client policy set binding of your choice.

For more detailed information about configuring the JAX-WS compliant web service clients, read the WebSphere Application Server product documentation.

Note: The configuration of the WSRP service clients is managed outside WebSphere Portal Express. The portal WSRP Consumer supports all service client configurations that are configured in WebSphere Application Server. This support includes message level security, transport level security, and other quality of service configuration. However, the service client configuration of the WSRP Consumer must be compatible with the web service configuration of the WSRP Producer. Example: If the service providers of the Producer portal are configured for WS-Security, the service references of the Consumer portal must also be configured for WS-Security. Otherwise, the WSRP communication fails.

You can configure your WSRP Consumer portal to consume portlets from Producer portals that have different web service configurations. To do so, you deploy extra service references for the WSRP service clients. You can assign the service references to the ports of a Producer definition. This way, you can configure multiple WSRP Consumer side web service configurations. You can configure each service reference separately. Use this option only if your WSRP Consumer portal communicates with multiple Producer portals that have different web service configurations. For more detailed information, read the following topics.

1. "Communicating with Producer portals with different web service configurations"
   You can use your Consumer portal to communicate with Producer portals that have different web service configurations.
2. "Using the WSRP policy sets and client policy set bindings" on page 1468
   WebSphere Portal Express provides a set of default WSRP policy sets and client policy set bindings. You can use them to configure the WSRP service clients and service references. In this case, you do not have to create your own policy sets and client policy set bindings. By alternative, you can create and use a policy set and client policy set binding of your choice.
3. "Creating and deploying custom service references" on page 1470
   In your WSRP Consumer portal, you can deploy extra service references for the WSRP service clients.

Related information:

Administering web services

Administering message-level security for JAX-WS web services

Securing web services using policy sets

Communicating with Producer portals with different web service configurations:

You can use your Consumer portal to communicate with Producer portals that have different web service configurations.

About this task

For example, such a configuration can be necessary if your WSRP Consumer portal communicates with two separate remote WSRP Producer portals with different web services configurations:

- One of the Producers does not configure Web Services Security for the WSRP web services.
- The other Producer requires WSeb Services Security for its WSRP web services by using a token type that is different from LTPA and username.

In such a case, you deploy an extra service client reference on your WSRP Consumer portal.

This option to deploy extra service references makes it possible to manage multiple WSRP Consumer side web service configurations. If your WSRP Consumer portal does not communicate with multiple Producers that have different web service configurations, you do not need to deploy extra service client references.

If the Producers require web service security based on LTPA or username tokens and do not require further service configuration, you do not need to deploy extra service references. In this case, configure the token type for web service security by following the description in If the Producers require web service security based on LTPA or Username tokens and do not require further service configuration, you do not need to deploy extra service references. In this case, it is recommended to configure the token type for web service security by following the description in*Configuring WSRP Producer ports for message authentication*.

To deploy an extra service client reference for communication with Producers with different web service configurations, proceed as follows:

Procedure

1. Configure the service references separately:
   - Leave the default service reference as it is. Do not configure Web Service Security for the default service reference.
   - Deploy an extra custom service reference and configure it for WS-Security compliant Web Service Security.
2. When you create or edit the Producer definitions that represent the WSRP Producers, you can select the appropriate custom service references for the Producer ports:
   - Select the additional service reference for all ports of the Producer definition that represents the secure Producer.
   - For the non-secure ports of the Producer definition, you do not need to select a service reference. If no specific service reference is selected for a Producer port, the WSRP Consumer portal uses the default service reference.

Related tasks:

"Configuring WSRP Producer ports for Web Service Security on the Consumer portal" on page 1464
You can configure each WSRP port of a particular Producer definition for web service security by using LTPA or username tokens.

Using the WSRP policy sets and client policy set bindings:

WebSphere Portal Express provides a set of default WSRP policy sets and client policy set bindings. You can use them to configure the WSRP service clients and service references. In this case, you do not have to create your own policy sets and client policy set bindings. By alternative, you can create and use a policy set and client policy set binding of your choice.

About this task

The following table describes the provided WSRP application policy sets and the provided WSRP client policy set bindings:

Table 183. Provided WSRP application policy sets and provided WSRP client policy set bindings

	WSRP application policy sets	WSRP client policy set bindings
LTPA based	**LTPA-based message authentication policy set** This policy set defines LTPA token-based message authentication. It does not define other security mechanisms such as message confidentiality, or other web service mechanisms such as WS-Addressing.	**Username and LTPA-based message authentication client binding** You must use this client policy set binding with each of the two WSRP policy sets. It defines the corresponding client binding.
Username based	**Username-based message authentication policy set** This policy set defines Username token-based message authentication. It does not define other security mechanisms such as message confidentiality, or other web service mechanisms such as WS-Addressing.	

The WSRP application policy sets and client policy set bindings are contained in compressed format in the directory `PortalServer/doc/policy-sets-samples` of the portal installation.

For instructions about how to import and attach policy sets and client policy set bindings, read the IBM WebSphere Application Server documentation.

To use the WSRP policy sets and client policy set bindings for service configuration, proceed as follows:

Procedure

1. Import the WSRP policy sets and client policy set bindings. To do so, use a WebSphere Application Server administrative client, such as the WebSphere Integrated Solutions Console:
   a. Open the **Application policy sets** panel.
   b. Select **Import (From Selected Location)**.
   c. Select the LTPA-based message authentication policy `set.zip` file or the username-based message authentication policy `set.zip` file that you want to import.
   d. Open the **General client policy set bindings** panel.
   e. Select **Import (From Selected Location)**.
   f. Select the LTPA-based message authentication client `binding.zip` file or username-based message authentication client `binding.zip` file that you want to import.
2. Attach the policy set and client policy set binding to a WSRP service client or service reference. Proceed as follows:
   a. Open the **Service clients** panel.

b. Open the service reference that you want to configure. Do not select one of the default service references, such as `service/wsrp/WSRPService` or `service/wsrp/WSRPService_v2`.
c. Select the service reference.
d. Select the **Override** option.
e. Use the **Attach Client Policy Set** option to select and attach the LTPA-based message authentication policy set or the username-based message authentication policy set.
f. Select the service reference.
g. Use the **Assign Binding** option to select and assign the username and LTPA-based message authentication client binding.
h. Save your changes to the master configuration.

3. After you have completed this configuration, restart your portal.

Creating and deploying custom service references:

In your WSRP Consumer portal, you can deploy extra service references for the WSRP service clients.

About this task

The service clients and service references are defined in the `web.xml` deployment descriptor of the `wps.war` web application that is contained in the portal enterprise application archive `wps.ear`. To add service references, you export the portal enterprise application, modify the `web.xml` deployment descriptor, and update the portal enterprise application. To add service references for your WSRP Consumer portal, proceed by the steps that are given in the following.

Procedure

1. Export the portal application EAR file. You start with this step so that you can later add the service references in the `web.xml` deployment descriptor. To export the portal application `wps.ear` to a temporary portal EAR file, use a WebSphere Application Server administrative client, for example, the WebSphere Integrated Solutions Console. For details about exporting an enterprise application, read the WebSphere Application Server documentation.
2. Modify the `web.xml` deployment descriptor. To import the portal EAR file and to edit the `web.xml` deployment descriptor, you can either use an assembly tool, or you can extract or open the portal EAR file and edit the `web.xml` deployment descriptor file directly.
3. Add service references to the `web.xml` deployment descriptor. You can add one or multiple service references for the WSRP service clients to the `web.xml`. For more information, read the following topic about *Adding service references to the `wps.war` web application*.
4. Update the portal application. After you modified the `web.xml` deployment descriptor, proceed by one of the following options:
   - Export the portal EAR file from the assembly tool.
   - Save the modified `web.xml` file, and save or compress the portal EAR file. Then, update the portal application from the modified portal EAR file. To do so, you use a WebSphere Application Server administrative client, for example, the WebSphere Integrated Solutions Console. For details about updating an enterprise application, read the WebSphere Application Server documentation.

5. Restart your portal. In a stand-alone installation, restart the portal server. In a cluster configuration, restart the portal on each node.

   "Adding service references to the wps.war web application"
   The WSRP service clients and service references are defined in the web.xml deployment descriptor of the wps.war web application. That web application that is contained in the portal enterprise application archive wps.ear. To add service references, you modify the web.xml deployment descriptor.

Adding service references to the wps.war web application:

The WSRP service clients and service references are defined in the web.xml deployment descriptor of the wps.war web application. That web application that is contained in the portal enterprise application archive wps.ear. To add service references, you modify the web.xml deployment descriptor.

About this task

The following excerpt from file web.xml shows the definition of the default WSRP service references:

```
<service-ref>
    <description>WSRP 1.0 Default Service Reference</description>
    <service-ref-name>service/wsrp/WSRPService</service-ref-name>
    <service-interface>javax.xml.ws.Service</service-interface>
    <service-qname xmlns:pfx="http://www.ibm.com/xmlns/prod/websphere/portal/wsrp/wsdl/v1">
        pfx:WSRPService
    </service-qname>
</service-ref>
<service-ref>
    <description>WSRP 2.0 Default Service Reference</description>
    <service-ref-name>service/wsrp/WSRPService_v2</service-ref-name>
    <service-interface>javax.xml.ws.Service</service-interface>
    <service-qname xmlns:pfx="http://www.ibm.com/xmlns/prod/websphere/portal/wsrp/wsdl/v2">
        pfx:WSRPService_v2
    </service-qname>
</service-ref>
```

You can define extra service references by adding one or more service-ref elements. The following list shows the subelements that a new service-ref element must define:

description
The value for this subelement is user-defined. Specify a text string of your choice.

service-ref-name
The value for this subelement is partly user-defined. Specify service/wsrp/*service-ref-id* , where the prefix service/wsrp/ is fix, and service-ref-id is the user-defined ID of the service reference.

service-interface
The value for this subelement is fix. Specify javax.xml.ws.Service.

service-qname
For the value of this subelement, you can specify one of two options. Depending on your requirements, specify the service Qname for WSRP 1.0 or 2.0 as in one of the examples that are given here:

```
<service-qname xmlns:pfx="http://www.ibm.com/xmlns/prod/websphere/portal/wsrp/wsdl/v1">
   pfx:WSRPService
</service-qname>
```

or:

```
<service-qname xmlns:pfx="http://www.ibm.com/xmlns/prod/websphere/portal/wsrp/wsdl/v2">
   pfx:WSRPService_v2
</service-qname>
```

The following code sample shows how you can add two new service references to the `web.xml` file. Note the position of the new `service-ref` elements immediately after the default service references:

```
service-ref>
    <description>WSRP 1.0 Default Service Reference</description>
    <service-ref-name>service/wsrp/WSRPService</service-ref-name>
    <service-interface>javax.xml.ws.Service</service-interface>
    <service-qname xmlns:pfx="http://www.ibm.com/xmlns/prod/websphere/portal/wsrp/wsdl/v1">
        pfx:WSRPService
    </service-qname>
</service-ref>
    <service-ref>
    <description>WSRP 2.0 Default Service Reference</description>
    <service-ref-name>service/wsrp/WSRPService_v2</service-ref-name>
    <service-interface>javax.xml.ws.Service</service-interface>
    <service-qname xmlns:pfx="http://www.ibm.com/xmlns/prod/websphere/portal/wsrp/wsdl/v2">
        pfx:WSRPService_v2
    </service-qname>
</service-ref>
    <service-ref>
    <description>WSRP 2.0 Alternative Service Reference 1</description>
    <service-ref-name>service/wsrp/AlternativeWSRPService_v2</service-ref-name>
    <service-interface>javax.xml.ws.Service</service-interface>
    <service-qname xmlns:pfx="http://www.ibm.com/xmlns/prod/websphere/portal/wsrp/wsdl/v2">
        pfx:WSRPService_v2
    </service-qname>
</service-ref>
    <service-ref>
    <description>WSRP 2.0 Alternative Service Reference 2</description>
    <service-ref-name>service/wsrp/WSRPService_v2_Second_Alternative</service-ref-name>
    <service-interface>javax.xml.ws.Service</service-interface>
    <service-qname xmlns:pfx="http://www.ibm.com/xmlns/prod/websphere/portal/wsrp/wsdl/v2">
        pfx:WSRPService_v2
    </service-qname>
</service-ref>
```

The example defines two new service references: `service/wsrp/AlternativeWSRPService_v2` and `service/wsrp/WSRPService_v2_Second_Alternative`. After you updated the portal application, you can configure the new service references in a WebSphere Application Server administrative client. After you restart your portal, the WSRP Consumer can find the service references. When you create or edit a Producer definition, you can assign the new service references to Producer ports. For the WSRP Consumer to find the service references, you must name the service reference name according to the following syntax:

service/wsrp/*service-ref-id*
: where the prefix `service/wsrp/` is fixed, and the `service-ref-id` is user-defined.

As the administrator of the Consumer portal, you can assign a service reference to a Producer port of a Producer definition. Use the `service-ref-id` as the identifier for selecting that service reference. If a service reference name of a new service reference does not contain the prefix `service/wsrp/`, the WSRP Consumer cannot find the service reference. When you assign a service reference to a Producer port, the Web Services Configuration portlet shows the list of all defined service references. The list shows only the service reference IDs of the service reference

without the common prefix `service/wsrp`.

Enabling Portal Access Control for a WSRP Consumer portal:

You can configure Portal Access Control for the remote portlets that you consume on your Consumer portal.

Before you begin

When you consume portlets from a Producer portal in your Consumer portal, these remote portlets behave just like local portlets in your portal. You can assign access permissions to users on these portlets by using Portal Access Control.

Working with Producer definitions

To make a WSRP Producer known to your Consumer portal, you create a Producer definition for that WSRP Producer in your Consumer portal. You can also configure the Producer definition.

About this task

When you create a Producer definition, you use the information that you obtained from the Producer to configure the WSRP connection to the Producer. If required, you also register with the Producer.

There are different scenarios for creating a Producer definition:

1. You can use either the Web Service Configuration portlet or the XML configuration interface to create the Producer definition, depending on whether you work online or offline:
   a. If you can connect to the Producer portal and access all files that the Producer's WSDL service definition references, you can work online when you create the Producer definition. In this case, you can use either the Web Service Configuration portlet or the XML configuration interface to create the Producer definition. For example, this can be the case if you want to consume WSRP services in your current portal.
   b. If you cannot connect to the Producer portal and access all files that the Producer's WSDL service definition references, you work offline. In this case, you can use only the XML configuration interface to create the Producer definition. For example, this case can occur if you prepare your Consumer portal in a staging environment and do not connect to the Producer portal to consume WSRP services until you transfer your portal to the next stage.
2. There are three types of Producers. The type of Producer that you create determines which information you must provide with the Producer definition and whether you can create the Producer definition online or offline:
   a. The Producer does not require registration. In this case, you specify the information that you obtained from the Producer. You can create this type of Producer definition in online or offline mode. The IBM WebSphere Portal Express Producer is of this type.
   b. The Producer requires registration and is enabled for WSRP registration. Such a Producer provides a registration port. In this case, you can specify extra registration properties when you create the Producer definition in your Consumer portal. You can create this type of Producers only in online mode.

c. The Producer requires registration and is not enabled for WSRP registration. In this case, you must provide a registration handle that you obtained from the Producer. You can create this type of Producer definition in online or offline mode.

You can create all three types of Producers by using either the Web Service Configuration portlet or the XML configuration interface.

Table 184. Scenarios for creating a Producer definition

Type of Producer	Creating the Producer definition online by using . . .	Creating the Producer definition offline . . .
The Producer does not require registration. In this case you specify the information that you obtained from the Producer.	. . . the Web Service Configuration portlet or the XML configuration interface.	. . . by using the XML configuration interface.
The Producer requires registration and is enabled for WSRP registration. In this case the Producer provides a registration port. In this case, you can specify extra registration properties when you create the Producer definition in your Consumer portal.	. . . the Web Service Configuration portlet or the XML configuration interface.	. . . is not a valid scenario.
The Producer requires registration and is not enabled for WSRP registration. In this case you must provide a registration handle that you obtained from the Producer.	. . . the Web Service Configuration portlet or the XML configuration interface.	. . . by using the XML configuration interface.

Notes:

1. The current implementation of the WSRP Producer in the portal does not support the WSRP registration interface. However, the WSRP Consumer in the portal can handle Producers that support WSRP registration interfaces.
2. When you specify user attributes, make sure to avoid that any of the following events occur:
   - Sending security relevant attributes, such as passwords, over unsecured network connections
   - Passing sensitive data about your users to the Producer.

   "Different types of Producers"
   The information that you must provide with the Producer definition depends on the type of Producer.

   "Using the Web Service Configuration portlet to work with Producer definitions online" on page 1476
   If you work online, you can use the Web Service Configuration portlet to create a Producer definition.

   "Using the XML configuration interface to work with Producer definitions" on page 1477
   You can use the XML configuration interface to work with Producer definitions in different ways.

Different types of Producers:

The information that you must provide with the Producer definition depends on the type of Producer.

There are three different types of Producers. The Producer type influences how you create the Producer definition in your Consumer portals and what information you must provide:

The Producer does not require registration.

The IBM WebSphere Portal Express Producer is of this type. In this case you specify the following information when you create the Producer in your Consumer portal:

- The URL for the WSDL service definitions of the Producer. This item of information is mandatory.
- A name. This item of information is mandatory.
- A description. This item of information is optional.
- User attributes. This item of information is optional. The values for the selected user attributes are later passed on to the Producer when your portal users use the Producer's web service. For example, if you select the attribute for user name, then the Producer's web service can address your portal users by their name. By selecting specific attributes, you prevent sensitive data about your users from being passed to the Producer.

The registration handle and registration properties are not required.

The Producer requires registration and is enabled for WSRP registration.

In this case, the Producer provides a registration port. As a Consumer, you specify the following information when you create the Producer in your portal:

- The URL for the WSDL service definitions of the Producer. This item of information is mandatory.
- A name. This item of information is mandatory.
- A description. This item of information is optional.
- Registration properties. This item of information is optional. The registration properties are passed on to the Producer during your registration. For example, they can provide information about your geographical location, such as the postal code. The Producer can then adapt the web service to your location. If you live near the mountains, the Producer might then provide information or offers for ski vacation or hiking.
- User attributes. This item of information is optional.

The registration handle is not required in this case. After the Consumer completes the registration with the Producer, the Producer passes the registration handle to the Consumer, where it is stored in the Consumer portal database.

The Producer requires registration, but is not enabled for WSRP registration.

In this case, the registration consists of the following steps:

1. The owner of the Consumer portal registers with the owner of the Producer portal by separate communication, for example by email. The Consumer portal owner can provide registration properties. The Consumer portal owner receives the registration handle from the Producer portal owner.
2. The Consumer portal administrator creates the Producer and provides the following information:
   - The URL for the WSDL service definitions of the Producer. This item of information is mandatory.

- A name. This item of information is mandatory.
- A description. This item of information is optional.
- The registration handle that the Consumer received from the Producer by outside communication. This item of information is mandatory.
- User attributes. This item of information is optional.

The Consumer provides no registration properties during creation because the Producer personnel already received this information during the registration by outside communication.

Using the Web Service Configuration portlet to work with Producer definitions online:

If you work online, you can use the Web Service Configuration portlet to create a Producer definition.

Before you begin

Note: Creating the Producer definition by using the Web Service Configuration portlet works only if both the Producer portal and the Consumer portal are online. Under this condition, the Consumer portal can connect to the Producer portal and to all files that are referenced in the Producer's WSDL service definition. If one of the portals is offline and you want to create a Producer definition on the Consumer portal, you must use the XML configuration interface. For example, this case can occur if you prepare your Consumer portal in a staging environment and do not connect to the Producer portal to use WSRP services until you transfer your portal to the next stage.

About this task

When you create a new Producer definition, you do the following tasks:

1. Define the name for the new Producer definition.
2. Give a description for the Producer definition.
3. Specify the URL for the WSDL service description document of the Producer portal.
4. Specify the registration handle for the Producer.

What to do next

Optionally, you can also do the following tasks:

- Set `registration properties` for the Producer definition.
- Set `user attributes` that you want to be passed on to the Producer.
- Set `language-specific names and descriptions` for the Producer definition.
- Assign access permission to your portal users on a Producer definition.
- Delete a Producer definition from your portal.

After you create a Producer definition, you can proceed to using the portlets that are provided by that Producer, that is, integrating them into your Consumer portal as remote portlets.

To work with the **Web Service Configuration** portlet, click the **Administration menu** icon. Then, click **Portlet Management** > **Web Services**.

For details about how to work with the portlet refer to the portlet help.

Using the XML configuration interface to work with Producer definitions:

You can use the XML configuration interface to work with Producer definitions in different ways.

Before you begin

About this task

If you use the XML configuration interface to create a Producer definition, you can work online or offline. Depending on these scenarios, you must consider some differences.

You can also use the XML configuration interface to export a Producer definition, and to create a Producer definition and consume a portlet from that Producer by using a single XML script.

"Creating a Producer definition offline"
When you create a Producer definition offline, be aware which information you must provide.

"Using the XML configuration interface to create a Producer definition" on page 1478
You can use the XML configuration interface to create a Producer definition online or offline.

"Exporting a Producer definition by using the XML configuration interface" on page 1482
You can use the XML configuration interface to export a Producer definition. You might, for example, export the Producer from a test portal to update your production portal with it later.

"Creating a Producer definition and consuming a portlet by a single XML script" on page 1482
You can use a single XML script to create a Producer definition and then consume portlets from that same Producer.

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Creating a Producer definition offline:

When you create a Producer definition offline, be aware which information you must provide.

About this task

If you work offline, that is you cannot connect from your Consumer portal to the Producer portal, you can use only the XML configuration interface to create the Producer definition. In this case you specify the following information:

- The endpoint address URLs to the Service Description and the Markup interfaces of the Producer.
- If the Producer supports the two optional WSRP interfaces Portlet Management and Registration, you specify them as well.

If the Producer requires certain registration properties or a registration handle with the registration by the Consumer, you must specify this information as well.

The XML configuration interface writes all parameters that you provide into the portal database. This includes all of the following information:

- The URL for the Producer's WSDL service definitions
- A name for the Producer definition
- A description for the Producer definition
- The registration handle that you received from the Producer by outside communication
- The registration properties
- The user attributes.

The portal transfers the user attributes and the registration handle to the Producer later when **both** of the following conditions apply:

1. You work online, that is you can connect to the Producer portal.
2. You update the portal configuration with an XML script that includes the scripting code for consuming a portlet.

The user attributes are sent to the Producer with every markup or action request, that is every time when a user interacts with a remote portlet of that Producer.

Using the XML configuration interface to create a Producer definition:

You can use the XML configuration interface to create a Producer definition online or offline.

About this task

The main tag to specify when you use the XML configuration interface to create a Producer is the `wsrp-producer` tag. The following table lists possible **subtags** that you can specify with the `wsrp-producer` tag:

Table 185. Tags for creating a Producer definition

Possible subtag for the `wsrp-producer` tag	Description
`wsdl-url`	This subtag describes the URL to the Producer's WSDL document. This subtag is the only tag that is required for creating a Producer when you use an online connection. The other URLs are extracted from this WSDL file.
`port-type`	Each of the Producer ports (Service Description, Markup, Portlet Management, and Registration) can be described by one tag. These tags are required for offline creation.
`parameter`	This subtag is used for registration properties.

Table 185. Tags for creating a Producer definition (continued)

Possible subtag for the wsrp-producer tag	Description
preferences	This subtag is used for user attributes.
localedata	This subtag is used to specify globalization names and titles.
access-control	This subtag is used to specify access control.

The following table lists possible attributes that you can specify with the wsrp-producer tag:

Table 186. Attributes for creating a Producer definition

Possible attributes for the wsrp-producer tag	Possible values	Description
registration-required	true|false	This attribute indicates whether the Producer requires registration.
force	true|false	This attribute forces creation of the Producer.
default	true|false	Set this attribute to true if this Producer is the default Producer.

Use the following subtags to specify the Producer ports.

Table 187. Subtags for creating a Producer definition

Subtag	Description
secure-url	Specifies the secure URL of the port (HTTPs).
unsecure-url	Specifies the unsecure URL of the port (HTTP).
ws-security-profile	Specifies the Web Service Security token type or service reference that is assigned to the port. The value of the subtag must refer to a defined WSRP service or to one of the defined token types.

Use the following attributes to specify the Producer ports.

Table 188. Attributes for creating a Producer definition

Attribute	Possible values	Description
type	Specify one of the following values: service-description markup portlet-management registration	Defines the port to which the tag applies.

Table 188. Attributes for creating a Producer definition (continued)

Attribute	Possible values	Description
`defaultbinding`	Specify one of the following values: `secure` `unsecure` `onrequest` `undefined`	Defines whether to use the secure or unsecure URL. The value `onrequest` applies to the Markup port only.

The following attributes are listed for the sake of completeness only. Do not change them.

Table 189. Attributes for creating a Producer definition that are not to be changed. These attributes are listed for completeness only. Do not change them.

Possible attribute for the `wsrp-producer` tag	Possible values	Description
`state`	(binary data)	The Producer's state. This attribute is specified as Base64 encoded binary data. It is only used during export and update by the XML configuration interface.
`cookiepolicy`	One of: `none per_user per_group undefined`	The Producer's cookie policy. The policy and possible values are defined in the WSRP specification.
`wsrpversion`	`V1 | V2`	This attribute indicates the WSRP protocol version that is used for communication between consumer and producer.

"XML samples for creating Producer definitions"
You can modify use these XML samples and use them to create Producer definitions,

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

XML samples for creating Producer definitions:

You can modify use these XML samples and use them to create Producer definitions,

XML sample script for creating a Producer definition

You can use these samples to create the Producer definition online or offline.

The following XML sample shows how you use the XML configuration interface to create a Producer if you work offline:

```
<?xml version="1.0" encoding="UTF-8" ?>
<request type="update" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd" create-oids="true">
   <portal action="locate">
      <wsrp-producer action="update" uniquename="ibm.portal.MySampleProducer1">
         <porttype type="service-description" update="set">
             <unsecure-url> http://producer_portal_host:producer_port/wp_contextRoot/WSRPServiceDescriptionService</unsecure-url>
         </porttype>
         <porttype type="markup" update="set">
             <unsecure-url>http producer_portal_host:producer_port/wp_contextRoot/WSRPBaseService</unsecure-url>
         </porttype>
         <localedata locale="en">
             <title>My Sample Producer 1</title>
         </localedata>
      </wsrp-producer>
   </portal>
</request>
```

Replace `http://producer_portal_host:producer_port/wp_contextRoot` with the appropriate values for the environment of your Producer.

This sample specifies the minimum required mandatory WSRP interfaces Service Description and Markup.

XML sample script for creating a Producer definition for a Producer who requires registration

The following XML sample shows how you use the XML configuration interface to create a Producer who requires registration. You can use this sample if you work online and have access to the Producer's WSDL document.

```
<?xml version="1.0" encoding="UTF-8" ?>
<request type="update" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd" create-oids="true">
   <portal action="locate">
      <wsrp-producer action="update" registration-required="true" uniquename="ibm.portal.MySampleProducer2">
         <wsdl-url>  http://producer_portal_host:producer_port/wp_contextRoot/wsdl/wsrp_service.wsdl</wsdl-url>
         <parameter name="regprop1" type="string" update="set">value1</parameter>
         <parameter name="regprop2" type="string" update="set">value2</parameter>
         <preferences name="userattributes" update="set">
            <value>cn</value>
            <value>o</value>
            <value>uid</value>
         </preferences>
         <localedata locale="en">
            <title>Producer 2</title>
         </localedata>
      </wsrp-producer>
   </portal>
</request>
```

Replace `http://producer_portal_host:producer_port/wp_contextRoot/wsdl/wsrp_service.wsdl` with the appropriate values for the environment of your Producer.

This sample also includes specification of user attributes.

To use this sample with a WebSphere Portal Express Producer portal, set `registration-required="false"` and remove the parameter tags. This modification is necessary because the WebSphere Portal Express Producer does not support registration.

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal

configurations.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Exporting a Producer definition by using the XML configuration interface:

You can use the XML configuration interface to export a Producer definition. You might, for example, export the Producer from a test portal to update your production portal with it later.

About this task

The following example shows how you use the portal XML configuration interface to export a Producer definition.

```
<?xml version="1.0" encoding="UTF-8" ?>
<request xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd" type="export">
   <portal action="locate">
      <wsrp-producer action="export" objectid="*" />
   </portal>
</request>
```

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Creating a Producer definition and consuming a portlet by a single XML script:

You can use a single XML script to create a Producer definition and then consume portlets from that same Producer.

About this task

If you consume the portlet in an XML script that has already created the Producer in a previous step, you must specify the following items in the XML script:

1. A `locate` action on the Producer.
2. A `locate` action on the `web-app` tag with the `uid="com.ibm.wps.wsrp.proxyportletapp.webmod"` attribute.
3. The `servlet` subtag with the `remotehandle` and `wsrp-producerref` attributes.
4. A `locate` action on the `portlet-app` subtag with the `uid="com.ibm.wps.wsrp.proxyportletapp"` attribute.
5. The `portlet` subtag with the `servletref` attribute.

The following XML sample shows how you use the XML configuration interface to integrate a remote portlet:

```
<?xml version="1.0" encoding="UTF-8" ?>
<request type="update" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd">
   <portal action="locate">
      <wsrp-producer action="locate" objectid="Producer_OID" uniquename="wps.myProducer1">
      <web-app action="locate" uid="com.ibm.wps.wsrp.proxyportletapp.webmod">
         <servlet action="update" objectid="Servlet_OID" remotehandle="Remote_handle"
             wsrp-producerref="Producer_OID"/>
         <portlet-app action="locate" uid="com.ibm.wps.wsrp.proxyportletapp">
            <portlet action="update" active="true" defaultlocale="en"
               servletref="Servlet_OID" name="Sample_Portlet">
               <localedata locale="en">
                  <title>Sample Portlet</title>
                  <description>Simple sample portlet as Web service</description>
               </localedata>
               <access-control externalized="false" owner="undefined" private="false"/>
            </portlet>
         </portlet-app>
      </web-app>
   </portal>
</request>
```

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Consuming portlets in a Consumer portal

After you create a Producer definition, you can proceed to consume the portlets that are provided by that Producer. This way, you integrate them into your Consumer portal as remote portlets.

About this task

To consume a remote portlet in your Consumer portal, you can use either of the following tools:

- The **Manage Web Modules** portlet.
- The portal XML configuration interface. For more information about the XML configuration interface, read the appropriate topics.

Notes:

1. On the Consumer side, all remote portlets behave like standard API-compliant portlets, independent of their implementation on the Producer side.
2. When a Producer provides a portlet, only settings that are made in the Configure mode of the portlet are available at the consumed remote portlet. Adding a remote portlet to a page on the Consumer side creates a new instance of the provided portlet on the Producer side. But this instance can be modified only on the Consumer and is not visible on the Producer portal.
3. Customization of the consumed portlets by Consumer portal users can be exported by using the XML configuration interface.

Proceed by selecting the appropriate topic from the following links:

"Using the Manage Web Modules portlet to consume portlets from a Producer portal"
To consume portlets from a Producer portal, you use the **Manage Web Modules** portlet and consume a web module.

"Using the XML configuration interface to consume portlets from a Producer portal"
You can use the XML configuration interface (XMLAccess) to consume portlets from a Producer portal.

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Using the Manage Web Modules portlet to consume portlets from a Producer portal:

To consume portlets from a Producer portal, you use the **Manage Web Modules** portlet and consume a web module.

About this task

Click the **Administration menu** icon. Then, click **Portlet Management** > **Web Modules**.

For details about how to work with the portlet refer to the portlet help.

For the relevant topics of the portlet help refer to the following links:

Using the XML configuration interface to consume portlets from a Producer portal:

You can use the XML configuration interface (XMLAccess) to consume portlets from a Producer portal.

About this task

To consume a portlet by using the XML configuration interface, specify the handle of the remote portlet and the Producer portal that provides the remote portlet. Specify this information by using the attributes `remotehandle` and `wsrp-producerref` with the `servlet` subtag. Both values are required to use the remote portlet. The `remotehandle` attribute is defined by the owner of the Producer portal. The owner of the Producer portal provides the handle to you by appropriate means, such as email.

Producers with an IBM WebSphere Portal Express Producer can obtain this information by exporting all provided portlets on their Producer portal by using the XML configuration interface.

Table 190. Tags for integrating WSRP services

subtag	Attribute for the subtag	Description
`servlet`	`remotehandle`	As provided by the Producer
`servlet`	`wsrpproducerref`	ID of the Producer

After successful integration, the remote portlets are available in the portal administration. They are handled in the same manner as local portlets.

Notes:

- You can consume a portlet only if you work online and can access the Producer's WSDL document.
- To obtain a list and descriptions of the portlets that a specific Producer provides, the WSRP implementation of the portal uses the discovery mechanism that is defined in the WSRP standard.
- An integrated portlet is always treated as a standard API-compliant portlet.

To delete integrated remote portlets, use the Manage Web Modules portlet.

"Creating a Producer definition and consuming a portlet by one single XML script"
You can use a single XML script to create a Producer definition and then consume portlets from that same Producer.

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Creating a Producer definition and consuming a portlet by one single XML script:

You can use a single XML script to create a Producer definition and then consume portlets from that same Producer.

About this task

If you consume the portlet in an XML script that has already created the Producer in a previous step, you must specify the following items in the XML script:

1. A `locate` action on the Producer.
2. A `locate` action on the `web-app` tag with the `uid="com.ibm.wps.wsrp.proxyportletapp.webmod"` attribute.
3. The `servlet` subtag with the `remotehandle` and `wsrp-producerref` attributes.
4. A `locate` action on the `portlet-app` subtag with the `uid="com.ibm.wps.wsrp.proxyportletapp"` attribute.
5. The `portlet` subtag with the `servletref` attribute.

The following XML sample shows how you use the XML configuration interface to integrate a remote portlet:

```
<?xml version="1.0" encoding="UTF-8" ?>
<request type="update" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd">
   <portal action="locate">
      <wsrp-producer action="locate" objectid="Producer_OID" uniquename="wps.myProducer1">
      <web-app action="locate" uid="com.ibm.wps.wsrp.proxyportletapp.webmod">
         <servlet action="update" objectid="Servlet_OID" remotehandle="Remote_handle"
             wsrp-producerref="Producer_OID"/>
         <portlet-app action="locate" uid="com.ibm.wps.wsrp.proxyportletapp">
            <portlet action="update" active="true" defaultlocale="en"
               servletref="Servlet_OID" name="Sample_Portlet">
               <localedata locale="en">
                  <title>Sample Portlet</title>
                  <description>Simple sample portlet as Web service</description>
               </localedata>
               <access-control externalized="false" owner="undefined" private="false"/>
            </portlet>
         </portlet-app>
      </web-app>
   </portal>
</request>
```

Customizing the WSRP configuration of your Consumer portal

You can customize some aspects of you WSRP Consumer portal.

About this task

This customization is optional.

"Customizing the WSRP resource proxy"
The WSRP resource proxy is used by the WSRP Consumer to load resources that are referenced by remote portlets.

CF05 "Configure timeout properties for the WSRP communication" on page 1490
You can configure web service timeout properties for the WSRP communication on the Consumer. You can configure the timeout properties in the WP Configuration Service or as a preference specifically for remote portlets.

CF05 "Configure a limit for the size of file uploads" on page 1491
You can configure a limit for the size of file uploads during an action request. You can also configure the Consumer behavior if a file upload exceeds the defined limit.

CF05 "Switch off the caching of Producer service descriptions" on page 1491
By default, the Consumer portal caches the WSRP service descriptions that it receives from Producers. If required, you can switch off the caching.

"Customizing Client Cookie Forwarding" on page 1492
A client can send cookies to the WSRP Consumer as part of an HTTP request. You can customize the WSRP Consumer to forward specific client cookies to the Producer ports or to other resources that are served by the WSRP Consumer as a proxy.

CF06 "Configuring remote session invalidation" on page 1493
You can configure the WSRP Consumer to invalidate the remote session when a user explicitly logs out of the Consumer portal. If you enable remote session invalidation and a user logs out of the Consumer portal, the Consumer sends a `releaseSessions` WSRP request to all the Producers with which the user interacted. The Producers portals can then invalidate these sessions.

Customizing the WSRP resource proxy:

The WSRP resource proxy is used by the WSRP Consumer to load resources that are referenced by remote portlets.

About this task

Depending on your environment, it can be necessary to customize the resource proxy behavior with regards to the following aspects:

- SSL settings for secure connections to remote resources
- LTPA token forwarding for single sign-on scenarios
- Proxy server settings for outbound connections
- Support for relative URLs
- HTTP basic authentication for outbound connections
- HTTP headers that are not forwarded.

"Customizing the WSRP resource proxy SSL settings"
The WSRP resource proxy uses one SSL configuration for all secure outbound connections.

"Customizing the WSRP resource proxy for LTPA token forwarding" on page 1488
The WSRP resource proxy can forward single sign-on cookies (LTPA, LTPA2) from the client requests to resources in the same single sign-on domain.

"Customizing the WSRP resource proxy for proxy server support" on page 1488
The WSRP resource proxy supports HTTP proxy servers. It can connect directly to remote resources or to a proxy server that forwards the requests to remote resources.

"Disabling support for relative URLs for the WSRP resource proxy" on page 1489
By default, the WSRP resource proxy serves resources relative to the current resource proxy URI. You can disable the support for relative URLs.

"Customizing the WSRP resource proxy for basic authentication" on page 1489
You can customize the WSRP resource proxy for HTTP basic authentication.

"Customizing the WSRP resource proxy HTTP header forwarding behavior" on page 1490
By default, the WSRP resource proxy forwards all HTTP headers from the client request except for the host header and cookie headers. You can define further headers that you do not want to be forwarded.

Customizing the WSRP resource proxy SSL settings:

The WSRP resource proxy uses one SSL configuration for all secure outbound connections.

About this task

By default, the global SSL configuration as defined in the HTTP Client Service properties is taken for the WSRP resource proxy as well. To specify a different SSL configuration for the WSRP resource proxy only, you can overwrite the global configuration. To do so, proceed as follows:

Procedure

1. Set the following property in the HTTP Client Service:
   `wsrp.resourceproxy.ssl.configuration = SSL configuration name`. To specify the value, use the name of an SSL configuration as defined in the WebSphere Application Server security configuration that you want to use for the WSRP resource proxy.
2. Restart the portal or the cluster for the new setting to become active.

Customizing the WSRP resource proxy for LTPA token forwarding:

The WSRP resource proxy can forward single sign-on cookies (LTPA, LTPA2) from the client requests to resources in the same single sign-on domain.

About this task

You can influence the LTPA token forwarding behavior. To do so, proceed as follows:

Procedure

1. Set the following property in the HTTP Client Service: `wsrp.resourceproxy.sso.domain = Single sign-on domain`. Use this property to specify the single sign-on domain. If the WSRP resource proxy loads a resource from a host inside this domain and the client request contains LTPA or LTPA2 cookies, these cookies are forwarded to the remote resource.
2. Restart the portal or the cluster for the new setting to become active.

Related tasks:

"Enabling remote rendering with WSRP and the Web Content Viewer" on page 2069
To display web content on a portal that does not include IBM Web Content Manager, you can use the Web Content Viewer and the WSRP support in the portal. The Web Content Viewer can then retrieve and display content from a web content system on a different server.

Customizing the WSRP resource proxy for proxy server support:

The WSRP resource proxy supports HTTP proxy servers. It can connect directly to remote resources or to a proxy server that forwards the requests to remote resources.

About this task

By default, the global proxy configuration as defined in the HTTP Client Service properties is taken for the WSRP resource proxy as well. To specify a different proxy configuration for the WSRP resource proxy only, you can overwrite the global configuration by setting a combination of the following properties in the HTTP Client Service:

wsrp.resourceproxy.proxy.http.host
Use this property to specify a proxy host for HTTP URLs. If no proxy host is set, the portal tries to load all HTTP URLs directly.

wsrp.resourceproxy.proxy.http.port
Use this property to specify the port for the HTTP proxy. If no value is specified, 80 is used as the default value.

wsrp.resourceproxy.proxy.https.host
Use this property to specify a proxy host for HTTPs URLs. If no proxy host is set, WebSphere Portal tries to load all HTTPs URLs directly.

wsrp.resourceproxy.proxy.https.port
Use this property to specify the proxy port for HTTPs URLs. If no value is specified, 443 is used as the default value.

wsrp.resourceproxy.proxy.auth.credentialslot
Set this property if you want proxy authentication to be used for connections that use a proxy server. You must provide the user ID and

password in a credential slot of the portal credential vault. You then specify the name of this credential slot in this property. The credential must have the type `UserPasswordPassive`. Proxy authentication applies to the proxy server only, not to the target server of the outbound connection.

wsrp.resourceproxy.proxy.excludehost
Use this property to specify a particular host for which no proxy connection is used, even if a proxy is configured. You can set this property more than once. Specify one setting for each host that is excluded from proxy connections.

Note: You can explicitly set no value for a property that has a value in the global configuration. To do so, specify the value `none` to overwrite the global configuration value. Restart the portal or the cluster for the changes to take effect.

Disabling support for relative URLs for the WSRP resource proxy:

By default, the WSRP resource proxy serves resources relative to the current resource proxy URI. You can disable the support for relative URLs.

About this task

If you disable the support for relative URLs, the resource proxy accepts only requests for URLs that are rewritten and signed by the WSRP Consumer portal itself. URLs that are manipulated at the client side, for example by modifying URL parameters with JavaScript functions, are discarded.

To switch off the support for relative URLs, access the WebSphere Integrated Solutions Console and set the following property in the WP Configuration Service: `wsrp.resourceproxy.support.relative = false`. Restart the portal or the cluster for the changes to take effect.

Customizing the WSRP resource proxy for basic authentication:

You can customize the WSRP resource proxy for HTTP basic authentication.

About this task

To do so, set the following property in the WP Configuration Service:

wsrp.resourceproxy.basic.auth.credentialslot = *your_credential_slot*
Use this property to specify a credential vault slot that contains the user ID and password credentials. The resource proxy servlet uses the credentials from the credential vault slot when it fetches resources that are protected by HTTP basic authentication. The user ID and password are sent to all remote resources that are referenced in the markup of the remote WSRP portlet. By default no value is set for this property.

After you set the property, restart the portal or the cluster for your changes to take effect.

For details about creating a credential vault slot and about how to set configuration properties refer to the following links:

Customizing the WSRP resource proxy HTTP header forwarding behavior:

By default, the WSRP resource proxy forwards all HTTP headers from the client request except for the host header and cookie headers. You can define further headers that you do not want to be forwarded.

About this task

To do so, set the following property in the Configuration Service: `wsrp.resourceproxy.no.header.forwarding = comma-separated list of header names`. Use this property to specify the list of HTTP headers that are not forwarded from the client request in addition to the host header and cookie headers. The host header and cookie headers are never forwarded independent of how this property is set. The default behavior is that WSRP forwards all headers except for the host header and cookie headers. After you set this property, restart the portal or the cluster for the change to take effect.

Configure timeout properties for the WSRP communication:

You can configure web service timeout properties for the WSRP communication on the Consumer. You can configure the timeout properties in the WP Configuration Service or as a preference specifically for remote portlets.

For a description of the timeout properties and the respective default values, go to *JAX-WS timeout properties* in the WebSphere Application Server IBM Knowledge Center.

To define the timeout properties, configure the following configuration parameters on the Consumer:

```
wsrp.consumer.connectiontimeout = (timeout in seconds)

wsrp.consumer.writetimeout = (timeout in seconds)

wsrp.consumer.responsetimeout = (timeout in seconds)
```

By default each of these parameters is `undefined`. The default setting means that the Consumer does not set a timeout property on an outgoing WSRP request. In this case, WebSphere Application Server uses the default timeout properties as described in *JAX-WS timeout properties.*

If timeout properties are configured, the Consumer passes the timeout properties to WebSphere Application Server in the *MessageContext* of an outgoing WSRP request. You therefore do not need to configure timeout settings in WebSphere Application Server in a policy set.

You can set these parameters specifically for a remote portlet. To do so, set this parameter in the portal WP Configuration Service by using the WebSphere Integrated Solutions Console.

If you set a parameter both as a preference of a remote portlet and in the WP Configuration Service, the value that is defined in the preference of the remote portlet takes precedence.

Related information:

CF05 JAX-WS timeout properties

Configure a limit for the size of file uploads:

You can configure a limit for the size of file uploads during an action request. You can also configure the Consumer behavior if a file upload exceeds the defined limit.

To configure set the following configuration parameters on the Consumer:

wsrp.maxUploadDataLength = (the maximum size in Kilo Bytes, a integer value > 0)
Use this parameter to define a maximum size of upload data. By default this parameter is undefined. The default setting means that the Consumer does not limit the size of upload data. Also, if you specify a negative value, the Consumer does not limit the size of the upload data.

wsrp.uploadErrorHandledByProducer = (true,false)
Use this parameter to define the Consumer behavior if a file upload exceeds the defined limit. The default for this parameter is true. This default setting means that if the upload data exceeds the defined limit, the Consumer continues processing. The Consumer sends a WSRP request, which is cleared of all upload data, to the Producer. The remote portlet needs to handle this situation.

If this parameter has the value `false` and if upload data exceeds the defined limit, the Consumer stops processing this action request by throwing an exception. The Consumer does not send a WSRP request to the Producer.

You can set these parameters specifically for a remote portlet. To do so, set this parameter as a preference in the portlet definition of the remote portlet on the Consumer.

Alternatively, you can set this parameter for all remote portlets on the Consumer. To do so, set this parameter in the portal WP Configuration Service by using the WebSphere Integrated Solutions Console.

If you set a parameter both as a preference of a remote portlet and in the WP Configuration service, the value that is defined in the preference of the remote portlet takes precedence.

Switch off the caching of Producer service descriptions:

By default, the Consumer portal caches the WSRP service descriptions that it receives from Producers. If required, you can switch off the caching.

The caching of the WSRP service descriptions in the Consumer is determined by the portal Cache Manager Service property `cacheinstance.wsrp.cache.servicedescription.enabled`. The default setting is `true`. The caching increases performance and reduces network traffic.

If you encounter problems, or if you want to turn off caching of the service description for other reasons, switch off the caching. To do so, set the property `cacheinstance.wsrp.cache.servicedescription.enabled = false` in the WP Cache Manager Service. You set this parameter by using the WebSphere Integrated Solutions Console.

Customizing Client Cookie Forwarding:

A client can send cookies to the WSRP Consumer as part of an HTTP request. You can customize the WSRP Consumer to forward specific client cookies to the Producer ports or to other resources that are served by the WSRP Consumer as a proxy.

About this task

By default the WSRP Consumer does not forward client cookies to WSRP Producers or other resources. To customize the WSRP Consumer to forward specific client cookies, proceed as follows:

Procedure

1. Set the following property in the HTTP Client Service:
   `wsrp.consumer.cookieforward.`*`cookiename`* `=` *`hostdomainnames`* `[;` *`cookiepriority`*`]`

   Specify a separate property for each client cookie that you want to be forwarded.
2. Restart the portal or the cluster for the new setting to become active.

Results

Description:

wsrp.consumer.cookieforward.*cookiename* = *hostdomainnames* [; *cookiepriority*]
This property defines a cookie forwarding rule for the cookie that is identified by the value for *cookiename*. The cookie forwarding rule specifies a list of host names and domain names and optionally specifies the cookie priority.

cookiename
The value for *cookiename* must match the cookie name literally.

hostdomainname
This setting comprises a comma-separated list of host names and domain names. The cookie is forwarded to Producers whose WSRP endpoint address URLs match to entries in the host domain name list. The WSRP Consumer also forwards the client cookies to resources that are served by the WSRP Consumer as a proxy and whose URLs accord to the host domain name list. The host domain name can contain host names, such as `alpha.company.com`, or domain names that are identified by a leading period (`.`) such as `.sample.com`. The WSRP Consumer literally matches the host domain names to the Producer WSRP endpoint address URLs and resource URLs. It does not translate IP addresses into host domain names or vice versa.

cookiepriority
This setting is optional. You can use it to specify the cookie priority. The WSRP Consumer uses the cookie priority to control how priority conflicts between client cookies and producer set cookies with identical names are resolved. Valid values are as follows:

clientfirst
This priority setting directs the WSRP Consumer to prioritize a client cookie over an identically named cookie

that was set by the Producer. In other words, it instructs the WSRP Consumer to send the client cookie to the WSRP Producer or to the Producer resource, rather than to send the Producer cookie.

clientlast
This priority setting directs the WSRP Consumer to prioritize a cookie that was set by the WSRP Producer over a client cookie. In other words, it instructs the WSRP Consumer to send the cookie that was set by the WSRP Producer to the WSRP Producer or to the Producer resource, rather than to send the client cookie. This value is the default priority value. For example, it is applied if the cookie priority is not specified in the cookie forwarding rule.

Example properties that contain cookie forwarding rules are as follows:

wsrp.consumer.cookieforward.SAMPLECOOKIE = alpha.company.com,.sample.org
With this setting, the WSRP Consumer forwards the client cookie named `SAMPLECOOKIE` to the host `alpha.company.com` and to all hosts contained in the domain `sample.org`. It assumes the default cookie priority `clientlast` .

wsrp.consumer.cookieforward.COOKIE2 = .myorg.com,beta.sample.com;clientfirst
With this setting, the WSRP Consumer forwards the client cookie named `COOKIE2` to the host `beta.sample.com` and to all hosts in the domain `myorg.com`. It gives client cookies first priority.

Note: The WSRP Consumer does not send cookies when it requests the WSDL service description documents from a Producer.

Related concepts:

"Cookie support" on page 1500
The WSRP Consumer stores cookies that it receives from a WSRP Producer or from resources that the Consumer serves as a proxy. It forwards the cookies appropriately in subsequent requests to a Producer or other resources.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Configuring remote session invalidation: CF06

You can configure the WSRP Consumer to invalidate the remote session when a user explicitly logs out of the Consumer portal. If you enable remote session invalidation and a user logs out of the Consumer portal, the Consumer sends a `releaseSessions` WSRP request to all the Producers with which the user interacted. The Producers portals can then invalidate these sessions.

About this task

By default, remote session invalidation is disabled.

Before you enable remote session invalidation, carefully consider the performance impact:

- Remote session invalidation allows the Producer portal to invalidate unused sessions. This way, the Producer can free resources when the users log out.
- However, remote session invalidation creates extra WSRP communication during logout. There is at least one WSRP roundtrip to each Producer with which the user interacted. In some cases, the Consumer needs to send multiple `releaseSessions` requests to one Producer.

To configure remote session invalidation, set the following configuration property on the Consumer portal:

`wsrp.consumer.releaseSessions.enabled = (false|true)`
Use this property to enable or disable remote session invalidation. The default for this property is `false`. This default setting means that the Consumer does not invalidate remote sessions.

You set this property in the portal WP Configuration Service by using the WebSphere Integrated Solutions Console. For details about portal service configuration properties and how to set them, read Portal service configuration and Setting service configuration properties.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Related reference:

"Portal service configuration" on page 289
IBM WebSphere Portal Express comprises a framework of configuration services to accommodate the different scenarios that portals of today need to address. You can configure some of these services.

Using handlers for WSRP web services

You can extend your WSRP Producer or WSRP Consumer portal by handlers that comply with JAX-WS.

About this task

The WSRP web services conform to the JAX-WS standard. To create and process custom extensions of WSRP messages, you can create and deploy JAX-WS compliant handlers for the WSRP web services.

By default, the WSRP Producer and WSRP Consumer do not use any handler.

Migration note: Up to WebSphere Portal Express Version 8.0, WSRP was based on the JAX-RPC standard. If you upgrade your WebSphere Portal Express from Version 8.0 to Version 8.5, you must reimplement your existing JAX-RPC compliant handlers to comply with JAX-WS.
Before you can use a custom handler, you must first create a handler implementation according to the JAX-WS specification. For details about the handler framework, read the JAX-WS specification. For information about how to use handlers with JAX-WS web services, read the appropriate information in the WebSphere Application Server documentation.

"Using a handler on a WSRP Producer portal"
Find the procedure for configuring and using a handler on a WSRP Producer portal.

"Using a handler on a WSRP Consumer portal" on page 1496
Find the procedure for configuring and using a handler on a WSRP Consumer portal.

Related information:

JAX-WS specification

Using handlers in JAX-WS web services

Using a handler on a WSRP Producer portal

Find the procedure for configuring and using a handler on a WSRP Producer portal.

About this task

You can configure handlers for the WSRP Producer in the `wps.ear` portal application.

CF05 Starting with CF05, you can register handlers that implement the SOAPHandler interface as an extension of a WSRP extension point. In this case, you do not need to modify the portal application. For details, read the information in a later section.

To configure a JAX-WS handler in the portal application, use the following procedure:

Procedure

1. Create a handler implementation according to the JAX-WS specification. For details about the handler framework, read the JAX-WS specification. For information about using handlers with JAX-WS web services, read the appropriate information in the WebSphere Application Server documentation.
2. Provide the handler implementation in the class path of the `wps.ear` portal application, for example in the portal shared application directory.
3. Export the `wps.ear` portal application. The exported `wps.ear` file contains a `wps.war`. The `WEB-INF/classes` directory in the `wps.war` file contains two files `wp.wsrp.producer-v1-handlerchain.xml` and `wp.wsrp.producer-v2-handlerchain.xml`.
4. Modify either one of the files `wp.wsrp.producer-v1-handlerchain.xml` or `wp.wsrp.producer-v2-handlerchain.xml`. Update the default `handler-chains` element by adding one or more `handler` elements.
5. Update the `wps.ear` portal application.
6. Restart your WSRP Producer portal.

Results

You can now use the handler in your Producer portal

Example

The following samples show excerpts from the file `wp.wsrp.producer-v2-handlerchain.xml`. The samples contain a handler definition for the WSRP Producer.

Before CF 05:

```
<?xml version="1.0" encoding="UTF-8"?>
<ns:handler-chains xmlns:ns="http://java.sun.com/xml/ns/javaee">
    <ns:handler-chain name="wp.wsrp.producer-v2-handlerchain">
        <ns:handler>
            <ns:handler-class>sample.handler.ProducerHandler</ns:handler-class>
        </ns:handler>
    </ns:handler-chain>
</ns:handler-chains>
```

CF05 With CF 05 and later fix packs, you must not modify the default handler definition for the handler class `com.ibm.wps.wsrp.producer.handler.ProducerHandlerChain`. You can add more handlers as shown in the following example:

```
<?xml version="1.0" encoding="UTF-8"?>
<ns:handler-chains xmlns:ns="http://java.sun.com/xml/ns/javaee">
    <ns:handler-chain name="wp.wsrp.producer-v2-handlerchain">
        <ns:handler>
            <ns:handler-class>com.ibm.wps.wsrp.producer.handler.ProducerHandlerChain</ns:handler-class>
        </ns:handler>
        <ns:handler>
            <ns:handler-class>sample.handler.ProducerHandler</ns:handler-class>
        </ns:handler>
    </ns:handler-chain>
</ns:handler-chains>
```

Related information:

JAX-WS specification

Using handlers in JAX-WS web services

Registering a SOAPHandler as a WSRP extension: CF05

About this task

CF05 To register a handler that implements the SOAPHandler interface, you can use a WSRP extension point. In this case, you do not need to modify the portal application. To do so, provide a JAR file that contains the handler implementation and a `plugin.xml` file that defines an extension of extension point `com.ibm.portal.wsrp.producer.SOAPHandler`. You can also define multiple handlers inside one `plugin.xml` file as shown in the following example. The class attribute must specify the handler implementation class.

```
<?xml version="1.0" encoding="UTF-8"?>
<plugin
    id="sample.handler"
    name="Sample WSRP Handlers"
    version="1.0.0">

    <extension point="com.ibm.portal.wsrp.producer.SOAPHandler" id="SampleProducerHandler1">
        <impl class="com.ibm.wps.wsrp.test.handler.ProducerHandler1" weight="100"/>
    </extension>
    <extension point=" com.ibm.portal.wsrp.producer.SOAPHandler" id="SampleProducerHandler2">
        <impl class="com.ibm.wps.wsrp.test.handler.ProducerHandler2" weight="110"/>
    </extension>
</plugin>
```

Using a handler on a WSRP Consumer portal

Find the procedure for configuring and using a handler on a WSRP Consumer portal.

About this task

You can configure handlers for the WSRP Consumer in the `web.xml` file of the `wps.ear` portal application.

CF05 Starting with CF05, you can register handlers that implement the SOAPHandler interface as an extension of a WSRP extension point. In this case, you do not need to modify the `web.xml` file of the portal application. For details, read the information in a later section.

To configure a JAX-WS handler in the `web.xml` file of the portal application, use the following procedure:

Procedure

1. Create a handler implementation according to the JAX-WS specification. For details about the handler framework, read the JAX-WS specification. For information about using handlers with JAX-WS web services, read the appropriate information in the WebSphere Application Server documentation.
2. Provide the handler implementation in the class path of the `wps.ear` portal application, for example in the portal shared application directory.
3. Export the `wps.ear` portal application. The exported `wps.ear` file contains a `wps.war` file, which in turn contains a file that is named `web.xml`.
4. Modify the `web.xml` file by adding a `handler-chains` element to the service reference for which you want to deploy the handler.
5. Update the `wps.ear` portal application.
6. Restart your WSRP Consumer portal.

Results

You can now use the handler in your Consumer portal

Example

The following samples show excerpts from the file `web.xml`. The samples contain a handler definition for the WSRP Consumer.

Before CF 05:

```
<service-ref>
   <description>WSRP 2.0 Default Service Reference</description>
   <service-ref-name>service/wsrp/WSRPService_v2</service-ref-name>
   <service-interface>javax.xml.ws.Service</service-interface>
   <service-qname
      xmlns:pfx="http://www.ibm.com/xmlns/prod/websphere/portal/wsrp/wsdl/v2">pfx:WSRPService_v2</service-qname>
   <handler-chains>
      <handler-chain>
         <handler>
            <handler-name>sample handler</handler-name>
            <handler-class>sample.handler.ConsumerHandler</handler-class>
         </handler>
      </handler-chain>
   </handler-chains>
</service-ref>
```

CF05 With CF 05 and later fix packs, you must not modify the default handler definition named `WSRP 2.0 Default Consumer Handler`. You can add more handlers as shown in the following example:

```
<service-ref>
   <description>WSRP 2.0 Default Service Reference</description>
   <service-ref-name>service/wsrp/WSRPService_v2</service-ref-name>
   <service-interface>javax.xml.ws.Service</service-interface>
   <service-qname
```

```
      xmlns:pfx="http://www.ibm.com/xmlns/prod/websphere/portal/wsrp/wsdl/v2">pfx:WSRPService_v2</service-qname>
   <handler-chains>
      <handler-chain>
         <handler>
            <handler-name>WSRP 2.0 Default Consumer Handler</handler-name>
            <handler-class>com.ibm.wps.wsrp.consumer.handler.ConsumerHandlerChain</handler-class>
         </handler>
         <handler>
            <handler-name>sample handler</handler-name>
            <handler-class>sample.handler.ConsumerHandler</handler-class>
         </handler>
      </handler-chain>
   </handler-chains>
</service-ref>
```

Related information:

JAX-WS specification

Using handlers in JAX-WS web services

Registering a SOAPHandler as a WSRP extension: CF05
About this task

CF05 To register a handler that implements the SOAPHandler interface, you can use a WSRP extension point. In this case, you do not need to modify the `web.xml` file of the portal application. To do so, provide a JAR file that contains the handler implementation and a `plugin.xml` file that defines an extension of extension point `com.ibm.portal.wsrp.consumer.SOAPHandler`. You can also define multiple handlers inside one `plugin.xml` file as shown in the following example. The class attribute must specify the handler implementation class.

```
<?xml version="1.0" encoding="UTF-8"?>
<plugin
   id="sample.handler"
   name="Sample WSRP Handlers"
   version="1.0.0">

   <extension point="com.ibm.portal.wsrp.consumer.SOAPHandler" id="SampleConsumerHandler1">
      <impl class="com.ibm.wps.wsrp.test.handler.ConsumerHandler1" weight="100"/>
   </extension>
   <extension point=" com.ibm.portal.wsrp.consumer.SOAPHandler" id="SampleConsumerHandler2">
      <impl class="com.ibm.wps.wsrp.test.handler.ConsumerHandler2" weight="110/>
   </extension>
</plugin>
```

Reference for using WSRP with the portal

Reference information about using WSRP with the portal includes WSRP markup caching and Known limitations.

"WSRP Markup Caching" on page 1499
To improve performance, the WSRP implementation in the portal can use expiry-based markup caching for remote portlets. This caching reduces the number of interactions between the Consumer and the Producer. The Consumer caches remote portlet content, based on the cache-control data structure that the Producer provides as part of its response. You can enable markup caching on the Consumer for all remote portlets or specifically for selected remote portlets.

"Cookie support" on page 1500
The WSRP Consumer stores cookies that it receives from a WSRP Producer or from resources that the Consumer serves as a proxy. It forwards the cookies appropriately in subsequent requests to a Producer or other resources.

"WSRP two-phase rendering" on page 1501
The WSRP Consumer and the WSRP Producer in the portal support two-phase rendering for JSR 286 portlets. Two-phase rendering allows a remote portlet to set headers and cookies and to modify the HTML head section.

"Handling HTTP headers" on page 1502
You can configure both the WSRP Producer and the Consumer to ignore headers.

"Hints and tips for using WSRP with the portal" on page 1502
Here are some hints and tips for using WSRP with IBM WebSphere Portal Express.

"Troubleshooting WSRP" on page 1504
You can troubleshoot WSRP by using different methods such as logging and tracing, debugging, and monitoring.

WSRP Markup Caching

To improve performance, the WSRP implementation in the portal can use expiry-based markup caching for remote portlets. This caching reduces the number of interactions between the Consumer and the Producer. The Consumer caches remote portlet content, based on the cache-control data structure that the Producer provides as part of its response. You can enable markup caching on the Consumer for all remote portlets or specifically for selected remote portlets.

To enable the use of remote caches, the Consumer can use the cache control data structure to set the HTTP cache control header of a resource response or a render response during the render headers phase.

CF05 Starting with Portal 8.5 CF05, markup caching is no longer based on the portlet container's fragment caching feature. You can now use WSRP markup caching without enabling fragment caching. The WSRP Consumer uses the Portal cache `wsrp.cache.markup` to store WSRP *getMarkup* responses according to the cache control.

The Producer derives the WSRP cache control from the cache settings of the local standard API portlet. These settings are specified statically in the portlet definition or dynamically through the Portlet API during run time. They comprise the following information:

- **`Expiration`** specifies the duration in seconds that markup fragment remains valid. A value of `-1` indicates that the markup fragment never expires.

  Note: If an expiration value of `-1` is specified, the Consumer and all remote caches cache the content for unlimited time. This way, the content is never updated.
- **`Scope`** specifies a string that indicates when the cached markup can be used by various users. The markup is either cached specifically for one user or for all users. This parameter is relevant for remote caching. For more information, read the section about tuning your portal.

If the local portlet is not a standard API portlet, the Producer does not return any cache control information. It disables caching for this portlet.

The Consumer can configure WSRP markup caching by using the following configuration parameters:

CF05 **`wsrp.markupcaching.enabled = (false,true)`**
Use this parameter to enable or disable WSRP markup caching. The default for this parameter is `false`. This default setting means that WSRP markup caching is disabled, if no value is specified for this parameter.

Note: If the parameter **wsrp.requiresSeparateRenderPhase** is enabled, the Consumer automatically disables WSRP markup caching for the corresponding portlet and does not consider this parameter.

wsrp.caching.enabled = (true, false)
Use this parameter to enable or disable setting the HTTP cache control header in a resource response or in a render response during the render headers phase. The default for this parameter is `true`.

CF05 You can set these parameters specifically for a remote portlet. To do so, set this parameter as a preference in the portlet definition of the remote portlet on the Consumer.

Alternatively, you can set these parameters for all remote portlets on the Consumer. To do so, set this parameter in the portal WP Configuration Service by using the WebSphere Integrated Solutions Console.

CF05 If you set a parameter both as a preference of a remote portlet and in the WP Configuration Service, the value that is defined in the preference of the remote portlet takes precedence.

Cookie support

The WSRP Consumer stores cookies that it receives from a WSRP Producer or from resources that the Consumer serves as a proxy. It forwards the cookies appropriately in subsequent requests to a Producer or other resources.

The following list gives an overview of the supported use cases of the WSRP Consumer cookie handling.

Cookies that the Consumer receives from the Producer during session context initialization for a remote portlet:
If the Consumer receives a cookie from the Producer during session context initialization for a remote portlet, the Consumer forwards that cookie to remote portlets of the same Producer or of the same group. The forwarding target depends on the Producer initialization cookie policy. Additionally, the Consumer forwards the cookie to resources that the Consumer serves as a proxy and that the remote portlet addresses.

Cookies that the Consumer receives from a remote JSR 286 portlet:
If the Consumer receives a cookie from a remote JSR 286 portlet, the Consumer forwards that cookie to remote portlets and resources that the Consumer serves as a proxy, if the cookie domain and path match.

Cookies that the Consumer receives from resources that the Consumer serves as a proxy:
If the Consumer receives a cookie from resources that the Consumer serves as a proxy, the Consumer forwards that cookie to remote portlets and resources that the WSRP Consumer serves as a proxy, if the cookie domain and path match.

Cookies that the client sends to the Consumer:
A client can send cookies to the WSRP Consumer. You can customize the Consumer to forward these cookies to the WSRP Producer or to other resources that the WSRP Consumer serves as a proxy. By default, the Consumer does not forward any cookies that the client sends to the WSRP Consumer.

Related tasks:

"Customizing Client Cookie Forwarding" on page 1492
A client can send cookies to the WSRP Consumer as part of an HTTP request. You can customize the WSRP Consumer to forward specific client cookies to the Producer ports or to other resources that are served by the WSRP Consumer as a proxy.

"Customizing the WSRP resource proxy" on page 1486
The WSRP resource proxy is used by the WSRP Consumer to load resources that are referenced by remote portlets.

WSRP two-phase rendering

The WSRP Consumer and the WSRP Producer in the portal support two-phase rendering for JSR 286 portlets. Two-phase rendering allows a remote portlet to set headers and cookies and to modify the HTML head section.

CF05 The Consumer and Producer determine whether a JSR 286 portlet requires two-phase rendering. The following description explains how the Consumer and Producer process a portlet that supports two-phase rendering:

CF05 By default, the Consumer and Producer process both render phases during one WSRP *getMarkup* request. The default process is advantageous since the two-phase rendering requires only one WSRP request/response roundtrip. Here the Consumer sends a WSRP *getMarkup* request only during the render headers phase and uses the WSRP response for processing both the render headers phase and the render markup phase. The Producer starts the portlet's render method both in the render headers and the render markup phases.

CF05 Starting with Portal CF 05, you can configure WSRP to use separate WSRP requests for processing the render headers and the render markup phases. This configuration enables the use of portlets, which depend on increased separation between the render phases over WSRP. Here the Consumer sends separate WSRP *getMarkup* requests during the render headers phase and the render markup phase and uses the corresponding WSRP response during each phase.

To configure WSRP two phase rendering, configure the following configuration parameters on the Consumer:

wsrp.requiresSeparateRenderPhases=(false,true)
Use this parameter to define whether WSRP uses separate WSRP requests to process the render headers and the render markup phases. The default for this parameter is *false*. This default setting means that WSRP uses one WSRP request to process both render phases, if no value is specified for this parameter.

You can set this parameter specifically for a remote portlet. To do so, set this parameter as a preference in the portlet definition of the remote portlet on the Consumer.

Alternatively, you can set this parameter for all remote portlets on the Consumer. To do so, set this parameter in the portal WP ConfigurationService by using the WebSphere Integrated Solutions Console.

If you set the parameter as a preference for both a remote portlet and the WP Configuration Service, the value that is defined in the preference of the remote portlet takes precedence.

Note: Enabling this parameter automatically disables WSRP markup caching.

Related tasks:

"Using two-phase rendering with JSR 286 portlets" on page 2935
For portlets conforming to JSR 286, IBM WebSphere Portal Express includes support for two-phase rendering, which allows portlets to set cookies and the HTTP headers and to change the portal page title dynamically.

Handling HTTP headers

You can configure both the WSRP Producer and the Consumer to ignore headers.

To do so, use the following portal service configuration parameter in the portal Configuration Service:

wsrp.ignore.headers = (comma-separated list of headers)
Use this parameter to specify one or more headers that you want to be ignored by WSRP. The Producer portal does not pass any headers that are contained in this list to the Consumer. The Consumer portal does not pass any headers that are contained in this list to the Producer. The Consumer portal also ignores headers that are contained in this list if it receives them from the Producer portal. By default this parameter is not set. You can set this parameter in the portal WP Configuration Service by using the WebSphere Integrated Solutions Console. for details about how to do so, read Setting service configuration properties.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Hints and tips for using WSRP with the portal

Here are some hints and tips for using WSRP with IBM WebSphere Portal Express.

Consuming remote portlets with your portal

If you use WebSphere Portal Express to consume remote portlets, you must set the following JVM property for the portal JVMs:

com.ibm.ws.websvcs.useMultipleSetCookie = true
Use this property to enable the WSRP Consumer to process multiple Set-Cookie headers that are contained in a WSRP response.

This property was introduced with IBM WebSphere Application Server APAR PM91361. The APAR is contained in WebSphere Application Server Version 8.5.5.1 and later.

To set this JVM property, use the WebSphere Integrated Solutions Console. After you set this JVM property, restart your Portal server.

Consuming remote portlets from IBM WSRP Producer for WebSphere Application Server

If you use WebSphere Portal Express Version 8.5 to consume remote portlets from an IBM WSRP Producer for WebSphere Application Server that you own, use the July 2015 update of that Producer. If your Producer is an earlier version, update the Producer to the July 2015 update. You can obtain the IBM WSRP Producer for WebSphere Application Server from the IBM Collaboration Solutions Catalog.

Consuming remote portlets from an earlier version IBM WSRP Producer for WebSphere Application Server might not work properly under individual environments and scenarios.

Registration

The WSRP Producer for IBM WebSphere Portal Express does not support the WSRP Registration interface.

Remote portlets on unauthenticated pages

If you add remote portlets to unauthenticated pages that have public sessions turned off, you get the following two consequences:

1. Session data is lost for each request.
2. An extra request to the Producer is submitted to establish a session.

If you add remote portlets to unauthenticated pages, turn on public sessions. This way, you can benefit portal performance and avoid unexpected behavior that results from the lost session data.

Rendering URLs for forms

Submitting data to a portlet through forms is semantically an `action` request, as this request changes the state of the portlet. WSRP strongly enforces the separation of `action` and `render` requests according to the corresponding semantics. It prevents the submission of form data through `render` requests. As a result, portlets that use `render` URLs to submit form data do not work remotely.

Portlets cannot use portal internals

With WSRP, you cannot use portal internals in portlets, such as engine objects or engine tags. Portlets that use such internals do not work remotely as WSRP does not supply the infrastructure that is required for portlets to use portal internals.

Compatibility of portlets with WSRP

The following restrictions apply to the Compatibility of portlets with WSRP:

Some of the portlets that are included with WebSphere Portal Express cannot be provided as WSRP services.

The reason is that some additional and more advanced WebSphere Portal Express concepts and features are not reflected by the current WSRP standard yet. This group includes all portal administration portlets, the Portal Search portlets Manage Search and Search Center, and some other portlets that are provided with the portal.

If portlets contain URLs to other resources, the URLs must be encoded according to the Java portlet specification to work with WSRP.

You might have portlets that serve or link to resources such as images, CSS files, JavaScript files, or servlets that are packaged with the portlet. To work in a distributed environment such as WSRP, these portlets must handle the URLs correctly. The WebSphere Portal Express WSRP Producer runtime hooks into the URL generation code that is used by the Java Portlet Specification APIs. In such cases, the portal can generate WSRP-compliant URLs to allow resources to be proxied by the WRSP Consumer server. Therefore, URLs in the browser can point to a resource

proxy that the WSRP Consumer provides and that routes the request to the appropriate resource that the WSRP Producer host provides. For example, portlets might contain URLs that include CSS or JavaScript files, and you want to provide these portlets by WSRP. In such a case, you must make sure that the URLs point to the correct locations by encoding them in compliance with the Java portlet specification. If you do not encode the URLs by using the JSR API calls, the portlets might not work properly.

Note: The WebSphere Portal Express resource proxy implementation also supports the serving of relative URLs. Example: A URL points to a CSS file and is encoded by using the Java Portlet Specification API. The CSS file in turn contains relative links to further resources such as images. In such a case, requests for those images are also routed and served through the WSRP Consumer resource proxy.

WSRP does not support Consumer-side configuration of remote portlets that do not support shared configuration

The configuration of the `edit_defaults_compatibility` portlet mode is not supported for portlets that are consumed by using WSRP.

The PUMA SPI cannot be used with WSRP

The PUMA SPI does not allow use with remote portlets.

Tag Cloud, Tag Center, and Results List portlets do not support WSRP

The Tag Cloud and Tag Center portlets, including the Result List portlet, do not support WSRP. Therefore, a Producer portal cannot provide these portlets as remote web services, or for a Consumer portal to consume them so that its users can use them.

Related information:

IBM WSRP 2.0 Producer for WebSphere Application Server

IBM WSRP 2.0 Producer for WebSphere Application Server documentation

Troubleshooting WSRP

You can troubleshoot WSRP by using different methods such as logging and tracing, debugging, and monitoring.

- "Setting traces and using the portal run time log file for WSRP diagnosis"
- "Debugging and monitoring the WSRP protocol flow" on page 1505
- "Monitoring WSRP messages between the Consumer and Producer by using TCPMon" on page 1505

Setting traces and using the portal run time log file for WSRP diagnosis

You can diagnose problems that might occur during the use of WSRP. To do so, you set WSRP-specific traces and enable run time logs for the Consumer, Producer, and administration components of the WSRP implementation. Use the administration portlet Enable Tracing.

You can enable the following trace loggers for the WSRP implementation:

Table 191. WSRP trace loggers

Component	Trace string
Administration	com.ibm.wps.command.wsrp.*=all com.ibm.wps.wsrp.cmd.*=allcom.ibm.wps.wsrp.common.*=all com.ibm.ws.websvcs.trace.MessageTrace=all
Consumer	com.ibm.wps.wsrp.consumer.*=all com.ibm.wps.wsrp.common.*=all com.ibm.ws.websvcs.trace.MessageTrace=all
Producer	com.ibm.wps.wsrp.producer.*=all com.ibm.wps.wsrp.common.*=all com.ibm.ws.websvcs.trace.MessageTrace=all
XMLAccess	com.ibm.wps.command.xml.*=all com.ibm.wps.wsrp.common.*=all com.ibm.ws.websvcs.trace.MessageTrace=all

Debugging and monitoring the WSRP protocol flow

You can trace SOAP messages that are exchanged between a WSRP Consumer and a WSRP Producer. To do so, you use a monitor or sniffer application to capture network traffic between the two points. You can use the utility **TCPMon** that is shipped with the WebSphere Application Server. As an alternative, most Linux operating systems provide suitable utilities, for example, **tcpdump**. You can run **tcpdump** on either side and record network traffic that is captured by a network interface. There are also free tools available.

Monitoring WSRP messages between the Consumer and Producer by using TCPMon

You can use the TCPMon tool to monitor the WSRP messages between the Consumer and Producer. For information about how to do so, read the WebSphere Application Server Information Center topic about Tracing web services messages. The TCPMon application uses the man-in-the-middle approach. TCPMon listens on a TCP port, logs the HTTP or SOAP traffic, and forwards the request to the designated server and TCP port. Therefore, you must redirect the communication from the Consumer to the TCPMon application and have TCPMon forward the request to the WSRP Producer.

If you already have a Producer configuration on your Consumer portal, you can modify the host and port for each WSRP end point address URL that you want to monitor. You can change the WSRP endpoint address URLs that the Consumer uses to communicate with the integrated Producer. To do so, you can use either the portal administration user interface or the XML configuration interface.

Alternatively, to debug a certain scenario, you can also create a new Producer definition on the Consumer with a WSDL that contains WSRP endpoint address URLs that point to the TCPMon host and port.

In WebSphere Portal Express, you can manipulate the WSDL contents by adding URL parameters to the WSDL URL.

After you redirect the traffic, configure the TCPMon tool to listen on the port that you specified on the Consumer side to communicate with the Producer portal. Also, set the target port to the actual port values of the Producer WSRP interfaces.

For the WebSphere Portal Express Producer, these target ports are the ports that the WSDL file contains when you request the WSDL file without the `port` parameter.

To run the TCPMonitor tool, follow the instructions that are given in the WebSphere Application Server information center under *Tracing SOAP messages with tcpmon*.

Related information:

Tracing SOAP messages with tcpmon

Browser behavior and scenarios

Browser behavior for the back button, bookmarks and history affect user interaction and cause unexpected results.

"Back button behavior"
Learn how the behavior of the web browser Back button can affect portal navigation.

"Back button limitations" on page 1512
The following restrictions apply to backward navigability of portal pages.

"Configuring history expiration limits" on page 1513
You can configure how far back the WebSphere Portal Express tracks the history of the navigational state of pages. This allows you to control the balance between the performance of your portal and the retrievability of previously visited pages for users.

Back button behavior

Learn how the behavior of the web browser Back button can affect portal navigation.

With IBM WebSphere Portal Express users can use the Back button of the browser to navigate back through the recent history of the **views** of the pages that they visited.

When a user navigates backwards by using the browser Back button, the portal restores the views of pages that the user visited recently. This behavior of the portal might affect the following aspects of the pages:

- The sequence by which the user navigated through the portal and selected the pages.
- The expanded or collapsed state of the tree hierarchy that the navigation shows.
- The lateral scrolling position of the register tabs that users can use to switch between pages.
- The information of the user's default selection for a label. For example, this can be the information which page is selected for a label by default for that user. If the user selects a different page from that label, the portal displays that page as the default page for that label from then on.
- The portlet window information that distinguishes between different instances of the same portlet on different pages.
- Modifications of the view or window state of a portlet, such as the minimized, maximized, or normal or default state. For example, if a user views a portlet in its default view state, and then maximizes the portlet, clicking the Back button returns the portlet to its previous state, that is the portlet is displayed with its default size.

- The portlet mode selection options, such as View, Personalize, Edit Shared Settings, or Configure.
- The information as to whether a portlet is displayed in solo state, and the information which portlet is displayed in solo state.
- The information as to whether the link for toggling between **Show layout tools** and **Hide layout tools** is displayed. For example these tools include options for moving portlets on a page.
- If the portlet complies with the standard portlet API, the view state of the portlet is preserved as it is determined by the render parameters of the portlet. For detail about state changes to standard API compliant portlets, refer to Using the Back button with standard API portlets.

Terminology: The following terminology conventions are used for this topic:

1. The terms **view** state and **navigational** state are used synonymously here. **`Render parameters`** in the sense of the standard portlet API also refers to the same.
2. Everything that is said about the browser Back button here applies to the browser Forward button respectively.
3. The browser Back button does not have an undo function. It interacts with the view or navigational state only, not with the application state.

User experience

Users can click the Back button of the browser as many times as the browser history mechanism allows. In such cases the markup comes from the history cache of the browser and is served without an additional request to the server. Users see exactly the same markup as they saw when they visited the page previously.

When users use the Back button to return to a page, they can use any link on that previously visited page to navigate further as follows:

- If the link is only a render link and does not result in a portlet action, then no special case applies.
- If the link is an action link and includes a portlet action, the behavior differs, depending on whether the user had already clicked the same action link during a previous visit to the page in the same session:
  - If the user has not clicked the action link previously, the portal performs the action.
  - If the user has already clicked the action link previously, then the portal displays the markup of the state that resulted from the last invocation of the action link. In other words, the portal does not perform the same action twice.

From a user point of view the consequences of considering these parameters as view state are as follows:

- Each different combination of states is part of the address of the page, that is its URL, and can be bookmarked. Users can set separate bookmarks for different states of the same page.
- Users can use the Back button to navigate through modifications of the view states. For example, if a user maximized a portlet, the user can use the browser Back button to switch back to the previous window state of this portlet.
- The effect of the Back button on interactions with individual portlets strongly depends on the implementation of the portlet. If the portlet implements its links as render links by changing render parameters on a per-link basis, then the Back

button can be used to navigate through the history of the interactions with the portlet. This however is only possible for portlets written against the standard portlet API.

Example scenarios

Example scenario 1: Maximizing or minimizing portlets

1. The user views a page A with two portlets A1 and A2, both in their normal default window state.
2. The user maximizes portlet A1.
3. The user clicks the Back button. Portlet A1 is displayed in normal state again.

Example scenario 2: Switching between pages

1. The user views a page A with two portlets A1 and A2, both in their normal default window state.
2. The user maximizes portlet A1.
3. The user selects page B. The portal displays page B.
4. The user clicks the Back button. The portal returns to page A. Portlet A1 is displayed in maximized state.
5. The user clicks the Back button once more. Portlet A1 is displayed in normal state again.

Example scenario 3: Different user action leads to same result.

Initial scenario and user actions:

1. The user views a page A with a number of portlets on this page.
2. The user interacts with the portlets.
3. The user puts portlet A1 into edit mode.
4. The user clicks A2 into minimized state.
5. The user makes the document viewer portlet A3, which is a standard API portlet, display page 2 of the document that it shows.
6. Now the user switches to page B with portlet B1. Portlet B1 is displayed in normal state, view mode and in its default application state.
7. The user minimizes portlet B1.

Despite different subsequent user actions, the following sub-scenarios have the same final outcome:

Scenario 3a: Using the Back button

1. The user clicks the Back button. The portal displays portlet B1 in normal state.
2. The user clicks the Back button once more. The portal changes to page A. It displays portlet A1 in edit mode and portlet A2 in minimized state. Portlet A3 shows page 2 of the document.

Scenario 3b: Making a new selection

1. The user clicks page A *in the navigation* to "return" to page A. Conceptually this change is not considered a change *back* to page A, but *forward* to page A. As a result, the portal displays page A. It shows portlet A1 in edit mode and portlet A2 in minimized state. Portlet A3 shows page 2 of the document.

Setting bookmarks

All state information except the portlet application state is bookmarked, for example render parameters, window states, and modes. The invocation of a bookmark resets the application state to the default state as defined by that bookmark.

Note: The user can set several independent bookmarks for different windows of the same page. Therefore the result of the previous example scenario is independent of other bookmarks that the user might have set on different window states of the same page.

Example scenario: Setting bookmarks

1. The user selects a page A with portlets A1 and A2.
2. The user maximizes portlet A1 and minimizes portlet A2.
3. The user sets a bookmark on the page. The user names the bookmark `A1min_A2max`.
4. The user logs out and logs back in.
5. The user selects the bookmark A1min_A2max. The portal displays page A with portlet A1 maximized and portlet A2 minimized.

The user can set another separate bookmark `A1Edit_A2Default` on the same page A with portlet A1 in Edit mode and portlet A2 in its default state. Both bookmark will work independently of each other.

Considerations for administrators about Back button behavior

The Back button behavior of the portal is internally achieved by coding the combination of all states into the address of the view, that is into its URL. Thus each different combination of navigational states results in a different URL. This has the following additional advantages:

- Users can set separate bookmarks for different states of the same page.
- You can have pages cached by configuring specific requirements.

However, make no assumptions about the syntax or structure of portal URLs. For example, you cannot create valid URLs by simple concatenation. This will automatically be true if only the public API is used to create URLs.

Using the Back button with standard API portlets

If a portlet complies to the standard portlet API, users can use the Back button to navigate backwards through the view states of that portlet as determined by the render parameters of the portlet. When the user clicks the Back button after working with that type of portlet, all information defined by the render parameters is reversed. There is no general rule as to which information is stored in cache as render parameters for a portlet. The effect of the Back button on interactions with individual portlets depends on the implementation of the portlet. This is determined by the person who developed the portlet.

If the portlet implements its links as render links by changing render parameters on a per-link basis, then users can use the Back button to navigate through the history of the interactions with the portlet. However, this is only possible for portlets written against the standard portlet API.

As a general rule, all information that influences the *view* of the portlet rather than its application state should be implemented as render parameters. For more details about how to develop standard API compliant portlets for WebSphere Portal Express, refer to Best practices: Developing portlets using JSR 168 and WebSphere Portal Express.

Configuring the history expiration limit for portal pages

You can configure the portal so that it discards the render parameters for pages that the user has not visited recently within the same session. The purpose of this setting is to limit the URL length which might benefit your portal performance. The portal discards the navigational state of the portlet application of standard API portlets on pages that are too far back in the history.

You can specify the number of different page visits after which the history mechanism can discard the render parameters of the portlet. Your setting determines how far backwards users can at least navigate in the recent history of portal pages that they visited. The number that you specify defines the minimum number of different pages selected by the user after which the portal can discard the render parameters of a page. (The decision whether the render parameters of the page are actually discarded depends on the expiration policy of the internal cache that stores the render parameters of those pages.) If the user returns to a page after visiting the specified number of other pages *and* if the render parameters of that page have expired, the portal displays that page in its default state.

You configure this expiration limit by setting the value for the property `keymanager.lru.size= (integer)`. You set this property in the `StateManagerService`.

You can specify by which circumstances the render parameters of a page are stored or discarded:

1
: Each time that the user selects a different page, the render parameters of the portlets on the previously selected page can be discarded.

A positive integer
: Specify the required number of pages. The render parameters of a given page can be discarded after the user has visited that number of other pages.

0
: Render parameters are always stored in the portal session memory and never discarded.

Note: Do not specify a value smaller than zero (**0**). Negative values are considered to be not valid.

Example scenario: Configuring a limit to the history of viewed pages

1. The configuration setting is set to a history value of 2.
2. The portal pages A, B, and C contain separate instances of the standard portlet API compliant document viewer portlet A1, B1, and C1. Each of the portlets display page 1 of the same document on each page A, B, and C. This is the default for the portlet.
3. The user selects portal page A and navigates to page 2 of the document in portlet A1.

4. The user selects portal page B and navigates to page 3 of the document in portlet B1.
5. The user selects portal page C and navigates to page 4 of the document in portlet C1.
6. The user selects portal page B. Portlet B1 displays page 3 of the document as before.
7. The user selects portal page C. Portlet C1 displays page 4 of the document as before.
8. The user selects portal page A. Portlet A1 displays page 1 of the document because this is the default state for this portlet. The previous state has been discarded because it is back in the view history by 3 pages already, and thereby exceeds the configuration setting of only 2 pages.
9. The user selects page C. Portlet C1 still displays page 4 as before as it is within the configured history value of two stored pages.

Configuring the history setting might result in an inconsistent user experience, depending on the browser cache. If the user uses the Back button to navigate backwards rather than use explicit navigation, the user experience can be inconsistent if the user exceeds the configured view history size on the server. In this case the browser might display the portlets in their previous application state rather than the default state, because the markup is served from the client browser cache without a request to the server. However, if the view history has been exceeded, the server resets the portlet state to its default. Refreshing such a page displays the affected portlet in its default state rather than in the state that was displayed before the refresh operation. If you do not want to accept this behavior, set the size of the view history to the value `0` . This means that the portlet application state never expires.

View history limit influences only the navigational state of the application

The configured limit to the view history influences only the navigational state of portlet applications. It does not influence any of the following:

- The **portal** view or navigational state.
- The **session** state of a portlet application, that is any actions or transactions performed with the portlet.

Example scenario: View history limit influences only the navigational state of the application

1. The configuration setting is set to a history value of 3.
2. The user navigates to a shopping site.
3. The user navigates through several views of pages and looks at various products.
4. On a page X the user picks an item and puts it in the virtual shopping cart.
5. The user navigates through four more page views and looks at more products.
6. The user clicks the browser Back button four times to navigate backward to where the user placed the item in the shopping cart. The portal displays page X in its default state and not in the state in which it was when the user navigated to the next page. The reason is that the page view is back in the history by 4 navigational steps already and thereby exceeds the configuration setting of 3 views. However, the product that the user wants to purchase is not removed from the shopping cart as that information is part of the session state of the application.

Related concepts:

"Caching" on page 267
Caching affects the performance of your IBM WebSphere Portal Express environment. Learn about some simple ways to improve the caching performance. After you have reviewed this content, you should also review the WebSphere Portal Express and Web Content Manager Performance Tuning Guide which provides more information about caches for both WebSphere Portal Express and Web Content Manager.

Related information:

http://www.ibm.com/websphere/portal/library

http://jcp.org/en/jsr/detail?id=168

http://www.ibm.com/developerworks/websphere/library/techarticles/0403_hepper/0403_hepper.html

Back button limitations

The following restrictions apply to backward navigability of portal pages.

The administration portlets do not officially support the browser 'Back' or 'Refresh' functions.

For a listing and description of the administration portlets, refer to "Portal administration portlets" on page 1047.

The Back and Forward buttons have no undo or redo function

The browser Back and Forward buttons do not have an undo or redo function for actions or transactions performed by users. The portal marks completed actions by an internal identifier. If a user performs an action in a portlet and then navigates back to the same portlet panel by the Back or Forward buttons of the browser, the action is not performed once more.

For example, if a user initiates a money transfer or payment of a bill and then navigates further, clicking the Back and Forward buttons of the browser does not repeat the money transfer once again. An action is only performed once.

If a user wants to perform the same action again, the user can do so after refreshing the page. That portal marks that action with a different internal identifier. The protection against repeating the same action applies only to actions within the same markup fragment.

URL length limitation

In a browser, the possible length of a URL is limited to 2048 characters. This depends on the browser and its implementation. Reverse proxies usually have a length limit of 1024 characters for URLs. To circumvent this limit, you can keep the URLs shorter by configuring an expiration limit for pages that are too far back in the navigation history.

URL length limit on small devices

Small devices have a more severe length limit for URLs than browsers and proxies. For such small devices the complete view state is kept in the session on the server

and is referenced by an ID in the URL. Therefore users cannot bookmark pages with their view state. However, they can use the Back button the same way as described previously to navigate back to the same state of a page.

Structure of portal URLs

Do not make any assumptions about the syntax or structure of portal URLs. For example, you cannot create valid URLs by simple concatenation. This will automatically be true if only the public API is used to create URLs.

Bookmarks from previous versions of WebSphere Portal Express do not work

Browser bookmarks to pages from WebSphere Portal Express Version 7.0 do not work for Version 8.5. Users need to create their bookmarks new.

Server side bookmarks of the WebSphere Portal Express Favorites portlet are migrated automatically as you migrate to WebSphere Portal Express Version 8.5.

Related concepts:

"Portal administration portlets" on page 1047
Administration portlets are supplied with IBM WebSphere Portal Express. Use them to perform administration tasks and actions on portal resources, give other users limited access rights on selected resources, and deploy custom portlets, themes, or skins.

Related information:

http://www.ibm.com/websphere/portal/library

Configuring history expiration limits

You can configure how far back the WebSphere Portal Express tracks the history of the navigational state of pages. This allows you to control the balance between the performance of your portal and the retrievability of previously visited pages for users.

About this task

The further back your portal tracks the history of portal pages, the further back your users can go in their browsing history of pages that they visited before. However, this also increases the URL length of your portal pages, as the navigational history is stored in the page URL. Therefore limiting the page history might be of benefit for the performance of your portal.

WebSphere Portal Express provides two different approaches for controlling the page history in the navigational state:

1. The **history manager** allows you to control the number of previously visited pages for which the portal tracks the navigational state.
2. The history expiration limit for swapped render parameters

   "History manager for pages" on page 1514
   The history manager allows you to control for how many visited pages navigational state you want to tracked. In other words it controls the maximum number of pages whose state is contained in the portal URLs. The visited pages are tracked within the navigational state in a LRU algorithm based way.

   "History expiration limit for render parameters" on page 1515
   You can configure the portal so that it discards the render parameters for pages

that the user has not visited recently within the same session. The purpose of this setting is to limit the URL length. This might be of benefit for the performance of your portal. The portal discards the navigational state of the portlet application of standard API portlets on pages that are too far back in the history.

History manager for pages

The history manager allows you to control for how many visited pages navigational state you want to tracked. In other words it controls the maximum number of pages whose state is contained in the portal URLs. The visited pages are tracked within the navigational state in a LRU algorithm based way.

The history manager provides a configurable threshold to define the maximum number of pages that are tracked. If the number of different tracked pages exceeds that threshold, the history manager discards the page that was visited the longest time ago is dropped out. With this page the portal also drops the state of the portlets on this page. Furthermore, the history manager this functionality allows you to define strategies for how to proceed with public render parameters. Limiting the navigational state to a maximum number of pages can reduce the length of the portal URLs.

Notes:

1. The history manager applies to both anonymous and authenticated users, in other words it works on public and protected pages.
2. The history manager is independent of the history expiration limit for swapped render parameters, which determines how long render parameters are kept for swapping into the session.

Configuration of the history manager

The history manager provides the configuration settings listed in the following. You configure them in the WP State Manager Service in the WebSphere Integrated Solutions Console. For details about this service and how to configure these settings see the topics about Setting service configuration properties and the State Manager Service.

historymanager.enabled = (true)
: This parameter allows you to disable (`false`) or enable (`true`) the history manager. The default value is `true`, that is the history manager is enabled.

historymanager.threshold = (10)
: This parameter allows you to configure the maximum number of different pages for which the navigational state is tracked. Set this parameter to a positive integer value. The default value is `10` .

historymanager.prp.removalstrategy = [no_removal | wcm_id | explicit_bucket_assignment]
: This parameter allows you to specify a strategy that defines how to proceed with public render parameters if the history manager removes the state of a page and the portlets on this page. As public render parameters might be used by portlets on different pages, the removal of public render parameters needs special handling. The possible values have the following meaning:

no_removal
Public render parameters are not removed. This means that only the portlet specific state is removed, for example private render parameters.

wcm_id
Public render parameters are removed if the expired page has an explicit shared state bucket assigned that starts with the String `ibm.wcm.` .

explicit_bucket_assignment
Public render parameters are removed if the expired page has an explicit shared state bucket assigned, regardless of a prefix. This is a more general strategy than `wcm_id` . This is the default value.

Example scenarios

The following examples can help explain the history manager functionality. For all examples the maximum configured number of pages that are tracked in the navigational state is set to `3` .

Example 1:
A user visits portal pages in the following order: P1, P2, P3, P4. When the user navigates to page P4, the history manager removes the navigational state of page P1 and the states of all portlets on that page from the state.

Example 2:
A user visits portal pages in the following order: P1, P1, P2, P3, P2, P3, P1. The history manager removes no state.

Example 3:
A user visits portal pages in the following order: P1, P2, P3, P1, P4. When the user navigates to page P4, the history manager removes the navigational state of page P2 and the states of all portlets on that page.

History expiration limit for render parameters

You can configure the portal so that it discards the render parameters for pages that the user has not visited recently within the same session. The purpose of this setting is to limit the URL length. This might be of benefit for the performance of your portal. The portal discards the navigational state of the portlet application of standard API portlets on pages that are too far back in the history.

You can specify the number of different page visits after which the history mechanism can discard the render parameters of the portlet. Your setting determines how far backwards users can at least navigate in the recent history of portal pages that they visited. The number that you specify defines the minimum number of different pages selected by the user after which the portal can discard the render parameters of a page. The decision whether the render parameters of the page are actually discarded depends on the expiration policy of the internal cache that stores the render parameters of those pages. If the user returns to a page after visiting the specified number of other pages *and* if the render parameters of that page have expired, the portal displays that page in its default state.

You configure the expiration limit for render parameters by setting the following property to an integer in the WP State Manager Service in the WebSphere Integrated Solutions Console. For details about this service and how to configure these settings see the topics about Setting service configuration properties and the State Manager Service. You can specify by which circumstances the render parameters of a page are stored or discarded.

keymanager.lru.size

Specify one of the following values:

1 Each time that the user selects a different page, the render parameters of the portlets on the previously selected page can be discarded.

A positive integer

Specify the required number of pages. The render parameters of a given page can be discarded after the user has visited that number of other pages.

0 Render parameters are always stored in the portal session memory and never discarded.

Note: Do not specify a value less than zero (**0**). Negative values are considered to be not valid.

Example scenario: Configuring a limit to the history of viewed pages

1. The configuration setting is set to a history value of 2.
2. The portal pages A, B, and C contain separate instances of the standard portlet API compliant document viewer portlet A1, B1, and C1. Each of the portlets display page 1 of the same document on each page A, B, and C. This is the default for the portlet.
3. The user selects portal page A and navigates to page 2 of the document in portlet A1.
4. The user selects portal page B and navigates to page 3 of the document in portlet B1.
5. The user selects portal page C and navigates to page 4 of the document in portlet C1.
6. The user selects portal page B. Portlet B1 displays page 3 of the document as before.
7. The user selects portal page C. Portlet C1 displays page 4 of the document as before.
8. The user selects portal page A. Portlet A1 displays page 1 of the document because this is the default state for this portlet. The previous state has been discarded because it is back in the view history by 3 pages already, and thereby exceeds the configuration setting of only 2 pages.
9. The user selects page C. Portlet C1 still displays page 4 as before as it is within the configured history value of two stored pages.

Chapter 12. Securing

Security tasks include setting up property extension databases and custom user repositories, configuring and activating SSL, and configuring authentication. In addition, tasks such as activating Federal Information Processing Standards (FIPS) and NIST SP800-131a security modules and configuring external security managers such as Security Access Manager might be required to secure your portal environment.

If you have not configured your user registry yet, go to the Installing section, select the appropriate operating system, and then choose the appropriate deployment scenario. Then, select the Configuring portal to use a user registry topic.

"Configuring SSL" on page 1595
Secure socket layers (SSL) encrypt traffic between the client browser and the server to secure information exchanged over the network between the browser and IBM WebSphere Portal Express. You will need to configure your environment for SSL to activate this additional security feature.

"Enabling FIPS and (NIST) SP800-131a" on page 1608
IBM WebSphere Portal Express tolerates IBM WebSphere Application Server support of Federal Information Processing Standards (FIPS) and National Institute of Standards and Technology (NIST) SP800-131a. You can configure WebSphere Application Server to activate FIPS 140-2 compliant security modules. When you enable FIPS, you can use only FIPS to securely encrypt data. For this reason, you must also configure FIPS for systems that require secure transactions, which can include HTTP servers and LDAP servers.

"Configuring Session Security Integration" on page 1609
IBM WebSphere Application Server protects your session from access by other users.

"Enabling HTTP Basic Authentication for simple clients" on page 1610
IBM WebSphere Portal Express provides an HTTP Basic Authentication Trust Association Interceptor that can be enabled to allow specific clients to log into the portal by using HTTP Basic Authentication instead of HTTP Form Based Authentication.

"Setting up custom user repositories" on page 1615
A custom user repository is any repository that WebSphere Portal Express does not support out-of-box. However, you can configure WebSphere Portal Express to support any type of repository in a federated or stand-alone user registry, whether an LDAP directory, database, file system, and so on. Setting up custom user repositories involves tasks such as defining additional repositories to the default federated user registry, creating a custom stand-alone user repository, and updating your user repository to reflect changes in your environment. Learn what steps are required to create and update custom user repositories and what specific interfaces you must implement to enable communication between WebSphere Portal Express and a repository.

"External security managers" on page 1619
Use external security managers such as IBM Security Access Manager to perform authentication and authorization for IBM WebSphere Portal Express. You can use an external security manager for authentication only or for both authentication and authorization. Using an external security manager to perform only authorization is not supported at this time.

"Deleting passwords from properties files" on page 1669
The configuration tasks might require you to write security-sensitive information, such as passwords, into multiple properties files. When you no longer need this security-sensitive information for your configuration, you should remove them and move the files to a safe place or set the file permissions so that only authorized users can read them.

"Updating user ID and passwords" on page 1670
IBM WebSphere Portal Express and IBM WebSphere Application Server use some accounts from the registry (for example, the LDAP server) including administrative and bind IDs for authenticated access to databases and LDAP severs respectively, as well as the WebSphere Portal Express and WebSphere Application Server administrative IDs. Often this means that the account passwords are stored in the WebSphere Portal Express and WebSphere Application Server bootstraps configuration files, which allows the authentication process to work.

Related information:

Security and authentication considerations

Security and authentication are key elements of a production environment. Learn about single sign-on, credential vaults and external security managers.

Related information:

WebSphere Application Server V7.0 Security Guide

Authentication

Authentication requires users to identify themselves to gain access to a system or resources. The combination of a user ID and a password is the most common method of authentication. Users can identify themselves immediately upon entry to the system or the system can prompt users to identify themselves before accessing protected resources. After users successfully authenticate, the system identifies which resources-specific users have sufficient authorization to access.

Note: You can have simultaneous, multiple log ins with the same user ID and password. However, this method can result in a non-reliable behavior depending on the client or authentication method. For this reason, IBM WebSphere Portal Express does not support simultaneous, multiple log ins.

WebSphere Portal Express supports the following methods for login and authentication:

Form-based authentication
WebSphere Portal Express uses the IBM WebSphere Application Server Custom Form-based Authentication mechanism to prompt for identities. Users type their user ID and password in the Login portlet.

SSL client certificate authentication
You can configure authentication with certificates that are stored in the browser or on a smart card. The certificates are stored through a Secure Sockets Layer (SSL) client certificate authentication. The authentication is done for the users when they access the protected area of the portal.

Third-party authentication
You can also configure third-party authentication. An external security manager, such as IBM Security Access Manager, is an example. With this method the portal trusts that the authentication was done by the third-party product.

Federal Information Processing Standards and and (NIST) SP800-131a

Federal Information Processing Standards (FIPS) and NIST SP800-131a are standards and guidelines issued by the United States National Institute of Standards and Technology (NIST) for federal government computer systems. FIPS and SP800-131a are developed when there are compelling federal government requirements for standards, such as for security and interoperability, but acceptable industry standards or solutions do not exist.

WebSphere Portal Express tolerates WebSphere Application Server's support of FIPS 140-2 and SP800-131a. WebSphere Application Server integrates cryptographic modules such as Java Secure Socket Extension (JSSE) and Java Cryptography Extension (JCE), which are FIPS 140-2 certified. Throughout the documentation and the product, the FIPS 140-2 certified IBM JSSE and JCE modules are called as IBMJSSEFIPS and IBMJCEFIPS, which distinguishes the FIPS-certified modules from the prior, non-certified IBM JSSE and IBM JCE modules.

The FIPS 140-2 compliant toleration means that WebSphere Portal Express will continue to work after WebSphere Application Server is configured to activate FIPS 140-2 compliant security modules. The WebSphere Portal Express product has no self-contained Cryptographic Support and as a result is unaware of the module differences. Functions in WebSphere Portal Express that use encryption include:

- Secure Sockets Layer (SSL) connections inbound from clients (but this is basically the WebSphere Application Server and HTTP Server support for SSL connections, and is not apparent to WebSphere Portal Express)
- Internal connections between WebSphere Portal Express administrative functions and WebSphere Application Server administrative services (started, for example, when deploying a portlet which must create a web application in WebSphere Application Server)

All connections listed in this document are carried over SSL using FIPS-compliant encryption. Without FIPS 140-2 support, connections might not be encrypted.

Limitations

There are some restrictions in the level of support that WebSphere Portal Express provides in using FIPS-certified modules:

- By default, Microsoft Internet Explorer might not have TLS enabled. To enable TLS, open the **Internet Explorer browser** and click **Tools** > **Internet Options**. On the **Advanced** tab, select the **Use TLS 1.0** check box.
- The IBM Tivoli Directory Server provides the Use FIPS certified implementation option, which enables the directory server the FIPS-certified encryption algorithms uses. For information, see "Setting the level of encryption" within the IBM Tivoli Directory Server Administration Guide.
- You can use FIPS-certified JSSE providers if your servers and clients are using WebSphere Application Server.
- If you are trying to enable FIPS processing mode with GSKit bundled with ITDS, see technote 1578181.

Related tasks:

“Enabling FIPS and (NIST) SP800-131a” on page 1608
IBM WebSphere Portal Express tolerates IBM WebSphere Application Server support of Federal Information Processing Standards (FIPS) and National Institute of Standards and Technology (NIST) SP800-131a. You can configure WebSphere Application Server to activate FIPS 140-2 compliant security modules. When you enable FIPS, you can use only FIPS to securely encrypt data. For this reason, you must also configure FIPS for systems that require secure transactions, which can include HTTP servers and LDAP servers.

Related information:

IBM Tivoli Directory Server Administration Guide

Technote 1578181

Support for NIST SP 800-131 and NAS Suite B

Planning for single sign-on

Single sign-on provides a secure method of authenticating a user one time within an environment and using that authentication (for the duration of the session) to access other applications, systems, and networks. In the context of IBM WebSphere Portal Express there are two single sign-on realms; one realm from the client to the portal and other web applications and the other realm from the portal to the backend applications.

Single sign-on for the client realm is established using the IBM WebSphere Application Server Lightweight Third Party Authentication (LTPA) token functionality or an Authentication Proxy. The LTPA token can also establish

backend single sign-on if the backend application accepts it through the Credential Vault portlet or the Java Connector architecture.

WebSphere Portal Express and Java Authentication and Authorization Services

Single sign-on uses only the authentication portion of Java Authentication and Authorization Services (JAAS). WebSphere Portal Express builds a JAAS Subject for each logged on user. The Subject consists of Principals and Credentials. A Principal is a piece of data, such as the user ID or the distinguished name that gives the Subject's identity. A Credential is a piece of data, such as a password or a CORBA Credential that can be used to authenticate a subject. The Subject carries around the Principals and Credentials that the portlet can use directly or through the credential service.

Related information:

Lightweight Third Party Authentication

Secure communications using SSL

Configuring IBM WebSphere Portal Express for SSL adds security to the client-portal exchange. It encrypts all traffic between the client browser and the server, so that no one can "eavesdrop" on the information that is exchanged over the network between the client browser and WebSphere Portal Express. In addition, assuming that the IBM WebSphere Application Server is also configured to accept or require SSL connections, the LTPA Token and other security and session information can be protected against hijack and replay attacks.

Configuring WebSphere Portal Express for SSL is a multistep process that involves configuring the following components:

- Web (HTTP) server running in front of WebSphere Application Server
- WebSphere Application Server
- WebSphere Portal Express

In general, the web server must be configured to accept inbound SSL traffic. The WebSphere Application Server plug-in for the web server must be configured to forward traffic on that port to WebSphere Application Server and WebSphere Portal Express. Then, you must configure the virtual host information. Finally, WebSphere Portal Express must be configured to generate self-referencing URLs using SSL as the transport.

Note: This procedure might be slightly different if a front-end security proxy server such as Security Access Manager WebSEAL is used. In that case, the front-end security server handles the client SSL connections. The web server receives connections from the front-end security proxy server. Mutually authenticated SSL can be configured in the web server and the front-end security proxy server if needed. It is highly dependent on the security requirements of each deployment.

Related information:

WebSphere Application Server Security Guide: Chapter 5

Credential Vault

The Credential Vault is a service that stores credentials that allow portlets to log in to applications outside the realm on behalf of the user. It manages multiple identities for portlets and users.

Using Credential Vault, a portlet can retrieve a user's authentication identity and then pass the information to a backend application. The Credential Vault features the following level of sign-on:

Passive Credentials
Passive Credentials retrieve stored secret data such as user ID and password or certificates. This option is more flexible. However, it requires portlet writers to manage their own connections and authentication to backend applications with the credentials they retrieved from the Credential Vault.

Credential objects can also pass IBM Security Access Manager or Computer Associates eTrust SiteMinder single sign-on tokens to backend applications.

IBM WebSphere Portal Express provides one simple database vault implementation for mappings to secrets for other enterprise applications. By default, the Credential Vault contains an administrator-managed vault segment and a user-managed vault segment. Administrator-managed vaults allow users to update mappings; however, users cannot add new applications to this vault. The user-managed vault segment allows users to add application definitions, such as a POP3 mail account, under the user vault and store a mapping there. By default, the vault uses an encryption plug-in that encodes the passwords in Base 64.

WebSphere Portal Express initially provides two vault adapter configurations that write to the database:

- A default vault for administrator-managed vault segments that stores credentials in the release domain: default-release
- And a default vault for user-managed vault segments that stores credentials in the customization domain: default-customization

WebSphere Portal also supports the storage and retrieval of credentials from other vault services, such as Security Access Manager. WebSphere Portal includes a Credential Vault adapter for Security Access Manager. This plug-in works on the following operating systems:

- Windows

Caching considerations

Information that is protected by access control and is therefore restricted to a limited set of people needs special consideration when served from an access control agnostic cache. These considerations especially apply to server side caches but you also need to consider local browser caches.

Browser caches usually have no issues unless the computers are shared between multiple users with different levels of access. If you access WebSphere Portal Express from a shared computer, it is important to realize that all users who have access to the computer can access content that is cached in the local browser cache. To prevent this from happening, do not enable public or private caching of the content. Caching is disabled by default.

Depending on the type of browser you are using, you can still experience information leakage from shared computers, even if caching is completely disabled, because some browsers serve content that is accessed by clicking the browser's Back button from a separate history cache that is not affected by HTTP caching directives. As a result, if you click the Back button, you may see content generated from the previous user even if the previous user performed a logout. To prevent this from happening, the markup that is rendered on logout should explicitly clear the browser's history cache, which typically requires browser-specific script coding, or display a message to close all browser windows after logout. History cache can typically be disabled in the browser but it may be activated by default.

The WebSphere Portal Performance Tuning Guide provides information about caches for WebSphere Portal Express and Web Content Manager. After you have setup your environment, review the tuning guide to learn more about stand-alone and cluster tuning and then read about both WebSphere Portal Express and Web Content Manager caches.

Essential tuning and performance resources

Controlling access

After creating users and groups, you can assign them different levels of access to specific resources, roles, and policies. This access controls what actions they can perform on various pages, portlets, and applications.

Review the following information and perform the following tasks to control access within IBM WebSphere Portal Express:

"Managing Access Control" on page 1525
Get familiar with concepts related to administering IBM WebSphere Portal Express access control. To administer access control, use the Resource Permissions portlet, the User and Group Permissions portlet, the Manage Users and Groups portlet, the XML configuration interface, or the Portal Scripting Interface.

"Resources, roles, access rights, and initial access control settings" on page 1526
In order to fine-tune the security measures of your IBM WebSphere Portal Express environment, the administrator creates resources, roles, and access rights, which allows the administrator to control who has access to various information based on their roles and the access rights to that information.

"Access control scenarios" on page 1553
These scenarios provide helpful illustrations on how access control can be set up.

"Access control" on page 1556
You can restrict access to selected users and groups to the views within an authoring portlet, the items that are managed by the authoring portlet, and to elements and pages that are displayed within a website.

"Setting user and group permissions" on page 1567
The User and Group Permissions portlet lets you view and modify the roles that users and groups have on resources.

"Setting resource permissions" on page 1568
Assign and control access for different types of resources.

"Delegated Access Control Administration" on page 1569
IBM WebSphere Portal Express supports delegated access control administration.

"Access Control Caching" on page 1570
Access Control internally uses several caches to improve the access control decision times. You can improve access control performance for special scenarios by setting the lifetime and size properties of these caches in the Cache Manager Service. In most cases, WebSphere Portal Express will run smoothly with the default cache settings. However, if you have a large number of resources or a large number of customized resources, you may want to adjust cache settings and conduct some tests to find the best performance trade-offs.

Managing Access Control

Get familiar with concepts related to administering IBM WebSphere Portal Express access control. To administer access control, use the Resource Permissions portlet, the User and Group Permissions portlet, the Manage Users and Groups portlet, the XML configuration interface, or the Portal Scripting Interface.

Authorization

Authorization is sometimes referred to as access control. Authorization determines what interactions a user is permitted to have with a resource or a service. Administrators configure access to resources or services by assigning roles to users and groups.

WebSphere Portal Express supports fine-grained access control over resources. Users can select and view only those resources for which they have appropriate access rights. When rendering a resource, WebSphere Portal Express verifies that the user has appropriate rights to use the requested resource. You can administer access rights by using the following portal tools:

- Dedicated access control administration portlets called **User and Group Permissions** and **Resource Permissions**
- The group membership portlet called **Manage Users and Groups**
- The Portal Scripting Interface
- The XML configuration interface (XML Access)
- The Manage Pages portlet
- The Portal 8.5 theme.

Access control information is accessible through the XML configuration interface. By default access control data is stored in the WebSphere Portal Express database. Alternately, you can configure an external security manager, such as IBM Security Access Manager, to host parts of the access control data and to manage role assignments.

All unauthenticated users are considered anonymous users. The access control component provides a dedicated virtual principal called Anonymous Portal User to represent such users. Prior to authenticating, an anonymous user, represented by this virtual principal, has specific access to a resource or service. In order for users to benefit from user and group specific privileges, they must be successfully authenticated by the system. Access control is dependent on the authentication of actual users.

WebSphere Portal Express only protects resources and services. WebSphere Application Server protects J2EE artifacts (for example servlet URLs and Enterprise Java Beans™ methods) and its artifacts (like server or node configurations).

WebSphere Portal Express Administrator and Security Administrator

The `Administrator@Portal` and `Security Administrator@Portal` roles contain a special permission that is not available to any other role. This permission allows the Administrator or Security Administrator to make arbitrary changes to the access control configuration of all resources. The Administrator and Security Administrator can create and delete roles, role assignments, and role blocks. If the configuration allows an external security manager such as IBM Security Access Manager to manage role assignments, additional privileges need to be set to allow arbitrary changes to the access control configuration. To change the access control configuration for resources that are externally managed, you must have the `Administrator@External Access Control` or the `Security Administrator@External Access Control` role.

Resources, roles, access rights, and initial access control settings

In order to fine-tune the security measures of your IBM WebSphere Portal Express environment, the administrator creates resources, roles, and access rights, which allows the administrator to control who has access to various information based on their roles and the access rights to that information.

Review the following information to learn more about resources, roles, access rights, and initial access control settings:

"Resources"
Resources are organized in a hierarchy. Resources in the hierarchy propagate their access control configuration to all of their child resources. If a user has the Editor role on the Market News page, then that user also has the Editor role on child pages of the Market News page. Resource instances are specific resources, such as a single portlet or page. Each resource instance belongs to only one resource type. For example, the resource instance Market News Page would belong to the Content Nodes resource type.

"Roles" on page 1533
Roles provide task permissions for users on resources. For example, Editor is a role that allows users to view, modify, and create resources. Roles are denoted as `Role@`*`Resource`*; for example, `Editor@`*`Portal Page`*.

"Access permissions" on page 1537
Learn about sensitive operations for resources and the roles that are required to perform those operations. Sensitive operations include common tasks such as viewing portlets on specific pages and complex, high-risk tasks like running XML configuration interface scripts.

"Initial Access Control Settings" on page 1550
The administrative user who installs IBM WebSphere Portal Express has a default set of access rights.

Resources

Resources are organized in a hierarchy. Resources in the hierarchy propagate their access control configuration to all of their child resources. If a user has the Editor role on the Market News page, then that user also has the Editor role on child pages of the Market News page. Resource instances are specific resources, such as a single portlet or page. Each resource instance belongs to only one resource type. For example, the resource instance Market News Page would belong to the Content Nodes resource type.

Virtual resources are a unique resource type. Virtual resources have two functions:

- They protect sensitive operations that affect the entire portal or specific services in the portal. For example, the virtual resource XMLAccess protects the ability to run XML configuration interface scripts.
- They are parent resources for all resource instances. For example, the **Web Modules** virtual resource is the root node of all web modules instances. So by default role assignments on the **Web Modules** virtual resource are propagated to all individual **Web Modules** resources through inheritance.

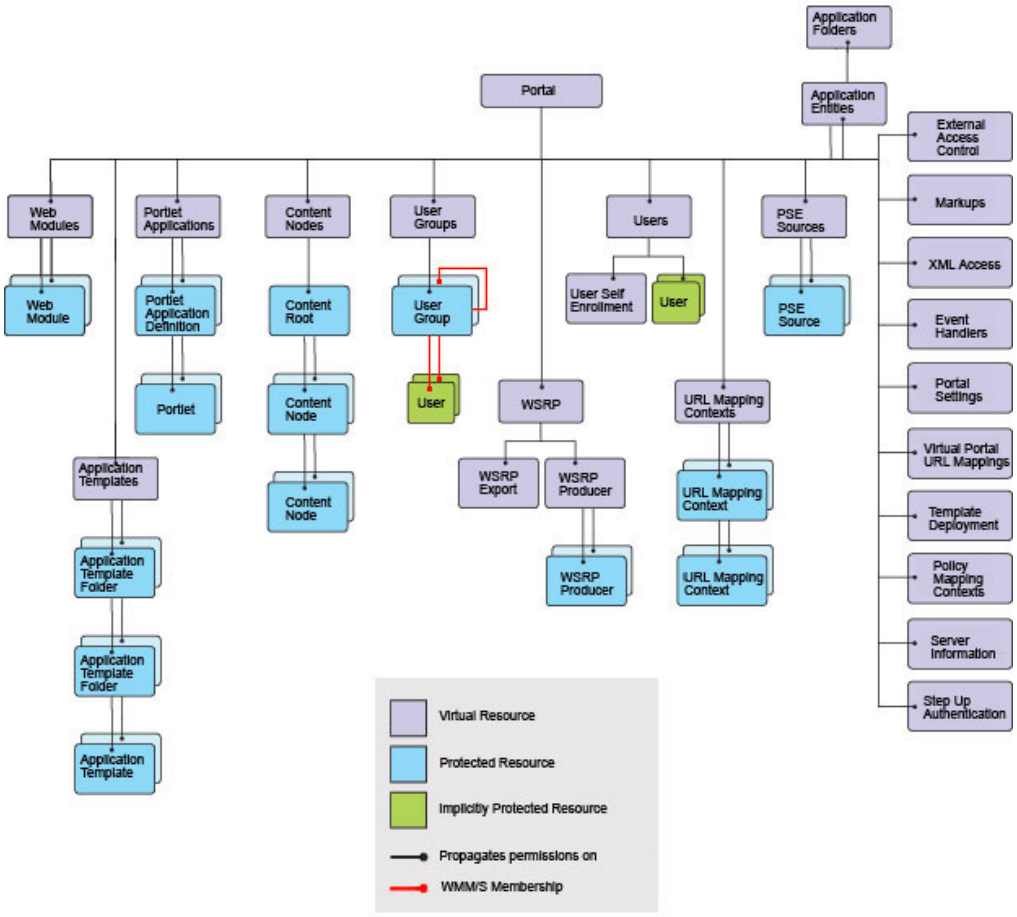
Resource data is stored in one of four different database domains. To have a consistent database back-up and restore, the access control data protecting individual resources is stored in the same database domain as the resource data. In each of the four domains, the protected resources are stored in a hierarchy as a single tree of resources. They are also known as the protected resource hierarchy.

Resources might be in different domains depending on the type of resource. JCR nodes are exclusively contained in the JCR domain. User customization data represented by private resources are exclusively contained in the customization domain. The community domain contains resources related to collaborative applications, and the release domain contains all remaining resources. Resources can be administered in the following ways:

- Protected Resources of the release domain can be managed through the access control administration portlets and through the XML Configuration interface
- Policy resources are stored in the JCR domain and can also be managed through the access control administration portlets and through the XML Configuration interface
- Resources in the community domain can be managed only through collaboration application-specific administrative portlets. Resources in this domain are not shown in the access control administration portlets
- The customization domain holds private resources for users only. No role assignments are possible in this domain, so resources in this domain are also not shown in the access control administration portlets

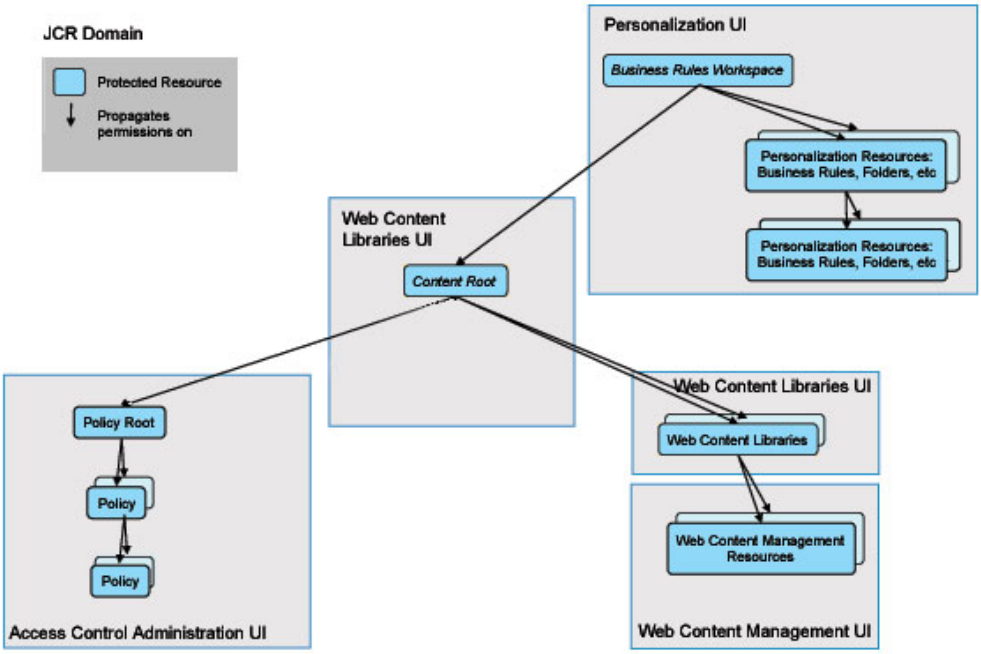
Note: Role inheritance never crosses domain boundaries, thus limiting the inheritance scope. A role assignment for a user on the Content Nodes virtual resource in the release domain grants access only to Content Nodes resources (pages) in the release domain.

Next are illustrations of the available resources tree, first for the release domain, and second for the JCR domain.



The following illustration shows the hierarchy of resources in the JCR domain. These resources are related to Personalization, Web Content Manager, and Resources Policies.

Note: This image represents an access control-specific view of resources in the JCR domain. It is not intended to show how the resources are stored and organized in the JCR domain.
Resource Permission inheritance applies to this hierarchy and to the release domain. Permission granted on the JCR Content Root node are propagated to all children in the hierarchy. Use Policies, Web Content Manager Libraries, Inheritance, and Propagation role blocks to reduce this propagation of permissions to children in the hierarchy.



A different user interface is provided to administer access control for each type of resource in the JCR domain. The following list shows the path to take within WebSphere Portal Express to reach the access control portlet for each resource stored in the JCR domain:

- Access Control Administration User Interface (UI):
  - Click the **Administration menu** icon. Then, click **Access** > **Resource Permissions**.
  - Click the **Administration menu** icon. Then, click **Access** > **User and Group Permissions**.
- Personalization user interface: **Personalization** > **Business Rules** > **Personalization Navigator portlet**
- Web Content Libraries user interface: Click the **Administration menu** icon. Then, click **Portal Content** > **Web Content Libraries**.
- Web Content Manager user interface: **Applications** > **Content** > **Web Content Management** > **Authoring portlet**

You can assign roles on virtual resources and on resource instances. Assigning roles on virtual resources reduces the time required to administer access control. All child resources inherit roles that are assigned to the parent resource by default. Assigning roles to specific resource instances offers more granular access control. You can assign roles to specific resource instances to override role blocks that block inheritance. For more information about role blocks, see the Roles topic.

The following information describes virtual resources. The resources listed might be different depending on other products that are installed with WebSphere Portal Express.

Administrative Slots

The root node of all administrative vault slots. It protects the administrative slot access and therefore the access to the credentials the slot holds.

Content Nodes
The root node of all pages, labels, and external URLs. Pages contain the content that determines the portal navigation hierarchy. If a new top-level page is created, it is automatically a child resource of the Pages virtual resource. If a new page is created beneath an existing page, the new page is automatically child of the existing page. Pages inherit access control configuration from their parent page unless role blocks are used.

Designer Deployment Service
Protects the ability to run the automatic deployment feature of IBM Workplace Designer.

Event Handlers
Protects management of Event Handlers. This virtual resource has no child resources.

External Access Control
Protects modifying access control configuration for resources that are controlled externally by a security manager such as Security Access Manager. Also protects the ability to externalize or internalize a resource. This virtual resource has no child resources.

Markups
Protects the ability to control markups for the portal. This virtual resource has no child resources.

Portal This resource is the root node of all resources in the release domain. Roles on this resource affect all other resources in the release domain by default through inheritance unless role blocks are used. Resources in other domains like Templates and Policies are not affected through role mappings on this resource.

Portal Settings
Protects portal settings that can be modified through the Portal Settings Portlet or the XML configuration interface. This virtual resource has no child resources.

Portlet Applications
The root node of all installed portlet applications. Portlet applications are the parent containers for portlets. If a new web module is installed, the applications that are contained within that module become child resources of the Portlet Applications virtual resource. Portlets that are contained within a portlet application are child nodes of that portlet application. Thus a two-layer hierarchy consisting of portlet applications and the corresponding portlets exists beneath the Portlet Applications virtual resource. Portlets inherit access control configuration from their parent portlet applications unless role blocks are used.

PSE Sources
The root node of all search collections. If a new search collection is created, it is automatically a child of this virtual resource. Roles on this resource affect all defined search collections unless role blocks are used.

URL Mapping Contexts
The root node of all URL mapping contexts. URL mapping contexts are user-defined definitions of URL spaces that map to portal content. If a new top-level URL mapping context is created, it is automatically a child resource of the URL Mapping Contexts virtual resource. If a new URL mapping context is created beneath an existing context, the new context is

automatically a child of the existing context. URL mapping contexts inherit access control configuration from their parent context unless role blocks are used.

User Groups
The root node of all user groups. Each user group in the portal inherits its access control configuration from the User Groups virtual resource. It is not possible to create role blocks on individual user groups.

User Self Enrollment
Protects the Selfcare and User Enrollment facilities (sign up and Edit My Profile). This virtual resource has no child resources.

Users This virtual resource has no child resources. The Users virtual resource protects sensitive operations that deal with user management. For example, in order to add a user to a user group, you must have the `Security Administrator@Users` role. Users are implicitly protected resources. Users cannot be protected individually, but only through their group membership. As a result, it is not possible to have a role assignment on a specific user. Roles must be on user groups instead. So, you can edit the user profile of Mary if you have a role assignment on some user group to which Mary belongs.

VP URL Mappings
Protects the ability to modify a URL Mapping linked to a virtual portal.

Web Modules
The root node of all **Web Modules**. **Web Modules** are portlet WAR files that are installed on WebSphere Application Server. **Web Modules** can contain multiple portlet applications. If a new **Web Modules** is installed, it is automatically a child of the **Web Modules** virtual resource. Roles on this resource affect all child resources (all installed **Web Modules**) unless role blocks are used.

WSRP This resource is the parent resource of the virtual resources `WSRP Export` and `WSRP Producers`. By default, roles on the `WSRP` resource affect the other two virtual WSRP resources and all WSRP resource instances through inheritance. If there are no role blocks in between, users who have role assignments on the `WSRP` resource have access rights on *all* WSRP resources.

WSRP export
This virtual resource controls the ability of a user to provide and withdraw portlets as a WSRP Service.

WSRP Producers
This resource is the root node of all registered Producer instances. Each Producer that is registered in the portal inherits its access control configuration from the `WSRP Producers` virtual resource unless role blocks are used.

XML configuration interface
Protects the ability to run XML configuration interface scripts. This virtual resource has no child resources.

ADMIN_SLOTS
The root node of all shared administrative credential vault slots that contain a system credential. This node controls access to modify and delete such vault slots and to retrieve its credentials.

POLICY MAPPING CONTEXTS
Identifies the Root node to all Policy items. This virtual resource is independent of the Mapping of a Policy.

SERVER INFORMATION
Protects the ability to create/modify/delete Mappings between a remote server information used for Federation and this WebSphere Portal Express server instance.

STEP UP AUTHENTICATION
Protects the ability to modify the binding of resources such as Portlets or Pages to an authentication level.

TAGS Users can apply keywords to describe, classify, or label web content resources.

RATINGS
Users can assign numeric values to web content resources for evaluation.

THEME MANAGEMENT
Users can update and modify the portal theme.

"Role blocks for resources"
Role blocks prevent inheritance and propagation through the resource hierarchy. This topic describes role blocks and provides examples of how role blocks affect resources.

Role blocks for resources:

Role blocks prevent inheritance and propagation through the resource hierarchy. This topic describes role blocks and provides examples of how role blocks affect resources.

Role blocks

There are two types of role blocks:

Inheritance blocks
Prevent child resources from acquiring role assignments from parent resources.

Propagation blocks
Prevent parent resources from extending role assignments to child resources.

Inheritance blocks and propagation blocks are similar in that they prevent parent resources from affecting child resources. You apply inheritance blocks to prevent parent resources from affecting only specific child resources. You apply propagation blocks to prevent parent resources from affecting all child resources.

Role blocks apply to specific resources. For example, the Market News page is the parent of the Europe Market News page and the USA Market News page. An inheritance block exists for the Editor role on the Europe Market News page. No inheritance blocks exist for the USA Market News page. All users with the Editor role on the Market News page (`Editor@Market News Page`) inherit the Editor role for the USA Market News page (`Editor@USA Market News Page`), but do not inherit the Editor role for the Europe Market News page.

Role blocks apply to specific roles. For example, an inheritance block exists for the Editor role on the Europe Market News page. This role block prevents the Europe

Market News page from acquiring any Editor role assignments from the Market News page. However, this role block does not affect inheritance of other roles. For example, no inheritance blocks exist for the Manager role. For this reason, all users with the Manager role on the Market News Page (`Manager@Market News Page`) inherit the Manager role on the Europe News Page (`Manager@Europe Market News Page`).

Creating and deleting role blocks

Use the following to create and delete role blocks:

XML configuration interface
Create and delete role blocks for all roles.

Portal scripting interface
Create and delete role blocks for all roles, except for Administrator and Security Administrator.

User and Group Permissions portlet
Create and delete role blocks for all roles, except for Administrator and Security Administrator.

Resource Permissions portlet
Create and delete role blocks for all roles, except for Administrator and Security Administrator.

For example, a user named Mary has the Administrator role on the Market News page (`Administrator@Market News Page`). The Market News page is the parent of the USA News Page. Mary automatically has the Administrator role on the USA Market News Page (`Administrator@USA Market News Page`) if you do not create a role block with the XML Configuration Interface.

All roles, including Administrator and Security Administrator, are automatically blocked for the following:

- Private pages
- Externalized resources that have an internal parent resource
- Internal resources that have an externalized parent resource

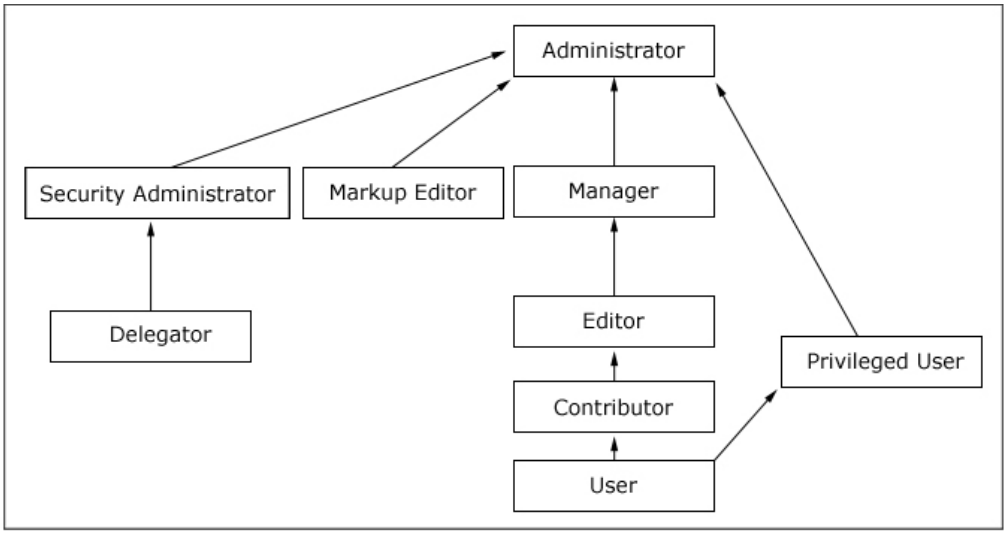
For example, WebSphere Portal Express controls access to the Market News page. IBM Security Access Manager controls access to the USA Market News page. This scenario is one in which an externalized resource, the USA Market News page, has an internal parent resource, the Market News page. In this scenario, the USA Market News Page, the child, does not inherit roles from the Market News Page, the parent. For this reason, if a user named Mary has the Editor role on the Market News Page (`Editor@Market News Page`), she does not automatically get the Editor role on the USA Market News page (`Editor@USA Market News Page`).

Roles

Roles provide task permissions for users on resources. For example, Editor is a role that allows users to view, modify, and create resources. Roles are denoted as `Role@`*`Resource`*; for example, `Editor@`*`Portal Page`*.

Roles are organized in a hierarchy. Roles are organized in a hierarchy. Roles that are higher in the hierarchy generally inherit the permissions of child roles. For example, to install web modules the Editor role on the virtual resource `Web Modules`, `Editor@Web Modules`, is the minimum role assignment for this operation. The Manager role is higher in the hierarchy than the Editor role. For this reason,

the Manager role includes the permissions of the Editor role. The `Manager@Web Modules` role also allows users to install web modules.



The following table describes the different allowed actions for roles:

Table 192. Allowed actions for roles

Role	Allowed Actions
Administrator	Unrestricted access on resources, which includes creating, configuring, and deleting resources. Administrators can also change the access control settings on resource; in other words grant other people access to those resources.
Security Administrator	Creating and deleting role assignments on resources. The Security Administrator role allows the user to act as a delegated administrator for that resource. The Security Administrator can delegate a subset of their privileges to other people according to the Delegated Administration Policy topic. For example, a user with Security Administrator and Editor roles can assign the Editor role to other people. The Security Administrator role on a resource does not give view or edit access to the resource.
Delegator	Assigning the Delegator role to principals (users and groups) allows roles to be granted to them. Having the Delegator role on other resources, such as specific portlets, is not useful. The set of roles that can be granted to those principals is defined through the Security Administrator and Administrator roles. For example, a user has a Delegator role on the SalesTeam user group but no Delegator role on the Managers user group. Therefore, this user can grant roles only to the SalesTeam or individual members of the SalesTeam user group but not to the Managers user group. The Delegator role on a resource does not give direct access to the resource. The purpose of the Delegator role is to allow the granting of roles to users or groups. Therefore, assigning the Delegator role on resources or resource types that are not users or user groups does not grant those users more privileges.

Table 192. Allowed actions for roles (continued)

Role	Allowed Actions
Can run as User (user impersonation)	After you enable the Impersonation feature, you can assign a user the Can run as User role. It allows them to view pages, portlets, and other portal components as another user. Support specialists can use this role to troubleshoot.
Manager	Creating new resources and configuring and deleting existing resources that are used by multiple users.
Editor	Creating new resources and configuring existing resources that are used by multiple users.
Markup Editor	Changing the HTML source for static portal pages.
Contributor	Viewing portal content and creating new resources. The Contributor role does not include the permission to edit resources. You can create only new resources. For example, a user is granted the Contributor role on the Template Category Teamspace. The user cannot modify the category itself but can create new templates in this category. **Note:** This role is only available for the following resources: • Application Templates • Application Template Categories • Application Template Root • Policies • All IBM Web Content Manager related documents
Privileged user	Viewing portal content, customizing portlets and pages, and creating new private pages.
User	Viewing portal content. For example, viewing a specific page.
No role that is assigned	Cannot interact with resource.

Application Roles

There is a higher level concept of roles called application roles. Application roles are identified by a unique name and can contain an arbitrary set of other roles (an example is `Editor@Market News page` and `Editor@Market News portlet`). It is possible to use application roles to bundle cohesive allowed actions, simplifying access control administration. Application roles with the same name in different database domains are correlated. It is possible to aggregate roles from different database domains within one application role.

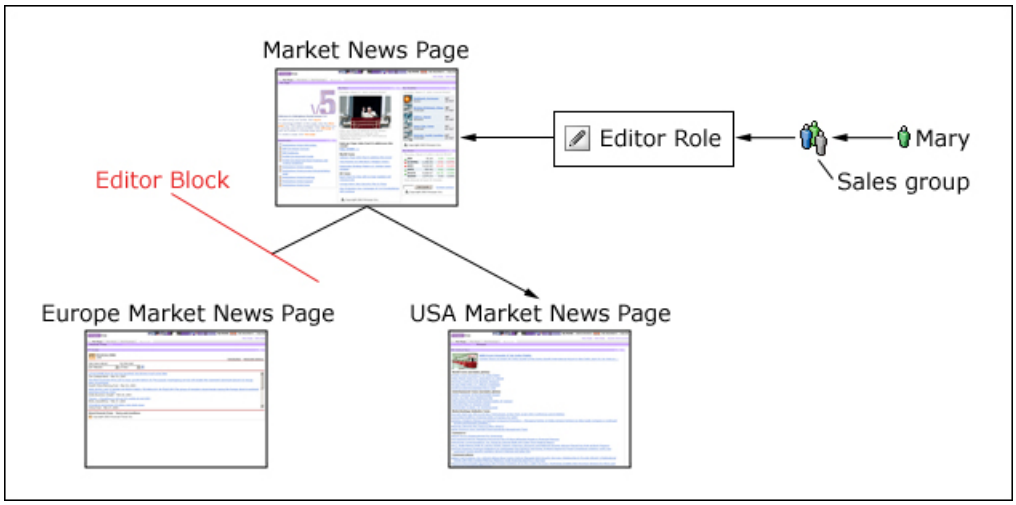
Inheritance

Resources are part of a hierarchy. By default, each resource in the hierarchy inherits the role assignments of its parent resource. This inheritance reduces the administration work. When you assign a group to a role on a parent resource, the group automatically acquires that same set of allowed actions for all child resources.

For example, suppose that a user, Mary, is a member of the Sales group. You can give Mary Editor access to the Market News page and all child pages by granting

the Editor@*Market News Page* role to the Sales group. All members of the Sales group implicitly acquire the Editor@*Market News Page* role. All members of the Sales group also inherit the Editor role on all child pages of the Market News page in the resource hierarchy. So, members of the Sales group automatically inherit the role Editor@*USA Market News Page*.

Inheritance through the resource hierarchy can be blocked at any level to provide more granular access control.



Role Assignments

Roles are assigned to users and groups that are contained in the user registry. Roles can be assigned by someone with the necessary authorization, such as the portal administrator, in any of three ways:

- Explicitly assigned to an individual user
- Implicitly assigned through group membership. If a group has a role, all members of the group automatically acquire the role. Nested groups (groups that are members of another group) inherit role assignments from their parent groups.
- Inherited through a role assignment on a parent resource. By default, roles on a resource automatically apply to all child resource unless role blocks are used.

Users and groups can have multiple roles on the same resource. For example, a user might have both the Editor and Manager roles on a particular page. One of these roles might be inherited through the resource hierarchy and the other might be explicitly assigned. If two roles in the same hierarchy are assigned for a user for the same resource, the higher role takes precedence. For example, if a user has the Manager and Editor role on a resource, the Manager role takes precedence over the Editor role.

Assign roles to individual users only in exceptional cases. Add or remove users from groups to reduce the number of role mappings and simplify maintenance.

Ownership

Each resource can have a dedicated owner. The resource owner can be a single user or a single user group. When a user creates a resource, such as a page, the user automatically becomes the initial owner of that resource. For non-private

resources, ownership provides the same set of allowed actions as the Manager role. For private resources, ownership provides the same set of allowed actions as the Privileged User role. This user can also delete the resource. For both non-private and private resources, these actions include the ability to delete the resource. Private resources can be owned only by users, not by user groups. It is not possible to define roles on private resources, and resource ownership cannot be inherited.

You can use the XML configuration interface or the Resource Permissions portlet to change the owner of a resource.

Private pages

A private page can be accessed only by its owner. Privileged Users (users assigned the `Privileged User` role) can explicitly create new private pages that are accessible only by themselves. Additionally, a Privileged User on a non-private page can personalize the page and create new private pages underneath it. Customizing a non-private page usually creates a private copy of the corresponding non-private page. Any changes that a Privileged User makes to a non-private page are not accessible by other users.

Note: Private pages cannot be controlled by an external security manager. Access control for private pages is always internally controlled by WebSphere Portal Express.

Traversal support

Users with role assignments on the resources `Page` or `URL Mapping` get the implicit permission to go to those resources. These users can go through all parent resources of those resources. Users see only the title of those resources. The corresponding resource content remains inaccessible unless those users have further role assignments that grant them normal access to those resources.

Related tasks:

"Automatically grant page access to community members" on page 759
If you want community members to automatically be able to access the page, without explicitly configuring access, you must enable that feature. Community membership must be integrated with portal security before you can enable this feature.

Related reference:

"Delegated Access Control Administration" on page 1569
IBM WebSphere Portal Express supports delegated access control administration.

Access permissions

Learn about sensitive operations for resources and the roles that are required to perform those operations. Sensitive operations include common tasks such as viewing portlets on specific pages and complex, high-risk tasks like running XML configuration interface scripts.

Roles provide permissions for user to perform specific operations on resources. The following tables denote roles as follows: `Role@`*`Resource`*.

Notes about the following tables:

- The following tables list minimum role assignments that are necessary to perform sensitive operations. Roles are organized in a hierarchy. Roles that are higher in the hierarchy generally include the permissions of roles that are lower

in the role hierarchy. For example, to install web modules the editor role on the virtual resource `Web Modules`, `Editor@Web Modules`, is the minimum role assignment for this operation. The manager role is higher in the hierarchy than the editor role. For this reason, the manager role includes the permissions of the editor role. `Manager@Web Modules` also allows users to install web modules.

- When access permissions are granted to any listed resource, it inherently requires access to the resource Access Control Administration.
- Use the Access Control Administration to change the owner of a resource.
- The resources that are listed can be different depending on other products that might be installed with the product. Some roles are required on virtual resources; other roles must be on resource instances.
- Users might also have access permissions for some operations through ownership of resources.
- Definition of terms:

  private
  Accessible only by the owner of the resource.

  Note: Creators of private resources automatically gain rights that are similar to the rights of a Manager. For example, if you create a private page, you have rights similar to manager for that page. You can also perform certain actions such as changing the page theme or deleting the page.

  non-private
  Accessible by those people who were granted access to the resource.

  public Accessible without authentication.

Table 193. Access permissions for Access Control Administration

Sensitive operation and description	Required role assignment
Viewing the access control configuration of a resource *R*	If *R* is under internal PORTAL protection: `Security Administrator@`*`R`* or `Security Administrator@`*`PORTAL`*. If *R* is under external protection: `Security Administrator@`*`R`* or `Security Administrator@`*`PORTAL`* + `Security Administrator@`*`EXTERNAL_ACCESS_CONTROL`* **Notes:** • *PORTAL* and *EXTERNAL_ACCESS_CONTROL* are virtual resources. • The `Security Administrator@`*`EXTERNAL_ACCESS_CONTROL`* role is created and managed in the External Security Manager. It must be modified with the external security management tools. For example, use the IBM Security Access Manager `pdadmin>` command line or the Computer Associates eTrust SiteMinder administrative console.

Table 193. Access permissions for Access Control Administration (continued)

Sensitive operation and description	Required role assignment
Creating a role *RT* on resource *R*	If *R* is under PORTAL protection: `Security Administrator@`*`R`* + *`RT@R`* or `Security Administrator@`*`PORTAL`* If *R* is under external protection: `Security Administrator@`*`R`* + *`RT@R`* or `Security Administrator@PORTAL` + `Security Administrator@`*`EXTERNAL_ACCESS_CONTROL`* **Notes:** • *PORTAL* and *EXTERNAL_ACCESS_CONTROL* are virtual resources. • The `Security Administrator@`*`EXTERNAL_ACCESS_CONTROL`* role is created and managed in the External Security Manager. It must be modified with the external security management tools. For example, use the Security Access Manager `pdadmin>` command line or the eTrust SiteMinder administrative console.
Deleting a role that is created from role *RT* on resource *R*. All corresponding role mappings are also deleted.	If *R* is under internal PORTAL protection: `Security Administrator@`*`R`* + *`RT`*`@R` + Delegator role on all assigned principals or `Security Administrator@PORTAL` If *R* is under external protection: `Security Administrator@`*`R`* + *`RT@R`* + Delegator role on all assigned principals or `Security Administrator@`*`PORTAL`* + `Security Administrator@`*`EXTERNAL_ACCESS_CONTROL`* **Notes:** • *PORTAL* and *EXTERNAL_ACCESS_CONTROL* are virtual resources. • The `Security Administrator@`*`EXTERNAL_ACCESS_CONTROL`* role is created and managed in the External Security Manager. It must be modified with the external security management tools. For example, use the Security Access Manager `pdadmin>` command line or the eTrust SiteMinder administrative console.
Creating or deleting a role assignment for user or group *U* created from role *RT* on resource *R*	If *R* is under internal PORTAL protection: `Security Administrator@`*`R`* + *`RT@R`* + `Delegator@`*`U`* or `Security Administrator@`*`PORTAL`* If *R* is under external protection: `Security Administrator@`*`R`* + *`RT@R`* + `Delegator@`*`U`* or `Security Administrator@`*`PORTAL`* + `Security Administrator@`*`EXTERNAL_ACCESS_CONTROL`* **Notes:** • *PORTAL* and *EXTERNAL_ACCESS_CONTROL* are virtual resources. • The `Security Administrator@`*`EXTERNAL_ACCESS_CONTROL`* role is created and managed in the External Security Manager. It must be modified with the external security management tools. For example, use the Security Access Manager `pdadmin>` command line or the eTrust SiteMinder administrative console.

Table 193. Access permissions for Access Control Administration (continued)

Sensitive operation and description	Required role assignment
Creating or deleting a role block for all roles that are created from role *RT* on resource *R*	If *R* is under internal PORTAL protection: `Security Administrator@R + RT@R` or `Security Administrator@PORTAL` If *R* is under external protection: `Security Administrator@R + RT@R` or `Security Administrator@PORTAL + Security Administrator@EXTERNAL_ACCESS_CONTROL` **Note:** A Security Administrator on this resource is always implicitly a Delegator on this resource. For all other roles, the `Security Administrator@R` plus the previous assignments are required. **Notes:** • *PORTAL* and *EXTERNAL_ACCESS_CONTROL* are virtual resources. • The `Security Administrator@EXTERNAL_ACCESS_CONTROL` role is created and managed in the External Security Manager. It must be modified with the external security management tools. For example, use the Security Access Manager `pdadmin>` command line or the eTrust SiteMinder administrative console.
Externalizing or internalizing resources:Moving a resource *R* back and forth from internal to external control. All non-private child resources of *R* move with it. Private resources cannot be externalized.	`Security Administrator@R + Security Administrator@EXTERNAL_ACCESS_CONTROL` or `Security Administrator@Portal + Security Administrator@EXTERNAL_ACCESS_CONTROL` **Notes:** • *Portal* and *EXTERNAL_ACCESS_CONTROL* are virtual resources. • The `Security Administrator@EXTERNAL_ACCESS_CONTROL` role is created and managed in the External Security Manager. It must be modified with the external security management tools. For example, use the Security Access Manager `pdadmin>` command line or the eTrust SiteMinder administrative console.
Modifying the owner of a resource:Setting a user or group *U1* as new owner of the non-private resource *R*, where the old owner was *U2*	`Delegator@U1`, `Delegator@U2`, `Manager@R`, and `Security_Administrator@R`

Table 194. Access permissions for Business Rules

Sensitive operation and description	Required role assignment
Viewing a Business Rule	`User@Business Rules Workspace` Set this permission on the Business Rules workspace in the Personalization navigator by selecting the root node and then choosing **Extra Action** > **Edit Access** from the menu.
Creating a Business Rule	`Contributor@Business Rules Workspace` **Important:** `Contributor@Business Rules Workspace` is the minimum required access permission to create a Business Rule. However, you must use `Editor@Business Rules Workspace` to create and maintain business rules and use the Portal administration facilities.
Deleting a Business Rule	`Manager@Business Rules Workspace`
Assigning a Business rule to a page *P*	For non-private pages: `Editor@P` and `User@Business Rules Workspace` For private pages: `Priviliged User@P` and `User@Business Rules Workspace`

Table 194. Access permissions for Business Rules (continued)

Sensitive operation and description	Required role assignment
Assigning a Business rule to a portlet *PO* on page *P*	For non-private pages: `Editor@P`, `User@PO`, and `User@Business Rules Workspace` For private pages: `Privileged User@P`, `User@PO`, and `User@Business Rules Workspace`
Additional actions	Use the **Set Access** icon in Personalization to add a user or a group to a role on the root of the workspace. The same role is given to that user or group for all Web Content Manager libraries, policies, and templates.

Table 195. Access permissions for Content Nodes such as pages, labels, and URLs

Sensitive operation and description **Note:** The operations in this column specifically refer to pages only, but also applies to labels and URLs in some cases.	**Required role assignment**
Traversing a page:Viewing the navigation of a page *P*	`User@P` or @ some child resource of *P*
Viewing the content of a page *P*, including page decoration and potentially the portlets on that page. The portlets on a page are protected separately. See the portlets on pages row of this table for information.	`User@P`
Modifying page properties includes the following actions: Adding or removing markup Adding or removing a locale Adding or removing parameters to or from a page *P*	`Editor@P`
Modifying page properties includes Set page layout properties of a static page.*P*	`Markup editor role`. If the resources are in secure locations of layout templates, use `Manager role`. For more information, see *Adapt the list of required runtime configuration changes for your theme* in the related links.
Changing the theme of a page *P*	`Editor@P`
Modifying the layout of a page *P* includes the following actions: Adding or removing wires Managing actions	For non-private pages: `Editor@P` For private pages: `Privileged User@P` For managing receiving actions of a portlet on a target page: `Editor@P` and `Editor@PO`
Customizing the layout of a non-private page: Creating a private, implicitly derived copy of a non-private page *P*	`Privileged User@P`
Adding a root page:Creating and adding a new top-level page *P*	For non-private pages: `Editor@Pages` For private pages: `Privileged User@PagesPages` is a virtual resource.
Adding a page:Creating a page under any Page *P*	For non-private pages: `Editor@P` For private pages: `Privileged User@P`
Creating a derived page:Creating a page underneath *P1* that is explicitly derived from page *P2*	New page is private: `Privileged User@P1` + `Editor@P2` New page is non-private: `Editor@P1` + `Editor@P2`
Deleting a page *P* and all descendant pages, including further subpages and the portlets on those pages	`Manager@P`
Moving page *P1* to a new parent page *P2*	For non-private pages: `Manager@P1` + `Editor@P2` For private pages: `Manager@P1` + `Privileged User@P2`
Locking or unlocking the contents of a non-private page *P*	`Editor@P` + `User@portlet` (Page Locks) + `User@page` (Locks)
Edit page associations for a non-private page *P*	`Editor@P`
Edit page associations for a private page *P*	`Privileged User@P`

Table 195. Access permissions for Content Nodes such as pages, labels, and URLs (continued)

Sensitive operation and description **Note:** The operations in this column specifically refer to pages only, but also applies to labels and URLs in some cases.	**Required role assignment**
Enabling membership-based access control delegation for a Community Page *P* associated to an IBM Connections Community *C* represented by the virtual user groups *G*. It is activated through the **Limit access to this page to only community members** Page Associations check mark.	`Editor@P + Security Administrator@P + Delegator@G + View Privileges@C(in IBM Connections)`
Activating Portal Page Security for a web content page *P* that is associated with site area *SA* in web content library *L*. This security is activated through the **Use Portal Page Security** check mark in the Page Associations window.	`Editor@P + User@SA + Administrator@L` and `Editor@P + User@SA + Administrator@L + Manager@VirtualResource CONTENT MAPPINGS`

Table 196. Access permissions related to Template Page instantiation

Sensitive operation	**Required role assignment**
Adding a root page Creating and adding a new top-level page *Pages* based on page template *T*	For non-private pages: `Editor@Pages` and `User@T` For private pages: `Privileged User@Pages` and `User@T` Additional roles can be required based on instantiation features associated to page template *T*: • *T* is associated to site area *SA1* in Web Content Manager, and the **wps.content.root** label is associated with site area *SA2*, with default content associations on each site area. Web Content Manager view permissions on *SA1* and Web Content Manager create content permissions on *SA2*. • *T* is associated to an IBM Connections community *C*. Grant the following privileges to the user in IBM Connections: – View *C* – Create new communities • *T* is configured to create a community during instantiation with the **ibm.portal.instantiation.community.create.new page** parameter. Grant the following privileges to the user in IBM Connections: Create new communities • *T* is enabled for Membership-based access control delegation: `Delegator@USER_GROUPS` *USER_GROUPS* is a virtual resource.
Adding a page Creating a page from Template *T* under any Page *P*	For private pages: `Privileged User@P` and `User@T` Additional roles can be required based on instantiation features associated to page template *T*: • *T* is associated to site area *SA1* in Web Content Manager, and the **wps.content.root** label is associated with site area *SA2*, with default content associations on each site area. Web Content Manager view permissions on *SA1* and Web Content Manager create content permissions on *SA2*. • *T* is associated to an IBM Connections community *C*. Grant the following privileges to the user in IBM Connections: – View *C* – Create new communities • *T* is configured to create a community during instantiation with the **ibm.portal.instantiation.community.create.new page** parameter. Grant the following privileges to the user in IBM Connections: Create new communities • *T* is enabled for Membership-based access control delegation: `Delegator@USER_GROUPS` *USER_GROUPS* is a virtual resource.

Table 197. Access permissions for the Credential Vault portlet

Sensitive operation and description	Required role assignment
Sensitive operation and description	Required role assignment
Adding, viewing, or deleting a vault segment	Management of the Credential Vault through the Credential Vault portlet requires access to an instance of the Credential Vault portlet.
Adding a shared administrative credential vault slot (containing a system credential)	Management of the Credential Vault through the Credential Vault portlet requires access to an instance of the Credential Vault portlet.
Retrieving the credential from a shared administrative credential vault slot (containing a system credential)	`User@slot` or `User@ADMIN_SLOTS`
Modifying a shared administrative credential vault slot (containing a system credential)	`Editor@slot` or `Editor@ADMIN_SLOTS`
Deleting a shared administrative credential vault slot (containing a system credential)	`Manager@slot` or `Manager@ADMIN_SLOTS`
Adding, viewing, deleting, or editing a non-shared vault slot	Management of the Credential Vault through the Credential Vault portlet requires access to an instance of the Credential Vault portlet.

Note: `Virtual resource: ADMIN_SLOTS` is a virtual resource. The permission on this node is propagated to all slots, if it is not blocked by an inheritance or propagation block.

Table 198. Access permissions for the **Enable Tracing** *portlet*

Sensitive operation and description	Required role assignment
Adding or deleting portal trace settings	Adding or deleting portal trace setting through the **Enable Tracing** portlet requires access to an instance of the **Enable Tracing** portlet.

Table 199. Access permissions for Event Handlers

Sensitive operation and description	Required role assignment
Managing event handlers:Creating, modifying, and deleting event handlers	`Security Administrator@Event Handlers` **Virtual resource:** *Event Handlers* is a virtual resource.

Table 200. Access permissions for the Manage Clients portlet

Sensitive operation and description	Required role assignment
Managing clients:Viewing the portlet; deleting, modifying, and adding clients in the `Manage Clients` portlet	`User@Manage Clients`

Table 201. Access permissions for Manage Search

Sensitive operation and description	Required role assignment
Creating a search index	`Editor@PSE_Sources` **Virtual resource:** *PSE_Sources* is a virtual resource.
Associating keywords with content items through the Search Center portlet, so that they are promoted to users who search for those keywords.	`Administrator@` for `Search Center Portlet` **Virtual resource:** *Search Center Portlet* is a virtual resource.
Modifying keywords that are associated with content items that exist in the Suggested Links portlet already.	`Administrator@` for `Suggested Links Portlet` **Virtual resource:** *Suggested Links Portlet* is a virtual resource.

Table 202. Access permissions for Manage Search

Sensitive operation and description	Required role assignment
Creating the New Virtual Portal	`Security Administrator@Portal` **Virtual resource:** *Portal* is a virtual resource.

Table 202. Access permissions for Manage Search (continued)

Sensitive operation and description	Required role assignment
Viewing the Virtual Portal	`Security Administrator@Portal` **Virtual resource:** *Portal* is a virtual resource.
Deleting the Virtual Portal	`Security Administrator@Portal` **Virtual resource:** *Portal* is a virtual resource.
Editing the Virtual Portal	`Security Administrator@Portal` **Virtual resource:** *Portal* is a virtual resource.

Table 203. Access permissions for Markups

Sensitive operation and description	Required role assignment
Managing Markups:Creating, deleting, or modifying a Markup	`Editor@Markups` **Virtual resource:** *Markups* is a virtual resource

Table 204. Access permissions for Policies

Sensitive operation and description	Required role assignment
Creating a Policy under any Policy	`Editor@Policy` and `User@Business Rules Workspace` **Notes:** • `Contributor@Policy` is the minimum required access permission to create a Policy under any Policy, though it is not recommended. `Editor@Policy` is recommended to create and maintain policies and use the Portal administration utilities. • If a rule must be created or edited during the creation of a Policy, then `Editor@Business Rules Workspace` and `Editor@Policy` is also required. • Business Rules workspace is the root node in the Personalization navigator for Business Rules resources. Set permissions on this node by selecting the workspace node and then choosing **Extra Action** > **Edit Access** from the menu.
Assigning a Business rule to a Policy	`User@Business Rules` and `Editor@Policy`
Editing a Policy	`Editor@Policy` and `User@Business Rules` **Note:** If a rule must be created or edited during the creation of a Policy, then `Editor@Business Rules` is also required.
Viewing a Policy	`User@Policy` + `User@Business Rules`
Importing a new Policy	`Editor@Policy_Root` **Important:** `Contributor@Policy_Root` is the minimum required access permission to import a new Policy, however, you must use `Editor@Policy_Root` to import and maintain policies and use the Portal administration utilities.
Deleting a Policy	`Manager@Policy` + `User@Business Rules` **Deleting policies:** When you delete a policy, the associated rule is not deleted.

Table 205. Access permissions for Portal Settings

Sensitive operation and description	Required role assignment
Viewing current portal settings	`User@Portal Settings` **Virtual resource:** *Portal Settings* is a virtual resource.
Modifying current portal settings	`Editor@Portal Settings` **Virtual resource:** *Portal Settings* is a virtual resource.

Table 206. Access permissions for Portlet Applications

Sensitive operation and description	Required role assignment
Viewing the portlet application definition information for a portlet application *PA*	User@*PA*
Modifying a portlet application includes the following actions: Adding or removing a locale Setting default locale Modifying settings to, from, or of the portlet application *PA*.	Editor@*PA*
Duplicating a portlet application:Creating a portlet application based on an existing portlet application *PA*	Editor@*Portlet Applications* + User@*PA* **Virtual resource:** *Portlet Applications* is a virtual resource.
Deleting a portlet application and removing all corresponding portlets and portlet entities from all pages within the portal	Manager@*PA*
Enabling or disabling a portlet application:Temporarily enabling or disabling the portlet application *PA*	Manager@*PA*

Table 207. Access permissions for portlets

Sensitive operation and description	Required role assignment
Viewing an installed portlet:Viewing the portlet definition information of a portlet *PO*	User@*PO*
Modifying an installed portlet includes the following actions: Adding or removing a locale Setting default locale Modifying settings to, from, or of the portlet *PO*.	For adding or removing locales and setting default locale: Editor@*PO* For modifying settings: Manager@*PO*
Duplicating an installed portlet:Creating a new installed portlet based on an existing portlet *PO* that is part of a portlet application *PA*.	Editor@*Portlet Applications* + User@*PO* + User@*PA* **Virtual resource:** *Portlet Applications* is a virtual resource.
Deleting an installed portlet *PO* and removing all corresponding portlet entities from all pages within the portal	Manager@*PO*
Enabling or disabling an installed portlet:Temporarily enabling or disabling a portlet *PO*	Manager@*PO*
Providing portlet *PO* as a WSRP service	Editor@*WSRP Export* and Editor@*PO* **Virtual resource:** *WSRP Export* is a virtual resource.
Withdrawing portlet *PO* from WSRP service	Manager@*WSRP Export* and Editor@*PO* **Virtual resource:** *WSRP Export* is a virtual resource.
Integrating the portlet of a WSRP Producer *PR* into the portal	If no portlet application exists for the group of portlets: Editor@*Portlet Applications* and User@*PR* **Virtual resource:** *Portlet Applications* is a virtual resource. If a Portlet Applications *PA* exists for the group of portlets: Editor@*PA* and User@*PR*
Deleting an integrated WSRP portlet *PO* contained in the portlet application *PA* from the portal	If this portlet is the last portlet in Portlet Applications: Manager@*PA* If more than one portlet is in Portlet Applications: Manager@*PO*

Table 208. Access permissions for portlets on pages

Sensitive operation and description	Required role assignment
Viewing a portlet *PO* on page *P*	User@*P* + User@*PO*
Configuring an installed portlet:Entering the configure mode of a portlet *PO* and modifying its configuration	Manager@*PO*

Table 208. Access permissions for portlets on pages (continued)

Sensitive operation and description	Required role assignment
Modifying a portlet on a page:Entering the edit mode of a portlet *PO* on page *P* and modifying its configuration **Note:** If *P* is a non-private page and the user has no Editor role for this page, then modifying the configuration of the portlet results in the creation of an implicitly derived copy of page *P*.	`Editor@P + Editor@PO` Or `Privileged User@P + Privileged User@PO`
Modifying page content:Adding or removing a portlet *PO* to/from a page *P* **Note:** If *P* is a non-private page and the user has no Editor role for this page, then modifying the content of *P* results in the creation of an implicitly derived copy of page *P*.	For non-private pages: `Editor@P + User@PO` Or For private pages: `Privileged User@P + User@PO`
Adding web content to a page: Adding a web content viewer portlet *PO* that is configured to render web content *C* from site area *SA* in Web Content Manager. Portlet *PO* is configured with the option **Create content (based on selection)**, and page *P* is associated with site area *SA*. **Note:** If *P* is a non-private page and the user has no Editor role for this page, then modifying the content of *P* results in the creation of an implicitly derived copy of page *P*.	• For non-private pages: `Editor@P + User @ PO` + Web Content Manager view permissions on *C* and Web Content Manager create content permissions on *SA*. • For private pages: `Privileged User@P + User@PO` + Web Content Manager view permissions on *C* and Web Content Manager create content permissions on *SA*
Restricting the content of a page:Adding or removing a portlet from the Allowed Portlet List of a page	`Editor@P + User@PO`

Table 209. Access permissions for Property Broker

Sensitive operation and description	Required role assignment
Operating with ActionSets or PropertySets for a portlet *PO*	`User@PO`
Creating, updating, or deleting a wire from a portlet *PO1* on Page *P1* to a portlet *PO2* on Page *P2*	Global wire: `Editor@P1, User@PO1, Editor@P2, User@PO2` Personal wire: `Privileged User@P1, User@PO1, Privileged User@P2, User@PO2` **Important:** To update or delete a personal wire, the user must have the previous role assignments and created the wire that they are updating or deleting.
Creating a wire from a portlet PO1 on Page P1 to a portlet PO2 on Page P2	Global wire: `User@P1, User@PO1, User@P2, User@PO2` Personal wire: `Privileged User@P1, User@PO1, Privileged User@P2, User@PO2` **Important:** To create a personal wire, the user must have the previous role assignments and created the wire that they are starting.
Viewing a wire from a portlet PO1 on Page P1 to a portlet PO2 on Page P2	Global wire: `User@P1, User@PO1, User@P2, User@PO2` Personal wire: `Privileged User@P1, User@PO1, Privileged User@P2, User@PO2` **Important:** To view a personal wire, the user must have the previous role assignments and created the wire that they are viewing

Table 210. Access permissions for PSE Sources

Sensitive operation and description	Required role assignment
Creating a PSE Source:Creating a search collection	`Editor@PSE Sources` **Virtual resource:** *PSE Sources* is a virtual resource.
Viewing a PSE Source:Viewing a search collection *SC*	`User@SC`
Facilitating a PSE Source:Using a search collection *SC*	`User@SC`
Editing a PSE Source:Editing a search collection *SC*	`Editor@SC`
Deleting a PSE Source:Deleting a search collection *SC*	`Manager@SC`

Table 211. Access permissions for tagging and rating

Sensitive operation and description	Required role assignment
Viewing community tags and ratings that other users applied. Creating and deleting personal public tags and ratings. Deleting community tags regardless of ownership.	`Manager@Tags + Manager@Ratings` **Virtual resource:** *Tags* and *Ratings* are virtual resources.
Viewing community tags and ratings that other users applied. Creating and deleting personal public tags and ratings.	`Contributor@Tags + Contributor@Ratings`
Viewing community tags and ratings that other users applied. Creating and deleting personal private tags and ratings.	`Privileged user@Tags + Privileged user@Ratings`
Viewing community tags and ratings that other users applied.	`User@Tags + User@Ratings`

Table 212. Access permissions for themes, skins, and layout templates

Sensitive operation and description	Required role assignment
Creating, viewing, editing, and deleting a Theme, Skin, or Layout Template	`Manager@THEME MANAGEMENT` **Virtual resource:** *THEME MANAGEMENT* is a virtual resource.

Table 213. Access permissions for the Unique Names portlet

Sensitive operation and description	Required role assignment
Managing unique names:Viewing the portlet; deleting, modifying, and adding unique names in the Unique Names portlet	`Editor@R + User@Unique Names`

Table 214. Access permissions for URL Mapping Contexts

Sensitive operation and description	Required role assignment
Creating a URL mapping context *UMC*	`Editor@URL Mapping Contexts` **Virtual resource:** *URL Mapping Contexts* is a virtual resource.
Traversing a URL mapping context:The ability to traverse a URL mapping context due to a role assignment to some child context of *UMC*	`User@UMC` or @ some child context of *UMC*
Viewing the definition of a URL mapping context *UMC*	`User@UMC`
Assigning a URL:Creating or editing a mapping between a URL mapping context *UMC* and a portal resource *R*	`Editor@UMC + User@R`
Modifying a URL mapping context:Changing the properties of an existing URL mapping context *UMC*; for example, editing the label	`Editor@UMC` If Virtual Portal Mapping: `Editor@VP URL Mappings` **Virtual resource:** *VP URL Mappings* is a virtual resource.
Deleting a URL mapping context *UMC* and all of its child contexts	`Manager@UMC`

Table 215. Access permissions for User Groups

Sensitive operation and description	Required role assignment
Creating a User group within the user registry	`Editor@User Groups` **Virtual resource:** *User Groups* is a virtual resource.
Viewing the User group profile information of a user group *UG*	`User@UG`
Modifying the profile information of a User group *UG*	`Editor@UG`
Adding or removing an existing User *U* or a User group *UG2* to or from an existing User group *UG1*	`Security Administrator@Users + Editor@UG1` **Virtual resource:** *Users* is a virtual resource.
Deleting a user group *UG*	`Manager@UG` **Deleting the user group:** The owner of the user group can also delete it.

Table 216. Access permissions for users

Sensitive operation and description	Required role assignment
Creating a user in the user registry	`Editor@User Self Enrollment` or `Editor@Users` `Editor@User Self Enrollment` allows the user to add new users. You can modify other existing users with `Editor@Users` **Virtual resource:** *User Self Enrollment* is a virtual resource. *Users* is also a virtual resource.
Viewing the user profile information of a user *U*	`User@UG` and *U* is a member of user group *UG* or `User@Users` **Virtual resource:** *Users* is a virtual resource
Modifying the profile information of a user *U*	`Editor@UG` and *U* is a member of user group *UG* or `Editor@Users` **Virtual resource:** *Users* is a virtual resource.
Deleting a user from the user registry and deleting all private pages that are created by this user	`Manager@Users` **Virtual resource:** *Users* is a virtual resource.
Impersonating a user to troubleshoot problems and view pages, portlets, and other portal components.	`Can Run As User@Users` **Restriction:** To use the **Can Run As User** role, you must enable the impersonation feature and assign the **Can Run As User** role to an appropriate user.

Table 217. Access permissions for **Web Clipping**

Sensitive operation and description	Required role assignment
Creating new clippings	`Editor@Portlet Applications` **Virtual resource:** *Portlet Applications* is a virtual resource.

Table 218. Access permissions for web modules

Sensitive operation and description	Required role assignment
Installing a new portlet application WAR file	`Editor@Web Modules` **Virtual resource:** *Web Modules* is a virtual resource.
Updating a web module *WM* by installing a corresponding WAR file	`Editor@Web Modules` + `Manager@WM`
Uninstalling a web module and removing all corresponding portlet applications and portlets from all pages within the portal	`Manager@WM` + Manager @ all portlet applications that are contained in *WM*

Table 219. Access permissions for RESOURCE

Sensitive operation and description	Required role assignment
Adding a remote WSRP Producer *PR* to the Portal	`Editor@WSRP Producers` **Virtual resource:** *WSRP Producers* is a virtual resource.
Editing the settings of a remote Producer *PR*	`Editor@PR`
Viewing the settings or display the list of portlets that are provided by a remote WSRP Producer *PR*	`User@PR`
Deleting a remote WSRP Producer from the portal	`Manager@PR`

Table 220. Access permissions for XML Access

Sensitive operation and description	Required role assignment
Running commands with the XML configuration interface	`Security Administrator@Portal` + `Editor@XML Access` **Virtual resources:** *Portal* and *XML Access* are virtual resources.

Table 221. Access permissions for vanity URLs

Sensitive operation and description	Required role assignment
Creating, modifying, or deleting a vanity URL that points to page P	`Editor@P` and `Editor@VANITY_URL` **Virtual resources:** VANITY_URL is a virtual resource.

Note: If a user deletes a page, all vanity URLs that point to that page are also deleted, independent of the rights that the user has on the virtual resource VANITY_URL.

Overlay reports and site promotions

Table 222. Access permissions for overlay reports and site promotions

Resource	Sensitive operation and description	Required role assignment
Overlay reports	Can view overlay reports on a resource.	`User@OverlayReports` + `User@Resource` **Note:** `OVERLAY_REPORTS` is a virtual resource.
Overlay reports	Can view all existing site promotions.	`User@SitePromotions` **Note:** `SITE_PROMOTIONS` is a virtual resource.
Overlay reports	Can create a site promotion.	`Editor@SitePromotions`
Overlay reports	Can update an existing site promotion.	`Editor@SitePromotions`
Overlay reports	Can delete a site promotion.	`Editor@SitePromotions`
Overlay reports	Can add a site promotion assignment on specific resource.	`Editor@SitePromotions` + `User@Resource`
Overlay reports	Can view a site promotion assignment on specific resource.	`User@SitePromotions` + `User@Resource`
Site promotions	Can remove a site promotion assignment on specific resource.	`Editor@SitePromotions` + `User@Resource`

Role Mappings and WSRP services

On the WSRP producer side, you can set the configuration property **`wsrp.security.enabled`** to enforce the access control decision for the provided portlets. If this property value is set to `true`, then all access control decisions in the producing portal are based on the authenticated principal. If **`wsrp.security.enabled`** is set to `false`, then the producing portal does not enforce any access control on incoming client portal WSRP requests.

When you use identity propagation, the user who is authenticated on the client portal needs to have the required role assignments. If no identity propagation is configured, but SSL client certificate authentication is enabled, then the ID of the certificate needs to have the required role assignments. If no authentication method is used, then the request is treated as if it comes from the Anonymous Portal Users. In the latter case, the required roles need to be assigned to the Anonymous Portal User. This assignment implies allowing unauthenticated access to the corresponding resources for all users who can access the producer portal.

Related tasks:

"Securing a WSRP Producer portal" on page 1447
To secure provided portlets, you can configure the WSRP Producer for web service message security, for example, for message authentication. If you configure message authentication, you must also configure Portal Access Control.

"Configuring security on the Consumer portal" on page 1461
You can configure security for the WSRP Consumer. If you enable security, the WSRP Consumer sends a security token as part of the WSRP request message to the WSRP producer. The security token represents the identity of the user who is logged in to the Consumer Portal. The WSRP Producer uses the security token to process the WSRP requests under the user identity that is represented by the security token.

"Configuring Portal Access Control for a WSRP Producer portal" on page 1452
If you configure security for WSRP services, you must also configure Portal Access Control for the Producer.

"Adapting the list of required runtime configuration changes for your theme" on page 2819
You must adapt the list of required runtime configuration changes for your theme.

Initial Access Control Settings

The administrative user who installs IBM WebSphere Portal Express has a default set of access rights.

Portal administrative user access rights

The installation creates a dedicated administrative user and corresponding administrative user group. By default the name of the administrative user group is **wpsadmins**. The following information describes the initial access control settings. A user always has the permissions that are associated with the User, Editor, and Privileged User role types on itself. There is no explicit role assignment for these permissions. They are a part of the administration policy.

Administrative user
: The administrative user for each domain is identified during the installation and configured in the Access Control Data Management Service. The administrative user has unlimited access on all resource in the corresponding domain.

wpsadmins
: The administrative group for each domain is identified during the installation and configured in the Access Control Data Management Service. This group has the same role mappings as the administrative user.

All Authenticated Portal Users
: User@ the following portlet applications:
  - portletWiring Web Application
  - Edit page content and layout
  - Concrete Properties Web App
  - appearance Web Application
  - com.ibm.wps.portlets.palette
  - com.ibm.wps.wp.spa.portlets
  - com.ibm.wps.resolver.friendly.disambiguation
  - Application Layout
  - LotusNotesPortlet
  - Manage Pages and Favorites App
  - peopleFinderJSR168.1
  - People Finder
  - Organize Favorites
  - View Group Members

  User@ the following Virtual Resources:
  - USERS
  - USER_GROUPS
  - PSE_SOURCES

- APPLICATION_FOLDERS
- URL_MAPPING_CONTEXTS
- WCM_REST_SERVICE

Privileged User@ the following portlet applications:
- Welcome
- Information Portlet Application
- wp.ap.selfcare
- Bookmarks
- com.ibm.lotus.search.portlets.SearchCenterPortletApplication
- com.ibm.wps.portlets.bookmarks.9730c9c350.web2
- Directory Search
- ReminderPortlet
- Banner Ad
- LotusNotesPortlet
- Microsoft Exchange 2010
- Document_Viewer_Portlet
- IBM Common Mail Portlet
- Personalization Editors
- Personalization Navigator
- Personalization Picker
- Personalized List
- Policy Status Application
- Search Admin Application
- ParamConfig Application
- Properties Portlet Application
- Roles Portlet Application
- Community Portlet Application
- Application Catalog Manager
- wp.ap.sitemap

Privileged User@ the following portlets:
- Directory Search
- Lotus Notes View
- Sametime Web 2.0 Contact List

User@ the following portlets:
- Add to Sametime List
- Directory Search
- iNotes
- Dynamic Person Tag
- Lotus Notes View
- Sametime Web 2.0 Contact List

User@ the following pages:
- Content Root

Note: Propagation is blocked.

- Home
- Add to Sametime List
- Application Root
- Temporary Application Root
- Applications
- Application Membership
- Application Roles
- Banner Links
- Calendar
- Directory Search
- Mail
- Lotus Notes
- Sametime
- Organize Favorites
- Page Customizer
- Page Properties
- People Finder
- Policy Status
- Quick Links
- Collapsed Quick Links Shelf Links
- Expanded Quick Links Shelf Links
- Explore
- Footer Links
- Groups Viewer
- Person Tag
- Theme Links
- Template and Application Layout
- Template and Application Properties
- Welcome

Privileged User@ the following pages:

- Business Rules
- Collaboration
- Edit My Profile
- Search Seedlist
- Site Map
- Personalization
- Personalization Picker
- Personalized List
- Content Palette
- People Palette
- Search Center
- Document Search
- iNotes
- Home
- Quick Links

- Theme Links
- Site Map
- Messaging

Contributor@ the following document library:

- Portal Site library

  Note: Propagation is blocked.

Editor@ the following document library.

- Portal Site library: **`Content`** resource type

Anonymous Portal User

User@ the following portlet applications:

- wp.ap.login
- wp.ap.selfcare
- wp.ap.sitemap
- IBM Common Mail Portlet
- Newsgroups
- Banner Ad

User@ the following portlets:

- Lotus Notes View

User@ the following pages:

- Login
- Edit My Profile

User@ the following Virtual Resources:

- URL_MAPPING_CONTEXTS
- Editor@USER SELF ENROLLMENT

User@ the following document library:

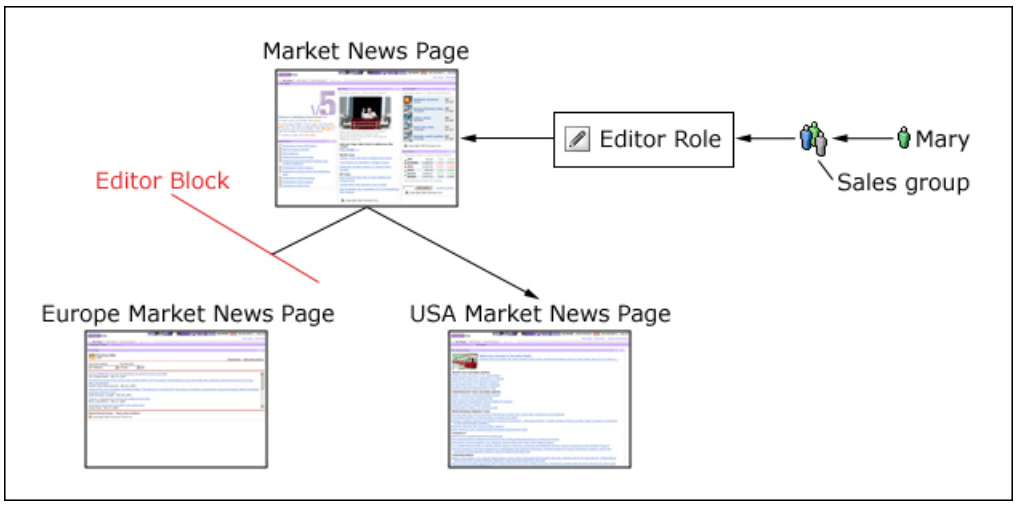
- Portal Site library

  Note: Propagation is blocked.
- Portal Site library: **`Content`** resource type

Access control scenarios

These scenarios provide helpful illustrations on how access control can be set up.

This section describes basic tasks for administering access control. The following examples use a hypothetical portal user called Mary and a hypothetical group called the Sales group. The following graphic helps illustrate the examples described in this topic.



Note: The tasks described here can be performed using either the administrative portlets or the XML configuration interface. For instructions about using the portlets mentioned here, refer to the portlet helps. For instructions about using the XML configuration interface, see the XML configuration interface section of the information center.

Give a user full access to the portal

Give the user the `Administrator@Portal` role. The `Administrator@Portal` role permits unrestricted access to all portal resources except the private pages of other users.

Give users this role in one of two ways:

- Add users to a group that has the `Administrator@Portal` role. The *wpsadmins* user group automatically receives the `Administrator@Portal` role during the portal installation. Use the `Manage Users and Groups` portlet to assign users to this group.
- Explicitly assign the `Administrator@Portal` role to specific users. Use the `Resource Permissions` portlet or the `User and Group Permissions` portlet to give users this role.

Allow a user to manage portlet applications with the Manage Applications portlet

Suppose that Mary needs to manage certain portlet applications. She must use the Manage Applications portlet to do this. Give Mary the following roles:

- User@*Web_Module*: This role permits Mary to see the information that is contained in a Web Module and to use the Manage Applications portlet to navigate to the Portlet Applications that are contained in this Web Module. You must assign Mary to a User role on each Web Module that she needs to access.
- Manager@*Portlet_Application*: This role permits Mary to administer the portlet application. You must assign Mary to a Manager role on each Portlet Application that she needs to administer.

There are two ways to give Mary these roles:

- Add Mary to a group that has these roles. Use the Manage Users and Groups portlet to assign her to this group.

- Explicitly assign the roles to Mary. Use the Resource Permissions portlet or the User and Group Permissions portlet to give her these roles.

Allow users to access a page and some subset of its child pages

Create an inheritance block on the appropriate page. For example, give the Sales group the `Editor@Market News Page` role. This allows members of the Sales group to edit the Market News page and all of its current and future child pages, including the Europe Market News page and the USA Market News page. To allow the Sales group to edit the USA Market News page, but not the Europe Market News page, insert an inheritance role block for the Editor role type on the Europe Market News Page. Use the Resource Permissions portlet or the XML configuration interface to insert this role block. This role block prevents members of the Sales group (and all other users and groups with an inherited or implicit Editor role on any parent pages of the Europe Market News page) from editing the Europe Market News page and all of its current and future child pages.

Allow users to access a portlet on a page

Give the group a role assignment on both the page and the portlet. Role assignments on a page do not contain access rights for portlets that appear on the page. Use the Resource Permissions portlet, the User and Group Permissions portlet, or the XML configuration interface to assign these roles.

For example, suppose there is a Market Targets portlet on the Market News Page. Give the Sales group (or a user group that contains the Sales group) the `Editor@Market Targets Portlet` role and the `Editor@Market News Page` role.

Allow users to access a page, but not its child pages

Use the Resource permissions portlet to create a propagation block on the appropriate page. For example, give the Sales group Editor access to the Market News page. To prevent this group from editing the USA Market News page and the Europe Market News page, create a propagation block for the Editor role type on the Market News page. It is not necessary to create a propagation block on the Market News child pages. This Market News page role block prevents the Sales group (and all other users and groups with an inherited or implicit `Editor@Market News Page` role) from editing all current and future child pages of the Market News Page.

Allow users to view and personalize a page and all of its child pages

Give the group the Privileged User role on the page and any portlets that appear on the page or its child pages. For example, give the Sales group to the `Privileged User@Market News Page` role. This allows all members of this group to view and personalize the Market News page and all of its current and future child pages. Then give the Sales group the Privileged User role on all portlets and portlet applications that appear on the Market News page and any of its child pages.

Giving the Sales group the Privileged User role instead of the Editor role allows members to create new private pages that are children of the Market News Page, but prevents members from creating new non-private pages.

The Editor role blocks that are created in the previous examples do not affect Privileged User roles in any way.

Allow a user to assign roles on a specific resource to members of a specific group

For example, to allow Mary to assign the Sales group to the role `Privileged User@Market News Page`, do either of the following steps:

- Give Mary the `Privileged User@Market News Page`, `Security Administrator@Market News Page`, and `Delegator@Sales Group` roles. This allows her to assign the Sales group (or individual members of this group) to the `Privileged User@Market News Page` role or the `User@Market News Page` role. Mary cannot assign anyone to the `Editor@Market News Page` role because she is not an Editor on the Market News Page. Mary cannot assign the Global Marketing group to the `Privileged User@Market News Page` role unless the Global Marketing group is a member of the Sales group.
- Give Mary the `Administrator@Portal` role. This allows her to assign any user or group to any role on any resource.

Note: To administer access control through the administrative portlets, Mary must have role assignments that allow her to view the User Group Permissions or the Resource Permissions portlets and the pages that contain these portlets. To administer access control through the XML configuration interface, Mary must have a role assignment that allows her to access the XmlAccess virtual resource.

Access control

You can restrict access to selected users and groups to the views within an authoring portlet, the items that are managed by the authoring portlet, and to elements and pages that are displayed within a website.

How access and security levels are set

There are three levels of access controls for web content

Library:
Library level access controls determine access to the library as a whole. If granted, it provides an entry point to the library. A user needs at least contributor access to a library to have access to it on the Authoring Portlet.

Item type per library:
Item Type access controls define the item type views and tasks a user can access within the authoring portlet for particular library. The permissions set for item types in a library do not automatically give you access to individual items. They give you access only to specific tasks and views within the authoring portlet.

Item level:
Item level access controls define the actions that a user can run on an individual item. For example, a Manager to the Components type has access to the Purge and Unlock actions but, if that user does not also have Manager access to an individual component, then the Purge and Unlock actions are not enabled when that component is selected.

"Users, Groups and Roles" on page 1557
Your content management system requires different types of users. You need to create a different group for each type of user and then assign those groups different roles within your system.

Users, Groups and Roles

Your content management system requires different types of users. You need to create a different group for each type of user and then assign those groups different roles within your system.

"Web content management roles"
You define the access of a user or group for a library to determine who has access to a library, and to define access to the different views within the authoring portlet.

"User roles and access" on page 1560
Different users will have different access to items and functions in your system depending on the role they are assigned. Roles can be assigned at the library level, and also assigned on individual items.

Related tasks:

"Setting up access" on page 1801
Access controls allow you to assign access to who has the ability to view rendered constant and pages, and who has the ability to edit or administer content, pages, or features.

Web content management roles:

You define the access of a user or group for a library to determine who has access to a library, and to define access to the different views within the authoring portlet.

Table 223. Roles

Roles	Rendering and authoring portlet access rights
• User	Users and groups that are assigned to this role can: • view items in a website or rendering portlet that they are assigned at least user access to. **Tip:** The simplest way to assign users to this role is to select any of the default user groups such as "All Authenticated Portal Users" or "Anonymous Portal User". Users require "user" access to an item before it is rendered in a website or rendering portlet.
• Reviewer	Users and groups that are assigned to this role can: • view items in a website or rendering portlet that they are assigned at least user access to. • run approve, next stage, and previous stage operations for workflowed items.
• Draft Creator	Users and groups that are assigned to this role can: • view items in a website or rendering portlet that they are assigned at least user access to. • access the create draft button if the user also has editor access. • access the restart workflow button if the user also has manager access.
• Contributor	Users and groups that are assigned to this role can: • view items in a rendering portlet or servlet-rendered website that they are assigned at least user access to. • view libraries that they are assigned contributor access to in an authoring portlet. • access the "My Items" and "All Items" views in an authoring portlet for libraries that they are assigned contributor access to. • access the item type view within the authoring portlet for item types that they are assigned at least user access to.

Table 223. Roles (continued)

Roles	Rendering and authoring portlet access rights
• Editor	Users and groups that are assigned to this role can: • view items in a rendering portlet or servlet-rendered website that they are assigned at least user access to. • view libraries that they are assigned contributor access to in an authoring portlet. • access the "My Items" and "All Items" views in an authoring portlet for libraries that they are assigned at least contributor access to. • for library item types that user and groups are assigned at least editor access to, editors can access the following actions in the authoring portlet: – access the item type view – create a new item – add/remove links – apply authoring template – copy – delete items they have created – edit – link to – move – restore a version – edit version labels
• Manager	Users and groups that are assigned to these roles can: • view items in a rendering portlet or servlet-rendered website that they are assigned at least user access to. • view libraries that they are assigned contributor access to in an authoring portlet. • access the "My Items" and "All Items" views in an authoring portlet for libraries that they are assigned at least contributor access to. For library item types that they are assigned manager access to, managers can access the all of the actions available to editors and also the following actions in the authoring portlet: • edit access settings • next stage • purge • unlock • edit user profile
• Administrator	Users and groups that are assigned to these roles can: • view items in a rendering portlet or servlet-rendered website that they are assigned at least user access to. • view libraries that they are assigned contributor access to in an authoring portlet. • access the "My Items" and "All Items" views in an authoring portlet for libraries that they are assigned at least contributor access to. • all actions in the authoring portlet for library item types that they are assigned administrator access to.

Table 223. Roles (continued)

Roles	Rendering and authoring portlet access rights
• Security Administrator • Privileged User • Markup Editor	These roles have no access to Web Content Manager items.

WebSphere Portal Administrators:

WebSphere Portal Express Administrators automatically have Administrator access to all item-types.

All Items view:

A user who is assigned access to an item can always view that item in the **All Items** view regardless of whether they have access to the related item-type view. For example, if a user does not have access to the presentation template view, but is granted editor access to a presentation template, they can still view, but not edit, the presentation template from the **All items** view.

Permission assignments for optimal performance

Inheriting permissions, instead of explicitly defining access rights on individual items, simplifies access rights maintenance and improves overall system performance. Define role assignments as high as possible in the library so that they apply to the largest set of items.

Explicitly defined permissions are a powerful and flexible way to control access to an item, but when the same user or group is being granted the same role on all items within an area, or across an entire library, then inherited permission is the best option

Assigning roles to anonymous or authenticated users

When accessing a website, users login as either anonymous users, or authenticated portal users.

The following pre-defined groups can be assigned roles in a library.

Table 224. pre-defined groups

Group	Details
Anonymous portal user	Select this user to assign a role to anonymous users.
All Authenticated Portal Users	Select this group to assign a role to users that have logged on to your server.
Users and User Groups	Select this group to assign a role to all users and groups.
All Portal User Groups	Select this group to assign a role to all groups.

User roles and access:

Different users will have different access to items and functions in your system depending on the role they are assigned. Roles can be assigned at the library level, and also assigned on individual items.

Assigning access to items

There are two methods that are used to assign roles to access controls on items:

- Selecting users or groups directly in the access section of an item.
- Allowing assigned roles to be inherited from parent items up to and including the library. Access roles are inherited in the following hierarchies:
  - Library/site area/content item
  - Library/taxonomy/category
  - Library/folder/component
  - Library/folder/authoring template
  - Library/folder/presentation template
  - Library/workflow
  - Library/workflow stage
  - Library/workflow action

  You can stop inheritance at any point in an inheritance hierarchy. For example, you might allow inheritance down to a site area, but assign access roles manually for each content item under that site area.

  Inheritance from a library is based on the role that is assigned to the overall library, not on the role that is assigned to specific item types. For example, you might not have access to the presentation template view on a library, but if you inherit the role of editor to a presentation template, you are able to view and edit that presentation template from the All Items view.

  Inheritance does not apply to draft items.

Note: By default, inheritance is enabled for all roles and items.

Viewing an item's security settings

The following sections are displayed on the security section of each item.

Table 225. Security settings

Section	Details
User Defined	If the item is not participating in a workflow, the user can edit access under user-defined.

Table 225. Security settings (continued)

Section	Details
Workflow	If an item is participating in a workflow, then the user-defined option does not appear and the workflow settings are displayed. This cannot be edited. Workflow-defined access is set in workflow stages. **Published items and workflow defined item security:** • If you grant a user editor access to an item in a workflow stage that uses a publish action, then those users are able to edit the published item directly. No draft is created. The same is true for administrator defined security when applied to published items. • If you grant a user manager access to an item in a workflow stage that uses a publish action, then those users are able to edit and delete the published item directly. No draft is created. The same is true for administrator defined security when applied to published items. • If you grant a user reviewer access to an item in a workflow stage that uses a publish action, then those users are able to create drafts of the published item.
Administrator Defined	Administrators can edit user access to an item at any time by changing the administrator defined settings.
Inheritance	You can also choose to inherit access that is assigned in the current web content library, or from an item's parent. Inheritance for all user roles is enabled by default.

How security is set

When a new item is created, the creator is automatically given manager access to the item. Extra user and group security can be added in the user-defined and system defined settings.

If an item is participating in a workflow, the creator is given manager access to the item only in the first workflow stage. As the item progresses through a workflow, the item security is determined by the combined workflow and system defined security.

Table 226. Security matrix

Security level	No workflow	First workflow stage	Extra workflow stages
User **Contributor** **Editor** **Manager** **Reviewer** **Draft Creator**	• User defined • Administrator defined • Inherited	• Administrator defined • Workflow defined	• Administrator defined • Workflow defined

Table 226. Security matrix (continued)

Security level	No workflow	First workflow stage	Extra workflow stages
Administrator	If you are assigned the administrator role to a library, you automatically inherit all administration access down to the item-level. It cannot be turned off.	If you are assigned the administrator role to a library, you automatically inherit all administration access down to the item-level. It cannot be turned off.	If you are assigned the administrator role to a library, you automatically inherit all administration access down to the item-level. It cannot be turned off.

Deleting items:

When a new item is created, the creator can also delete the item. If an item is participating in a workflow, the creator can delete the item in the first workflow stage only.

Assigning access to different types of users or groups

When you access a website or rendering portlet, users login as either anonymous users, or authenticated portal users.

The following user and groups can be used to grant access to items.

Table 227. Users and groups

User or group	Details
anonymous portal user	Select this user to grant access to anonymous users
[all users]	Select this group to grant access to all users, anonymous and authenticated.
[all authenticated portal users]	Select this group to grant access to all authenticated users.
[all portal user groups]	Select this group to grant access to all user groups.
[creator]	Select this group to grant access to the creator of the item.
[authors]	Select this group to grant access to users who are selected as an "author" of the item.
[owners]	Select this group to grant access to users who are selected as an "owner" of the item.

The access required to view a rendered item

To view an item on a rendered page, you need the following:

1. You need at least user access to the library the item is stored in.
2. You need at least user access to the presentation template used to display the current content item.
3. There must be a valid template map.
4. You must have sufficient access to view the item itself.

   Rendered item behavior varies depending on how you specify the `wcm.path.traversal.security` property in the `WCM WCMConfigService` service. If the property is not specified, the default value is `false`.

   If set to `false`:

   - Menus display content regardless of whether a user has access to all site areas in the content path.

- Navigators do not display site areas a user does not have access to, but can show content under these site areas in specific circumstances such as within breadcrumb navigators.
- URLs are only checked for content access, not site area access.

If set to `true`:

- Menus and navigators do not display content under secure site areas if the user does not have access to all site areas in the content path.
- Directly accessing content under secure site areas using a URL fails if the user does not have access to all site areas in the content path.

Rendering performance is slower if set to true.

Button access

You assign item-level access by assigning users and groups different roles for each item. The role that you assign determines what actions a user has access to for each item. The following table describes the minimum access that is required for access to each button in the user interface. If you enable inheritance at the library level, the library access level is inherited by item level access by default. For example, giving a user editor access to a library is automatically applied to new items they create if inheritance is enabled.

Table 228. Item access controls

Actions	Minimum item access	Minimum role access to library resources	Minimum library access	Item status
Add or move children	Contributor access or higher.	Editor access or higher to the library resource type.	Contributor access or higher.	N.A.
Add or remove child links	Contributor access or higher.	Editor access or higher to the library resource type.	Contributor access or higher.	N.A.
Add or remove workflows	Manager access or higher.	When first created, you require manager access to the library resource in any library. When saved, you require manager access to both the item and library resource in the library the item is stored in.	Contributor access or higher.	N.A.

Table 228. Item access controls (continued)

Actions	Minimum item access	Minimum role access to library resources	Minimum library access	Item status
Apply authoring template (authoring portlet)	N.A.	Manager access or higher to the authoring template library resource. **Note:** This default behavior can be changed to allow all users to apply authoring templates to items they have edit access to. See "Web content authoring options" on page 396 for further information.	Manager access or higher.	N.A.
Apply authoring template (content form)	Editor access or higher.	Contributor access or higher to the authoring template library resource.	Contributor access or higher.	N.A.
Approve	Reviewer or administrator.	Editor access or higher to the library resource type.	Contributor access or higher.	N.A.
Approve Project	Reviewer.	Not required.	Contributor access or higher.	N.A.
Batch-edit access controls	Editor access or higher.	Editor access or higher to the library resource type.	Contributor access or higher.	N.A.
Cancel draft	Manager access or higher.	Editor access or higher to the library resource type.	Contributor access or higher.	N.A.
Copy	Contributor access or higher.	Editor access or higher to the library resource type.	Contributor access or higher.	N.A.
Create draft	Draft Creator access.	Editor access or higher to the library resource type.	Contributor access or higher.	Only published or expired items.
Delete	Manager access or higher.	Editor access or higher to the library resource type.	Contributor access or higher.	N.A.

Table 228. Item access controls (continued)

Actions	Minimum item access	Minimum role access to library resources	Minimum library access	Item status
Edit	Editor access or higher.	Editor access or higher to the library resource type.	Contributor access or higher.	N.A.
Link to	Contributor access or higher, or Reviewer access.	Editor access or higher to the library resource type.	Contributor access or higher.	N.A.
Manage elements (In site areas and content items, not authoring templates.)	Administrator access If the **Allow elements to be managed by editors** option is selected on the authoring template that is used by an item, then this button is enabled for users with Editor access or higher. This option is enabled by default on Site Areas that are created by using the default Site Area template.	Editor access or higher to the library resource type.	Contributor access or higher.	N.A.
Move	Editor access or higher.	Editor access or higher to the library resource type.	Contributor access or higher.	N.A.
Next Stage	Reviewer access.	Editor access or higher to the library resource type.	Contributor access or higher.	N.A.
Preview item and view rendered item	User access or higher, or Reviewer access.	Not required.	Contributor access or higher.	N.A.
Previous Stage	Manager access or higher, or on workflow stages that have been configured to enable Reviewers access to the previous stage button.	Editor access or higher to the library resource type.	Contributor access or higher.	N.A.
Process now	N.A.	Not required.	Administrator access	N.A.

Table 228. Item access controls (continued)

Actions	Minimum item access	Minimum role access to library resources	Minimum library access	Item status
CF05 **Publish Project**	Editor access or higher.	Not required.	Not required.	Only when a project is in pending state.
Purge	Manager access or higher.	Not required.	Manager access or higher.	N.A.
Read	User access or higher, or Reviewer access.	Not required.	Contributor access or higher.	N.A.
Reference	User access or higher, or Reviewer access.	Not required.	Contributor access or higher.	N.A.
Reject	Reviewer or administrator access.	Editor access or higher to the library resource type.	Contributor access or higher.	N.A.
Reject Project	Reviewer.	Not required.	Contributor access or higher.	N.A.
Restart workflow	Draft Creator access.	Manager access or higher to the library resource type.	Contributor access or higher.	Only published or expired items.
Restore	Editor access or higher.	Editor access or higher to the library resource type.	Contributor access or higher.	N.A.
Save version	Editor access or higher.	Editor access or higher to the library resource type.	Contributor access or higher.	N.A.
Show hidden fields	N.A.	Not required.	Administrator access	N.A.
Submit for review(Workflows)	Reviewer access.	Editor access or higher to the library resource type.	Contributor access or higher.	N.A.
Submit for review (Projects)	Editor access or higher.	Editor access or higher to the library resource type.	Contributor access or higher.	Only when a project is in an active state.
System security	N.A.	Not required.	Administrator access	N.A.
Unlock	Manager access or higher.	Not required.	Manager access or higher.	N.A.
CF05 **Validate** (Projects)	User access or higher.	Not required.	Not required.	Only when a project is in active, review, pending, or publish failed states.

Table 228. Item access controls (continued)

Actions	Minimum item access	Minimum role access to library resources	Minimum library access	Item status
View references	User access or higher, or Reviewer access.	Not required.	Contributor access or higher.	N.A.
View versions	User access or higher, or Reviewer access.	Not required.	Contributor access or higher.	N.A.
Withdraw approval	Reviewer.	Not required.	Contributor access or higher.	Only when a project is in the review state. Only when Joint Approval is selected.
Withdraw from review	Reviewer.	Not required.	Contributor access or higher.	Only when a project is in the review state.

Creating new items:

The ability to create new items is set at the library level, not item level. You must have at least contributor access to a library and editor access to an item-type to create a new item. If you have access to create any item type, you can also create folders and projects.

Button access on content items:

You can choose to hide selected buttons on content item forms when you create an authoring template. This means that a user may not have access to all buttons on a content item form regardless of their role. Administrators can choose to display hidden buttons if required.

Profiling versus security:

Using profiling to personalize a site is different from using security to limit what items a user can access. In a profile-based personalized site, although a user might not be able to access all the pages by using personalized menus, they might still be able to access other pages by using navigators, or by searching for content. In a secured site, a user can only view items that they are granted access to.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Setting user and group permissions

The User and Group Permissions portlet lets you view and modify the roles that users and groups have on resources.

About this task

The User and Group Permissions portlet indicates whether roles are:

- Explicitly assigned
- Implicitly acquired through group membership
- Inherited through the resource hierarchy

Roles for virtual resources: Resource types have predefined sets of applicable roles. You can assign access to all roles, including the Can Run As User role, for the following virtual resources:

CONTENT_MAPPINGS
DESIGNER DEPLOY SERVICE
EVENT HANDLERS
MARKUPS
POLICY MAPPING CONTEXTS
PORTAL
PORTAL SETTINGS
RATINGS
STEP_UP_AUTHENTICATION
TAGS
TEMPLATE DEPLOYMENT
THEME MANAGEMENT
USER SELF ENROLLMENT
VP URL MAPPINGS
WSRP EXPORT
XML ACCESS

To access the User and Group Permissions portlet and find instructions for managing access rights for users and user groups, do the following:

Procedure

1. Log in to WebSphere Portal Express as an administrator.
2. Click the **Administration menu** icon ..
3. Select **Access** > **User and Group Permissions**. The User and Group Permissions portlet displays.
4. Click the portlet menu icon and select **Help** to access instructions for managing the access rights that users and user groups have for resources.

Setting resource permissions

Assign and control access for different types of resources.

The Resource Permissions portlet allows you to:

- Control whether or not a resource propagates or inherits its role assignments through the resource hierarchy
- Assign roles explicitly to specific users and groups
- Place resources under the control of an external security manager or bring back externalized resources under the control of IBM WebSphere Portal Express
- Create or delete roles on externalized resources

- View inherited role assignments for specific users and groups

See the Resource Permissions portlet help for detailed instructions on setting resource permissions.

Access the Resource Permissions portlet from the administration menu. Click the **Administration menu** icon. Then, click **Access** > **Resource Permissions**. From there, you select the resource type that you want to work with, an example is Pages. An **Assign Access** icon is available for each page listed.

Delegated Access Control Administration

IBM WebSphere Portal Express supports delegated access control administration.

Description of delegated administration and policy

This topic describes the delegation of access control administration and policy, and gives an example of the delegated administrative policy.

Delegated Administration

WebSphere Portal Express supports delegated access control administration. An administrator is a user who is authorized to modify the access control configuration by changing role assignments and creating or deleting role blocks. Administrators can delegate specific subsets of their administrative privileges to other users or groups. These users or groups can in turn delegate subsets of their privileges to additional users and groups. The delegated administration policy determines how users are permitted to delegate their privileges.

Delegated Administration Policy:

The delegated administration policy defines which role assignments are necessary in order to perform specific changes to the access control configuration.

The general policy for creating or deleting role assignments is as follows: A user *U* can create or delete a role assignment for a specific user or group *UG* to a role identified by role type *RT* and resource *R* in either of the following cases:

- All of the following criteria are met:
  - *U* has the Security Administrator@*R* or Administrator@*R* role
  - *U* has at least a role of type *RT* on *R*
  - *U* has the Delegator@*UG*, Security Administrator@*UG*, or Administrator@*UG* role.
- *U* has the Administrator@Portal or Security Administrator@Portal role

For example, in order to assign a group to a role of type Editor on a resource, you must have at least the Delegator@*Group* + Security_Administrator@*Resource* + Editor@*Resource* roles.

Note: The `Security Administrator@Portal` and `Administrator@Portal` roles allow users to make unrestricted changes to the access control configuration of resources that are under internal portal control. Users also need the `Administrator@External Access Control` role or the `Security Administrator@External Access Control` role in order to change the access control configuration for resources that are externally controlled by a security manager such as Security Access Manager.

The general policy for creating or deleting role blocks is as follows: A user *U* can create or delete a role block on a specific resource *R* and a role type *RT* in either of the following cases:

- If both of the following criteria are met:
  - *U* has the `Security Administrator@R` or `Administrator@R` role
  - *U* has at least a role of type *RT* on *R*
- **or** if *U* has the `Security Administrator@Portal` or `Administrator@Portal` role.

Example of the delegated administration policy

Mary needs the authority to delete Hans from the `Editor@Market News Page` role. Hans is a member of the Marketing group. She can do this if all of the following conditions are true:

- Mary is either `Security Administrator@Market News Page` or `Administrator@Market News Page`. She can acquire this role through an explicit role assignment, through an Administrator or Security Administrator role assignment on a parent resource, or by belonging to a group that has the appropriate role assignment.
- Mary is at least `Editor@Marketing News Page`, since Hans will be deleted from the Editor role type.
- Mary has a `Delegator@Marketing Group` role. Mary cannot delete arbitrary users or groups from the `Editor@Market News Page` role. She can delete only those users and groups for which she has a `Delegator` role. Because Hans is a member of the Marketing group, Mary has a `Delegator` role for Hans.

Access Control Caching

Access Control internally uses several caches to improve the access control decision times. You can improve access control performance for special scenarios by setting the lifetime and size properties of these caches in the Cache Manager Service. In most cases, WebSphere Portal Express will run smoothly with the default cache settings. However, if you have a large number of resources or a large number of customized resources, you may want to adjust cache settings and conduct some tests to find the best performance trade-offs.

Cache Invalidation Behavior

In most cases, all necessary access control caches are immediately invalidated when a change to the access control configuration is done such as assigning roles to a user. If using External authorization, it is not possible to permanently synchronize the caches with changes to externally managed roles. For users that already have an authenticated portal session when the change becomes valid, they must wait for a cache timeout or logout and re-login for the changes to become active for that user. Performing a login or logout always enforces the invalidation of all caches related to the current user.

There are three additional cases where the invalidation is not performed immediately and require the user to re-login or to wait for a cache timeout:

- If a role assignment is granted to or revoked from a user group, the change of permissions is not propagated immediately to those members of this group that are currently logged in
- If a role block is set or removed, the resulting change of permissions is not propagated immediately to those users currently logged in

- If nested groups are enabled and group A is added or removed from group B, the permission changes are not propagated immediately to those members of group A that are currently logged in

As in the external case, you can enforce a permissions refresh by performing a logout and login for the user. Alternatively, modify the following properties in the `wp_profile_root/PortalServer/config/CacheManagerService.properties` file to influence the caching behavior:

- To switch it off completely and allow immediate propagation of all permission changes, remove the comment tag from the **cacheinstance.com.ibm.wps.ac.AccessControlUserContextCache.enabled** property and set the value to `false`. This can have a considerable performance impact.
- To speed up permission refresh by timeout, the lifetime of this cache can be decreased by setting the **cacheinstance.com.ibm.wps.ac.AccessControlUserContextCache.lifetime** property (in seconds) to a smaller numerical value. These settings can affect performance.
- If you configure access control to use nested groups, disable **cacheinstance.com.ibm.wps.ac.groupmanagement.GroupCache** and modify the values of the **enabled** and **lifetime** properties for **cacheinstance.com.ibm.wps.ac.groupmanagement.NestedGroupCache**.

  If you use group cache, disable **cacheinstance.com.ibm.wps.ac.groupmanagement.NestedGroupCache** and modify the values of the **enabled** and **lifetime** properties for **cacheinstance.com.ibm.wps.ac.groupmanagement.GroupCache**. These settings can affect performance.
- All user and user group specific cache data can be explicitly invalidated upon user login authentication through the following two configuration settings through the WebSphere Integrated Solutions Console:

  Note: These two settings will cause high Dynamic Replication Service load in the application server in a cluster environment and therefore can affect performance.

  – Navigate to **Resource environment providers** > **WP PACGroupManagementService** > **Custom properties**. Either add or update **accessControlGroupManagement.invalidateGroupCacheOnLoginLogout** with a value of `true`.

  – Navigate to **Resource environment providers** > **WP AccessControlDataManagementService** > **Custom properties**. Either add or update **accessControlDataManagement.invalidateResourceCacheOnLoginLogout** with a value of `true`.

Note: After modifying the `CacheManagerService.properties` file, run the following task, from the *wp_profile_root*`/ConfigEngine` directory, to make the changes effective:

- Linux: `./ConfigEngine.sh update-properties -DWasPassword=`*`password`*
- IBM i: `ConfigEngine.sh update-properties -DWasPassword=`*`password`*
- Windows: `ConfigEngine.bat update-properties -DWasPassword=`*`password`*

Enabling Attribute Based Security

Attribute based security for Web Content Manager content is an access filter in the product filter chain. You can extend the access control permission checks for Web Content Manager content beyond the user or group-based decisions. You can define your own criteria. The criteria might involve categories, keywords, textComponents, htmlComponents, or shortTextComponents for an item.

Procedure

1. Place your attribute-based security implementation class in the *PortalServer_root*/shared/app directory.
2. Restart the WebSphere_Portal server.
3. Log in to the WebSphere Integrated Solutions Console.
4. Go to **Resources** > **Resource Environment** > **Resource Environment Providers**.
5. Go to **WP AccessControlDataManagementService** > **Custom properties**.
6. Change the **accessControlDataManagement.enableAttributeBasedSecurityFilter** property to true.
7. Change the **accessControlDataManagement.AccessControlAttributeBasedSecurityImpl** property to the value of the security implementation class in the *PortalServer_root*/shared/app directory. For example, enter com.ibm.portal.ac.AccessControlAttributeBasedSecurityImpl.
8. Save your changes.
9. Restart the WebSphere_Portal server.

Java 2 security with WebSphere Portal Express

Java 2 (J2SE) security provides a policy-based, fine-grain access control mechanism that increases overall system integrity by checking for permissions before allowing access to certain protected system resources. J2SE security allows you to set up individual policy files that control the privileges assigned to individual code sources. If the code does not have the required permissions and still tries to execute a protected operation, the Java Access Controller will throw a corresponding security exception.

Policy files assign individual permissions to individual code sources. The syntax and semantics of the policy files are defined in the Java Language Specification. WebSphere Application Server uses a specific set of policy files to set up Java 2 Security. The following table contains information on the policy files and their protection scope:

Table 229. Protection scope for policy files

Default location and policy file	Protection scope
AppServer_root//java/jre/lib/security/java.policy	This is the root policy file that contains permissions for all the processes launched by WebSphere Application Server.
wp_profile_root/properties/server.policy	This policy file grants default permissions to all product servers.
wp_profile_root/properties/client.policy	This policy file grants default permissions for all of the product client containers and applets on a node.

Table 229. Protection scope for policy files (continued)

Default location and policy file	Protection scope
wp_profile_root/config/cells/*cell_name*/nodes/*node_name*/spi.policy	This template is for the Service Provider Interface (SPI) or the third party resources that are embedded in the product. The default permission is java.security.AllPermissions.
wp_profile_root/config/cells/*cell_name*/nodes/*node_name*/library.policy	This policy grants default permissions (empty) to code contained in the shared library (Java library classes) to use in multiple product applications.
wp_profile_root/config/cells/*cell_name*/nodes/*node_name*/app.policy	This policy grants default permissions to all enterprise applications running on this node in this cell.
wp_profile_root/config/cells/*cell_name*/applications/*ear_file_name*/deployments/*application_name*/META-INF/was.policy	This policy assigns permissions to a specific enterprise application, imbedded within *EAR*:/META-INF/was.policy.
rar_filename/META-INF/was.policy.RAR	This file can have a permission specification that is defined in the ra.xml file. The ra.xml file is embedded in the RAR file.

All code artifacts, installed with the WebSphere Portal Express product, run with **java.security.AllPermission** specified either in the server.policy file for the portal shared libraries or in the individual was.policy files for the individual portlets.

Portlets that are installed on WebSphere Portal Express after installation can bring along their own was.policy files defining the allowed interactions of the portlet code with the system resources; see Portlet concepts for additional information.

Note: The application server searches for was.policy files in the enterprise application archive rather than the Web application archive comprising a portlet. Therefore, the portal server copies was.policy from the appname.war/META-INF directory to the generated appname.ear/META-INF directory during deployment of a portlet WAR file.

Related concepts:

"Portlet concepts" on page 2932
Learn about portlets from a user's and an application developer's perspective. View a brief comparison between a portlet and a servlet and understand basic portlet concepts; know the effect of Java 2 security enablement on the operation of portlets that rely on certain privileges for processing.

Related information:

Java 2 security

Java 2 Platform Security

Integrating with OpenID authentication

Web applications provide information and services to public users and personalized information and services to authenticated users. Users often work with multiple web applications, which require multiple IDs and passwords. This requirement can be difficult to maintain. Integrating identity providers (Google, Yahoo, or Facebook) into your site can simplify logging in for your users.

About this task

Google, Yahoo, Facebook, and other web platforms host information for users and they also provide access to their existing user communities. Reusing these communities on your website can increase acceptance of your business or services.

There are multiple approaches to creating a relationship between an identity provider and a service provider (IBM WebSphere Portal Express). WebSphere Portal Express uses OpenID and OAuth to integrate a relationship to an identity provider. OpenID provides a method of decentralized user management where users can select an identity provider to host their profile information, including user ID and password. Google and Yahoo are known identity providers that use OpenID specifications. Facebook uses OAuth.

WebSphere Portal Express requires that a trusted relationship exists between the identity provider and IBM WebSphere Application Server. Therefore, WebSphere Application Server provides a plug-in point, called a trust association interceptor (TAI), designed to create a trust based on the identity provider information.

WebSphere Portal Express provides a new implementation of this plug-in point that handles the communication between the identity provider and WebSphere Portal Express as the service provider. WebSphere Portal Express trusts the identity provider and grants the user entrance.

There are two options to integrate external users into the WebSphere Portal Express environment:

- You can require an existing binding between a local portal account and a remote identity provider account. This option provides you with the possibility to request additional validation from the users and to have internal accounts for the users. The binding is stored in a user attribute, which requires a writable user repository.
- You can give all users of an identity provider account access to your portal environment as an identified user. If you grant special access rights to these users, they do not need to register individually with WebSphere Portal Express. This option requires fewer steps for your business users.

Complete the following tasks to configure the identity providers that are appropriate for your business requirements.

"Configuring OpenID authentication" on page 1575
Identity providers include sites such as, but not limited to, Google, Yahoo, and Facebook. As an Administrator, you can select how to configure authentication to work with these identity providers.

"Modifying the list of OpenID providers" on page 1582
You can change the list of identity providers that your users can access. You can add or remove providers from the list. You can change the order that the identity providers display in the **Login** and **Profile Management** portlet user interfaces.

"Configuring transient users" on page 1583
In addition to the basic OpenID authentication option, you can give users, who are trusted and verified from an identity provider, access to IBM WebSphere Portal Express. These trusted and verified users do not require a local, registered Portal user account.

"Disabling transient users and OpenID authentication" on page 1586
After using the transient users and OpenID authentication, you might decide you want to stop using the function. You can permanently or temporarily disable the transient users function or the full OpenID authentication function.

Configuring OpenID authentication

Identity providers include sites such as, but not limited to, Google, Yahoo, and Facebook. As an Administrator, you can select how to configure authentication to work with these identity providers.

Before you begin

Set your system time to match your local time.

About this task

Users can always register and log in with WebSphere Portal Express credentials. With the **enable-identityprovider-tai** task, you can choose between the following configurations:

Remember: Google and Yahoo are known identity providers that use OpenID specifications. Facebook uses OAuth.

Limitation: If you configure portal to use OpenID authentication with YAHOO as the identity provider, attribute exchange does not work. This limitation means that user attributes from your YAHOO OpenID account are not transferred to your portal user account.

- Configure Facebook only, use the parameters that are designated as Facebook
- Configure OpenID only, use the parameters that are designated as OpenID
- Configure Facebook and OpenID, use the parameters that are designated as Facebook and OpenID

Procedure

1. If you plan to configure Facebook, register your WebSphere Portal Express server instances as Facebook applications. You must register two applications. One application is the protected path: `/wps/myportal`. One application is the public, unprotected path: `/wps/portal`.

   After you register, Facebook provides you with an application ID and an application secret. Use this information when you run the **enable-identityprovider-tai** task.

   Tip: Your Facebook application has a private and public URL. The private URL is `http://`*`yourserver`*`:`*`yourport`*`/wps/`*`myportal`*`/`. The public URL is `http://`*`yourserver`*`:`*`yourport`*`/wps/`*`portal`*`/`.
2. Required: Run the following task from the *wp_profile_root*`\ConfigEngine` directory with the appropriate parameters:

Cluster note: Complete this step only on the primary node.
CAUTION:
If you rerun the `enable-identityprovider-tai` task, the task sets new properties and does not preserve the old configuration data. If you want to keep the existing data, you must add the new values to the existing values before you rerun the task.

- IBM i: `ConfigEngine.sh enable-identityprovider-tai -DWasUserId=`*`username`* `-DWasPassword=`*`password`*
- Linux: `./ConfigEngine.sh enable-identityprovider-tai -DWasUserId=`*`username`* `-DWasPassword=`*`password`*
- Windows: `ConfigEngine.bat enable-identityprovider-tai -DWasUserId=`*`username`* `-DWasPassword=`*`password`*

Add the following parameters to customize the task for your business requirements:

`-Didp.providerlist`
Set the value to `facebook,openid` if you want to configure both options. Set the value to `facebook` to configure Facebook only. Set the value to `openid` to configure only identity providers that use the OpenID specifications. If you leave this field blank, the default value is `facebook`.

`-Dfacebook_apps`
If you are configuring Facebook, you can use Facebook for authentication (`app`), self enrollment (`pub`), or both (`app,pub`). The default value is `app`. Set the value to one of the following values:

`app` Configures authentication only.

`pub` Configures self enrollment only.

`app,pub`
Configures both authentication and self enrollment.

`-Dfacebook_app_id`
If you are configuring Facebook for authentication, set the value to *`yourprivatefacebookappid`*, which you received when you registered your Portal server as a private Facebook application. This value is for your private URL.

`-Dfacebook_app_secret`
If you are configuring Facebook for authentication, set the value to *`yourprivatefacebookappsecret`*, which you received when you registered your Portal server as a private Facebook application. This value is for your private URL.

`-Dfacebook_app_site`
If you are configuring Facebook for authentication, set the value to `http://`*`yourserver`*`:`*`yourport`*`/wps/`*`myportal`*`/`. This value is the URL for your private WebSphere Portal Express server that Facebook uses after a successful authentication. A protected area requires authentication to access and is not available to an anonymous user.

`-Dfacebook_pub_id`
If you are configuring Facebook for self enrollment, set the value to *`yourpublicfacebookappid`*, which you received when you registered your Portal server as a Facebook application. This value is for your public URL.

-Dfacebook_pub_secret
If you are configuring Facebook for self enrollment, set the value to *yourpublicfacebookappsecret*, which you received when you registered your Portal server as a Facebook application. This value is for your public URL.

-Dfacebook_pub_site
If you are configuring Facebook for self enrollment, set the value to `http://yourserver:yourport/wps/portal/`. This value is the URL for your public WebSphere Portal Express server that Facebook uses after a successful authentication. A public area does not require authentication to access and is available to an anonymous user.

-Dopenid.servicenames
If you are configuring identity providers that use the OpenID specifications, enter a comma-separated list of the identity providers that you want configured; for example: `Google,Yahoo`.

-Dopenid.servicenames.endpoints
If you are configuring identity providers that use the OpenID specifications, enter a comma-separated list of OpenID endpoints (access addresses). These endpoints are for the identity providers in the **openid.servicenames** parameter. For example, type `https://www.google.com/accounts/o8/id,https://me.yahoo.com/`. There must be a one-to-one correspondence between the **openid.servicenames** and the **openid.servicenames.endpoints** parameters. If you entered three identity providers in the **openid.servicenames** parameter, you must enter three endpoints in the **openid.servicenames.endpoints** parameter and in the same sequence.

-Dprovider.openid.nonce_valid_time
If you are configuring identity providers that use the OpenID specifications, enter a value in seconds to protect old communications from being reused in replay attacks. If this parameter is not set, you might have nonce errors in your `SystemOut.log` file.

3. Required: Complete the following steps to configure the Profile Management and Login portlets:

   Cluster note: Complete these steps on one node in the cluster.

   a. Log on to WebSphere Portal Express as the administrator.
   b. Click the **Administration menu** icon. Then, click **Portlet Management** > **Portlets**.
   c. Locate the Login portlet and click the **Configure portlet** icon.
   d. Configure the Login portlet with the following parameters:

      Tip: During authentication, WebSphere Portal Express server retrieves attributes from the Identity Provider. Custom parameters, such as languages preferences, are not automatically retrieved. You must add these parameters to Portal. If the parameter does not exist, enter the parameter name in **New Preference** and the parameter value in **New value**. Then, click **Add** to add the new parameter to the Login portlet.

      show_idp_option
      Set this required parameter to `true` to show the identity provider authentication feature on the portlet.

      show_idp_max
      Set this required parameter to the maximum number of identity

providers that are shown on the portlet. You define the list of providers when you run the **enable-identityprovider-tai** task. If you defined five identity providers and want two to show on the portlet, set this parameter to 2. On the portlet, two identity providers are shown. Click **More** to show the complete list of identity providers.

show_idp_freeform_field
Set this required parameter to `true` to use the full OpenID string and not restrict it to certain known services. This option shows a free-form field on the portlet. If set, users can enter any OpenID identifier.

***providername*.image**
providername represents the case-sensitive name of the identity provider. For example, you would create the **Google.image** parameter. Set this optional parameter to define an image for the configured identity provider buttons. You can define whether a text button or an image is shown. Enter the URL of the identity provider image.

e. Click **OK** to save your changes.

f. Locate the Profile Management portlet and click the **Configure portlet** icon.

g. Configure the Profile Management portlet with the following parameters:

Tip: If the parameter does not exist, enter the parameter name in the **New Preference** field and the parameter value in the **New value** field. Then, click **Add** to add the new parameter to the Profile Management portlet.

show_idp_option
Set this required parameter to `true` to show the identity provider authentication feature on the portlet.

show_idp_max
Set this required parameter to the maximum number of identity providers that are shown on the portlet. You define the list of providers when you run the **enable-identityprovider-tai** task. If you defined five identity providers and want two to show on the portlet, set this parameter to 2. On the portlet, two identity providers are shown. Click **More** to show the complete list of identity providers.

show_idp_freeform_field
Set this required parameter to `true` to use the full OpenID string and not restrict it to certain known services. This option shows a free-form field on the portlet. If set, users can enter any OpenID identifier.

***providername*.image**
providername represents the case-sensitive name of the identity provider. For example, you would create the **Google.image** parameter. Set this optional parameter to define an image for the configured identity provider buttons. You can define whether a text button or an image is shown. Enter the URL of the identity provider image.

***providername*.required**
providername represents the case-sensitive name of the identity provider service name. For example, you would create the **Google.required** parameter. Set this optional parameter to define

the attribute mappings you want required between the identity provider and the Profile Management portlet. Enter a semicolon-separated list of attribute mapping pairs that are combined with a vertical bar (|); for example, *attributename*|*openidattribute*. You must create a parameter for each supported identity provider; for example: **Google.required** and **aol.required**. Check the schema documentation of each identity provider for the supported attributes. Some mapping examples include:

Google: all in one line

```
ibm-primaryEmail|http://axschema.org/contact/email;
preferredLanguage|http://axschema.org/pref/language;
givenName|http://axschema.org/namePerson/first;
sn|http://axschema.org/namePerson/last
```

Facebook: all in one line

```
ibm-primaryEmail|email;
givenName|first_name;
sn|last_name;
uid|id;
preferredLanguage|locale
```

***providername*.optional**

providername represents the case-sensitive name of the identity provider. For example, you would create the **Google.optional** parameter. Set this parameter to define the attribute mappings you want optional between the identity provider and the Profile Management portlet. Enter a semicolon-separated list of attribute mapping pairs that are combined with a vertical bar (|). You can create a parameter for each supported identity provider; for example: **Google.optional** and **aol.optional**. Check the schema documentation of each identity provider for the supported attributes. Some mapping examples include:

Google: all in one line

```
ibm-primaryEmail|http://axschema.org/contact/email;
preferredLanguage|http://axschema.org/pref/language;
givenName|http://axschema.org/namePerson/first;
sn|http://axschema.org/namePerson/last
```

Facebook: all in one line

```
ibm-primaryEmail|email;
givenName|first_name;
sn|last_name;
uid|id;
preferredLanguage|locale
```

***providername*.protocol**

providername represents the case-sensitive name of the identity

provider. Set this required parameter to define the Identity Provider Attribute Exchange protocol. Simple Registration (SREG) and Attribute Exchange (AX) are supported. The supported values for the parameters are `openid.sreg` for SREG or `openid.ax` for AX. You must create a parameter for each supported identity provider service name; for example: **`Google.protocol`** and **`aol.protocol`**.

facebook.required
Set this parameter to define required attribute mappings between Facebook and the **Profile Management** portlet. Enter a semicolon separated list of attribute mapping pairs that are combined with a vertical bar (`|`).

Some mapping examples include: *all in one line*

```
attributename|facebookattribute;
attribute2|facebookattribute2
```

The following item is a mapping example: *all in one line*

```
uid|id;ibm-primaryEmail|email;
givenName|first_name;sn|last_name;
preferredLanguage|locale
```

h. Click **OK** to save your changes.

4. Verify that the following `.jar` files were copied to the *AppServer_root*`\lib\ext` directory:

   Cluster note: Complete this step on each node in the cluster.

   PortalServer_root`\prereqs.infra\prereq.commons.httpclient\lib\ext\commons-codec-1.6.jar`

   PortalServer_root`\prereqs.infra\prereq.commons.httpclient\lib\ext\commons-httpclient-3.0.1.jar`

5. Required: Complete the following steps to add SSL certificates for the configured identity providers; some providers require multiple certificates:

   Attention: If an identity provider uses multiple server endpoints that require different SSL certificates, you might receive error message EJPAK0062E.

   Cluster note: In a clustered environment, you must complete these steps only on the Deployment Manager.

   Farm note: In a farm environment, you must complete these steps on each server in your farm.

   a. Log on to the WebSphere Integrated Solutions Console.
   b. Go to **Security** > **SSL certificate and key management**.
   c. Under **Configuration settings**, click **Manage endpoint security configurations**.
   d. Under **Outbound** > ***hostname*** > **nodes** > ***node_name*** > **servers**, click the **WebSphere_Portal** server option.
   e. Under **Related Items**, click **Key stores and certificates**.
   f. Click **NodeDefaultTrustStore**.

      Cluster note: Click **`CellDefaultTrustStore`** instead of **`NodeDefaultTrustStore`**.

   g. Under **Additional Properties**, click **Signer certificates**.
   h. Click **Retrieve from port**.

i. Enter the following information and then click **Retrieve signer information**:

Host Enter the endpoint for the identity provider without specifying the protocol, for example, `http://` or `https://`. Type `www.google.com` for Google or `graph.facebook.com` for Facebook.

Port Enter the port number for the identity provider, for example, type `443`.

Alias Enter the certificate alias name, which is specified in the SSL configuration; for example type `graph.facebook.com_cert` for Facebook.

j. Verify the **Retrieved signer information** and then click **Apply**.

k. Click **Save**.

l. If you receive error message EJPAK0062E, you might be missing a certificate. Open the `SystemOut.log` file and search for CWPKI0022E: SSL HANDSHAKE FAILURE. If this error message is present, import the certificate where the **domainname** is part of the **SubjectDN**.

6. Required: Complete the following steps to stop and restart the WebSphere_Portal server:

   Cluster note: Recycle the server or cluster instance.

   a. Open a command prompt and change to the following directory:

      IBM i: *wp_profile_root*`/bin`

      Linux: *wp_profile_root*`/bin`

      Windows: *wp_profile_root*`\bin`

   b. Enter the following command to stop the *WebSphere_Portal* server, where *WebSphere_Portal* is the name of the WebSphere Portal Express server:

      IBM i: `stopServer` *`WebSphere_Portal`* `-username` *`admin_userid`* `-password` *`admin_password`*

      Linux: `./stopServer.sh` *`WebSphere_Portal`* `-username` *`admin_userid`* `-password` *`admin_password`*

      Windows: `stopServer.bat` *`WebSphere_Portal`* `-username` *`admin_userid`* `-password` *`admin_password`*

   c. Enter the following command to start the *WebSphere_Portal* server, where *WebSphere_Portal* is the name of the WebSphere Portal Express server:

      IBM i: `startServer` *`WebSphere_Portal`*

      Linux: `./startServer.sh` *`WebSphere_Portal`*

      Windows: `startServer.bat` *`WebSphere_Portal`*

7. Required: Complete the following steps in a clustered environment to configure the **OpenidObjCache** cache instance:

   a. Log on to the WebSphere Integrated Solutions Console.

   b. Go to **Resources** > **Cache instances** > **Object cache instances**.

   c. Select **OpenidObjCache**.

   d. Under the **Consistency settings** section, set the following values:

      Select the **Enable cache replication** check box.

      Select `Both Push and Pull` for the **Replication type**.

   e. Click **OK**.

   f. Click **Save**.

   g. Stop and restart the cluster servers to propagate the changes.

8. Optional: Depending on the security settings for your identity providers, you must modify attributes for your identity provider trust association. Complete the following steps to modify your trust association:

   Cluster note: Complete these steps on one node in the cluster.

   a. Log on to the WebSphere Integrated Solutions Console.
   b. Go to **Security** > **Global security** > **Web and SIP security** > **Trust association**.
   c. Select **Interceptors** and then select **com.ibm.portal.auth.OpenIDTAI**.
   d. Add or modify properties to change the default behavior. For example, you can add or modify the following properties:

      bindattribute
      This property stores the user profile attribute that contains the identity provider user ID. The default value is `labeledURI`.

      loginattribute
      This property defines the attribute that is retrieved from the repository that uniquely identifies the user. The default value is `uid`.

Results

New users can register a new WebSphere Portal Express profile with a valid identity provider specified in **labeledURI**. Existing users can update their profile with a valid identity provider specified in **labeledURI**. To support the profile update, a writable user repository must exist. They can then log on to WebSphere Portal Express with the alternative login field. They are redirected to the identity provider login page.

Related tasks:

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

Related information:

Response Format

Federated Login for Google Account Users

User

Modifying the list of OpenID providers

You can change the list of identity providers that your users can access. You can add or remove providers from the list. You can change the order that the identity providers display in the **Login** and **Profile Management** portlet user interfaces.

Procedure

1. Log on to the WebSphere Integrated Solutions Console.
2. Go to **Security** > **Global security** > **Web and SIP security** > **Trust association**.
3. Select **Interceptors** and then select **com.ibm.portal.auth.OpenIDTAI**.
4. Modify the following properties that are based on your business requirements:

openid.servicenames
This property defines a comma-separated list of your identity providers that your users see. For example, type `Google,Yahoo`. The order of the names affects the user interface and the order that the providers are displayed.

provider.openid.servicenames.endpoints
When you configure identity providers that use the OpenID specifications, enter a comma-separated list of OpenID endpoints (access addresses) for the identity providers that you entered in the **openid.servicenames** parameter. For example, type `https://www.google.com/accounts/o8/id,https://me.yahoo.com/`. There must be a one-to-one correspondence between the **openid.servicenames** and the **openid.servicenames.endpoints** parameters. If you entered three identity providers in the **openid.servicenames** parameter, you must enter three endpoints in the **openid.servicenames.endpoints** parameter and in the same sequence.

5. Restart the WebSphere_Portal server.

Related tasks:

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

Configuring transient users

In addition to the basic OpenID authentication option, you can give users, who are trusted and verified from an identity provider, access to IBM WebSphere Portal Express. These trusted and verified users do not require a local, registered Portal user account.

About this task

Facebook and Google users can authenticate with the WebSphere Portal Express server instance with their identity provider credentials. They are granted access to certain data within WebSphere Portal Express without having a local account. You can grant the same access to all identity providers or you can configure different access rights that are based on the identity provider. With this option you, can provide a personalized view to unregistered users while still providing benefits to fully registered users.

Procedure

1. Run the following task from the *wp_profile_root*`\ConfigEngine` directory with the appropriate parameters:

   Cluster note: Complete this step only on the primary node.

   - IBM i: `ConfigEngine.sh enable-transient-user -DWasUserId=`*`username`* `-DWasPassword=`*`password`*
   - Linux: `./ConfigEngine.sh enable-transient-user -DWasUserId=`*`username`* `-DWasPassword=`*`password`*
   - Windows: `ConfigEngine.bat enable-transient-user -DWasUserId=`*`username`* `-DWasPassword=`*`password`*

   Add the following parameters to customize the task for your business requirements:

-Dtransparent.suffix
Set this value to a **dn** suffix that is used for transient users. The suffix must NOT match your current suffixes for fully registered users. The default value is `o=transparent`.

-Dtransparent.prefix
Set this value to a prefix that is used for transient users. For example, if you want to set the RDN attribute, set this value to `cn`.

Note: Complete the following steps if you entered the wrong value in the **transparent.suffix** parameter:

a. Log on to WebSphere Integrated Solutions Console as the administrator.
b. Go to **Security** > **Global Security**.
c. Go to **User account repository** > **Available realm definitions** and select **Federatedrepositories**.
d. Click **Configure**.
e. Go to **Repositories in the realm** and click the link in the Base Entry column for the **transientidp** repository identifier, for example, **o=transparent**.
f. Replace the value in the following fields with the new value:

Distinguished name of a base entry that uniquely identifies this set of entries in the realm For example, `o=transparent`.

Distinguished name of a base entry in this repository For example, `o=transparent`

g. Click **OK**.
h. Save your changes.
i. Stop and restart the WebSphere_Portal server.

2. Optional: Complete the following steps to create group objects for external providers to assign different access rights:

Important: After you run the **enable-transient-user** task, all identified users are identified with the all authenticated group and do not have explicit groups.

a. Log on to WebSphere Integrated Solutions Console as the administrator.
b. Go to **Security** > **Global Security**.
c. Go to **User account repository** > **Available realm definitions** and select **Federatedrepositories**.
d. Click **Configure**.
e. Go to **Repositories in the realm** and click **transientidp** in the **Repository Identifier** column.
f. Click **New** and add the following information:
   - **Name**: `buildgroupsfor`
   - **Value**: Enter the list of supported Identity Providers that you want to build groups for; for example: `facebook Google`. The items in the list must be separated by a space. The Identity Providers are case-sensitive and must match what you entered for the **idp.providerlist** and **openid.servicenames** parameters.
g. Click **OK**.
h. Save your changes.
i. Stop and restart the WebSphere_Portal server.

3. Optional: Complete the following steps to mark transient identity provider users as external:

Information: After you run the **enable-transient-user** task, the system builds internal groups for each identity provider. You can use these groups in the Resource Permissions portlet in the **Portal Administration** menu. Use the Resource Permissions portlet to build a set of pages and portlets that transient users can see and use.

You can also combine transient users with the external user feature in WebSphere Portal Express. You can identify a group of external or transient users with a database suffix. All external and transient users are then granted a special virtual principle in the access control. Use this virtual principle to grant a general set of access rights to these users.

a. Log on to WebSphere Integrated Solutions Console as the administrator.
b. Go to **Resources** > **Resource Environment** > **Resource Environment providers**.
c. Search for `WP PumaStoreService` and then click **Custom properties**.
d. Add the `parentDN.externalUsers` property with value you entered for **transparent.suffix**. If you did not enter a value in **transparent.suffix**, type `o=transparent`.
e. Save your changes.
f. Stop and restart the WebSphere_Portal server.

4. Complete the following steps to load user attributes during authentication:

Note: Transient users do not have attributes that are stored locally. Therefore, it is helpful to load attributes from the Identity Provider during authentication.

Note: If you want to allow transient users to create or modify pages, you must map a short name to the users. The attribute that is used for the short name is the User default search attribute. If you do not know the attribute name, you can find it defined in the PumaStoreService Resource Environment provider. The most common values are `uid` and `cn`.

a. Log on to WebSphere Integrated Solutions Console as the administrator.
b. Go to **Security** > **Global security** > **Web and SIP Security** > **Trust association** > **Interceptors**.
c. Select **com.ibm.portal.auth.tai.OpenidTAI**.
d. Add the following new properties for OpenID:
   - `provider.openid.loadattributes=provider|`*`method`*`;provider2|`*`method`*

     Note: *method* can either be `openid.sreg` or `openid.ax` depending on the type of OpenID your Identity Provider supports.
   - The following properties must be entered as one line.

     ```
     provider.openid.loadattributes.provider=portalattributename|
     idpattributename;portalattributename2|idpattributename2
     ```
   - The following properties must be entered as one line.

     ```
     provider.openid.loadattributes.provider2=portalattributename|
     idpattributename;portalattributename2|idpattributename2
     ```

   For example, you might add the following new properties for OpenID:
   - `provider.openid.loadattributes=google|openid.ax;yahoo|openid.ax`
   - The following properties must be entered as one line.

     ```
     provider.openid.loadattributes.google=cn|
     http://axschema.org/namePerson/first;sn|
     ```

```
http://axschema.org/namePerson/last;ibm-primaryEmail|
http://axschema.org/contact/email
```

e. Add the following new property for Facebook:

- The following properties must be entered as one line.

```
provider.facebook.loadattributes=portalattributename|
idpattributename;portalattributename2|idpattributename2
```

For example, you might add the following new property for Facebook:

- The following properties must be entered as one line.

```
provider.facebook.loadattributes=sn|
first_name;cn|last_name;uid|name
```

f. Save your changes.

g. Stop and restart the WebSphere_Portal server.

Related tasks:

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

Disabling transient users and OpenID authentication

After using the transient users and OpenID authentication, you might decide you want to stop using the function. You can permanently or temporarily disable the transient users function or the full OpenID authentication function.

About this task

Complete the following steps to disable transient users and the OpenID authentication.

Procedure

1. Complete the following steps to disable transient users:
   a. Log on to WebSphere Integrated Solutions Console as the administrator.
   b. Go to **Security** > **Global security** > **Web and SIP Security** > **Trust association**.
   c. Select **com.ibm.portal.auth.tai.OpenIdTAI**.
   d. Select **Custom properties**.
   e. Change the value of the **create_user_during_logon** property to `false`.
   f. Save your changes.
   g. Stop and restart the WebSphere_Portal server.
2. Complete the following steps to disable the OpenID authentication function:
   a. Log on to WebSphere Integrated Solutions Console as the administrator.
   b. Go to **Security** > **Global security** > **Web and SIP Security** > **Trust association**.
   c. Delete **com.ibm.portal.auth.tai.OpenIdTAI**.
   d. Save your changes.
   e. Stop and restart the WebSphere_Portal server.

Related tasks:

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

Enabling step-up authentication and/or the Remember me cookie

Using step-up authentication and/or the Remember me cookie lets you fine-tune user authentication to pages and portlets.

"Enabling step-up authentication and the **Remember me** cookie"
Step-up authentication provides different authentication levels for pages and portlets. The **Remember me** cookie is an encrypted HTTP cookie that supports authentication. You can present personalized portlets and pages in a public area without the need for users to manually authenticate. Remembered users can view anonymous pages and portlets with a standard or identified authentication level. When you provide a valid **Remember me** cookie, a user can also access protected pages and portlets that require the identified authentication level. If the authentication level is set to authenticated, the user must provide a user ID and password to view the page or portlet.

Enabling step-up authentication and the Remember me cookie

Step-up authentication provides different authentication levels for pages and portlets. The **Remember me** cookie is an encrypted HTTP cookie that supports authentication. You can present personalized portlets and pages in a public area without the need for users to manually authenticate. Remembered users can view anonymous pages and portlets with a standard or identified authentication level. When you provide a valid **Remember me** cookie, a user can also access protected pages and portlets that require the identified authentication level. If the authentication level is set to authenticated, the user must provide a user ID and password to view the page or portlet.

About this task

You can enable step-up authentication only. You can enable the **Remember me** cookie only. Or you can enable both with one task.

Note: Authenticated and remembered users must have cookies enabled on their browser. Users can access portal sites without cookies enabled if they are anonymous users. If you turn on session tracking for anonymous users, then anonymous users also require cookies.

Remember me cookies: After you enable the **Remember me** cookie, you might need to adjust the settings to fit your business needs. You can use the WebSphere Integrated Solutions Console to create new properties, if necessary, or to update existing properties. The `RememberMeConfigService.properties` file, including a short description, is in the *wp_profile_root*`\PortalServer\config` directory. Log in to the WebSphere Integrated Solutions Console. Go to **Resources** > **Resource Environment Providers** > **WP RememberMeConfigService** > **Custom properties** to edit the properties file. All property changes require that you restart the WebSphere Portal Express server for the changes to take effect.

"Enabling step-up authentication, the Remember me cookie, or both" on page 1588
You can choose to enable either step-up authentication or the Remember me cookie individually or you can choose to enable these features together.

Enabling step-up authentication, the Remember me cookie, or both

You can choose to enable either step-up authentication or the Remember me cookie individually or you can choose to enable these features together.

Before you begin

Log on to the WebSphere Integrated Solutions Console and go to **Security** > **Global security** > **Web and SIP security** > **Single sign-on (SSO)**. Verify that both **Interoperability Mode** and **Web inbound security attribute propagation** are enabled.

You can use step-up authentication with Web Services for Remote Portlets (WSRP) extensions. The authentication level that is defined for portlets on the Producer portal is automatically set on the Consumer portal when it consumes WSRP services. If you apply step-up authentication mechanisms on the Producer, users are also challenged for stronger authentication credentials on the Consumer portal as required. To use step-up authentication with a WSRP extension, ensure that your environment meets the following requirements:

- The Producer and Consumer portals are WebSphere Portal Express Version 8.5 or later.
- You enable step-up authentication on both the Producer and Consumer portals.
- The authentication levels are the same on the Producer and Consumer portals.

  Notes:
  - Portal administrators can change authentication levels on both the Producer portal or Consumer portal at any time.
  - If the authentication level on the Consumer portal is less than the authentication level on the Producer portal, the Producer portal gives the following error message: `AccessDeniedFault EJPWC1118E: User authentication not strong enough`. Then, users cannot access the portlet. For this reason, the authentication level on the Consumer portal must be the same as the authentication level on the Producer portal.

About this task

Important: The Remember me cookie does not extend the Portal Personalization feature to the public area. When the Remember me cookie identifies a user in a public area, the user is still considered anonymous from an access control point of view.

Web Content Manager note: The authoring portlet and the web content viewer do not fully support step-up authentication or the Remember me cookie. However, the user name component is aware of the Remember me cookie. If the Remember me cookie is set on a request and a user is not logged in, the anonymous user design is not used. Instead, it uses the user name design complete with the name or distinguished name of the user that is specified by the Remember me cookie.

Restriction: Step-up authentication requires the LtpaToken2 for single sign-on. Read Implementing single sign-on to minimize web user authentications for details.

Note: When you enable both step-up authentication and the Remember me cookie, you receive the following authentication levels:

standard

identified

authenticated

If you enable step-up authentication only, you receive the following authentication levels:

standard

authenticated

Procedure

1. Go to the *wp_profile_root*/ConfigEngine/properties directory.
2. Open the wkplc.properties file with a text editor.
3. Enter one of the following values for the **enable_rememberme** parameter under the StepUp Authentication heading:

   Note: Add the **enable_rememberme** parameter to the wkplc.properties file if it does not exist.
   - If you are enabling both step-up authentication and the Remember me cookie, enter true.
   - If you are enabling step-up authentication only, enter false.
   - If you are enabling the Remember me cookie only, leave blank.
4. Enter a value for the following parameters under the StepUp Authentication heading if you are enabling the Remember me cookie:

   Note: Go to the properties file for specific information about the parameters.

   sua_user

   sua_serversecret_password
5. Save your changes to the wkplc.properties file.
6. Open a command prompt.
7. Change to the *wp_profile_root*/ConfigEngine directory.
8. Choose one of the following tasks to modify your environment:
   - If you are enabling step-up authentication and or the Remember me cookie, run the **enable-stepup-authentication** task.
   - If you are enabling the Remember me cookie only, run the **enable-rememberme** task.

   Use the following command syntax:
   - Linux : ./ConfigEngine.sh *task_name* -DWasPassword=*password*

- IBM i: `ConfigEngine.sh` *`task_name`* `-DWasPassword=`*`password`*
- Windows: `ConfigEngine.bat` *`task_name`* `-DWasPassword=`*`password`*

Where *`task_name`* is either **`enable-stepup-authentication`** or **`enable-rememberme`**.

9. Check the output for any error messages before you run any additional tasks. If any of the configuration tasks fail, verify the values in the `wkplc.properties` file.
10. In a clustered environment, copy the `wp.auth.base.sua_loginmodule.jar` file in the *AppServer_root*`/lib/ext/` directory of one of the Portal nodes to the *AppServer_root*`/lib/ext/` directory of the deployment manager.
11. Stop and restart the appropriate servers to propagate the changes. For instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.
12. Complete the following steps to change the authentication level on a page or portlet:
    a. Click the **Administration menu** icon. Then, click **Access** > **Resource Permissions**.
    b. Click either the **Pages** link or the **Portlets** link.
    c. Locate the page or portlet you want to change and click the **Authentication Level** link.
    d. Choose one of the following levels:

       Note: The following Authentication Levels are provided. If you customized your step-up authentication, you might have different levels.

       Standard
       Set the Authentication Level to `Standard` if you want anonymous and identified users to view the page or portlet. The Standard level has the following two states that are based on the access control setting for the page or portlet:
       - If anonymous users have access to the page or portlet, no authentication is required.
       - If only authenticated users have access to the page or portlet, authentication is required.

       Identified (available if `enable_rememberme=true`)
       Set the Authentication Level to `Identified` if you want to control whether content is displayed to an unauthenticated user based on the existence of a persistent HTTP cookie. This option is intended for pages and portlets that are visible to anonymous users. An example is the **Remember me on this computer** option during login. This option generates the **`com.ibm.portal.RememberMe`** cookie.

       If a user previously authenticated to WebSphere Portal Express and then returns with the **`com.ibm.portal.RememberMe`** cookie, the user is "identified" and the content displays. If a user attempts to access WebSphere Portal Express without the **`com.ibm.portal.RememberMe`** cookie, the user is asked to authenticate before the content is displayed.

CAUTION:
Do not set the Access level to identified for the Login portlet. This action causes problems when a user logs in to WebSphere Portal Express.

Authenticated
Set the Authentication Level to `Authenticated` if you want anonymous and identified users to log in to view the page or portlet.

Configuring Remember me for Java Platform, Enterprise Edition authentication

You can configure a Remember me cookie for Java Platform, Enterprise Edition authentication that works with step-up authentication. When this feature is enabled, users are logged in automatically when they access a protected portal area by presenting a valid Remember me cookie.

About this task

If the portal area has a higher step-up authentication, users are asked to provider their user ID and password credentials. Otherwise, the users can enter the protected area without providing their credentials.

Procedure

1. Make sure the WebSphere_Portal server is running.
2. Log on to the WebSphere Integrated Solutions Console as an administrator.
3. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
4. Click **WP RememberMeConfigService** to open the properties view.
5. Click **Custom properties**.
6. Click **New**.
7. Enter the following values for the new property:
   a. **Name**: `j2eeAuthenticate`
   b. **Value**: `true`
   c. **Type**: `java.lang.Boolean`
8. Click **OK**.
9. Click the **Save** link in the Messages box to save the changes to the master configuration.

Disabling step-up authentication and the Remember me cookie

You can disable the step-up authentication feature or the Remember me cookie to remove them from your server.

Procedure

1. Go to the *wp_profile_root*`/ConfigEngine/properties` directory.
2. Open the `wkplc.properties` file with a text editor.
3. Enter one of the following values for the **`disable_rememberme`** parameter under the StepUp Authentication heading:

   Note: Add the **`disable_rememberme`** parameter to the `wkplc.properties` file if it does not exist.

   - If you are disabling both step-up authentication and the Remember me cookie, enter `true`.

- If you are disabling step-up authentication only, enter `false`.
- If you are disabling the Remember me cookie only, leave blank.

4. Save your changes to the `wkplc.properties` file.
5. Open a command prompt.
6. Change to the *wp_profile_root*`/ConfigEngine` directory.
7. Choose one of the following tasks to modify your environment:
   - If you are disabling step-up authentication and or the Remember me cookie, run the **disable-stepup-authentication** task.
   - If you are disabling the Remember me cookie only, run the **disable-rememberme** task.

   Use the following command syntax:
   - Linux : `./ConfigEngine.sh` *task_name* `-DWasPassword=`*password*
   - IBM i: `ConfigEngine.sh` *task_name* `-DWasPassword=`*password*
   - Windows: `ConfigEngine.bat` *task_name* `-DWasPassword=`*password*

   Where *task_name* is either **disable-stepup-authentication** or **disable-rememberme**.
8. Check the output for any error messages before you run any additional tasks. If any of the configuration tasks fail, verify the values in the `wkplc.properties` file.
9. Stop and restart the appropriate servers to propagate the changes. For instructions, go to "Starting and stopping servers, deployment managers, and node agents" on page 1216.

Step-up authentication properties

After you enable step-up authentication, you might want to adjust the settings to fit your business needs. You can use the WebSphere Integrated Solutions Console to create new properties, if necessary, or update existing properties.

For each additional authentication level you must create at least one properties file according to the following naming convention:

- The authentication level name.
- An underline character followed by a language locale. For example, en for English.
- The string `.properties`

For example, for an authentication level of `yourlevel`, you need to create a file such as `yourlevel.properties` or `yourlevel_en.properties`.

This file must be available in class path at `com/ibm/portal/auth/sua/nls/`.

The file must contain two key value pairs. For example, for the system specified authentication level `authenticated`, there are two definitions

```
display-name=authenticated
description=User authentication using username and password
```

These two strings are used to select the authentication level in the Administration portlet for configuration.

The following information contains all properties that apply to the appropriate portal configuration service, namely **WP StepUpConfigService**.

Log on to WebSphere Integrated Solutions Console and then go to **Resources** > **Resource Environment** > **Resource Environment Providers** > **WP StepUpConfigService** > **Custom properties** to access the properties.

Note: All property changes require that you restart the WebSphere_Portal server in order for the changes to take effect.

sua.enable
Use this property to enable and disable the step-up authentication mechanism.

Default: `false`

Type: `java.lang.Boolean`

sua.authLevel.enable
Use this property to provide a comma-separated list of authentication level names.

Notes:

- If step-up authentication is enabled, the authentication level name must be specified.
- If you want to use the **Remember me** cookie, make sure that it is enabled and add the authentication level name for this property.

Default: `authenticated`

Type: `java.lang.String`

sua.authLevel.*auth_level_name*.strength
Use this property to specify the authentication level strength of the authentication level with the name *auth_level_name*. The value is a non-negative integer that expresses the implied strength of a particular authentication method. The step-up authentication framework considers one authentication method to be stronger than another if it has a higher value.

Note: The value 0 is reserved by the step-up authentication engine, and therefore it is not allowed to assign values less than one. It is possible to leave gaps in the sequence of authentication level strengths. It is not possible to assign the same authentication level to multiple authentication level names.

Default: **sua.authLevel.identified.strength** =5
sua.authLevel.authenticated.strength =10

Type: `java.lang.Integer`

sua.authLevel.*auth_level_name*.required
Use this property to specify whether the authentication level with the name *auth_level_name* is optional or required. When a user accesses a resource with an optional authentication level, this resource might be accessed if the first required authentication level is verified. When an authentication level is flagged as required, it can be verified successfully only if all required authentication levels can be verified successfully.

Note: This property must not be set for the authentication level that is identified or authenticated. If one authentication level is set as optional, all previous levels must also be optional.

Default: `true`

Type: `java.lang.boolean`

sua.authLevel.*auth_level_name*.authLevelVerifier
Use this property to specify the fully qualified name of the class that implements the **com.ibm.portal.auth.stepup.AuthLevelVerifier** SPI. It also verifies whether the authentication level of **auth_level_name** is valid for a request.

Note: This property must not be set for the authentication level that is identified or authenticated.

Default: -

Type: `java.lang.String`

sua.authLevel.*auth_level_name*.stepUpAuthHandler
Use this property to specify the fully qualified name of the class that implements the **com.ibm.portal.auth.stepup.StepUpAuthHandler** SPI. It also establishes the authentication level of **auth_level_name**.

Note: This property must not be set for the authentication level that is identified or authenticated.

Default: -

Type: `java.lang.String`

sua.authLevel.*auth_level_name*.postRedirectionTargetProtected
The step-up authentication handler redirects a user with a certain authentication level to another page. For example, it redirects to a page with a login form. The step-up authentication framework redirects the user to the resource requested before the authentication level enforcement. This property specifies whether the redirection to the originally requested resource point to the public or the protected portal area. The implementation of the authentication level might move the user from an unauthenticated to an authenticated state.

Note: This property must not be set for the authentication level that is identified or authenticated.

Default: `false`

Type: `java.lang.Boolean`

Example: `true`

sua.authLevel.*auth_level_name*.property.*property_name*
Use this property to specify further properties that are available. The properties are received with their *property_name*. The prefix **sua.authLevel.auth_level_name.property** is omitted.

Default: -

Type: `java.lang.String`

Securing LTPA keys on a production environment

The Lightweight Third Party Authentication (LTPA) key holds cryptographic keys that secure the user authentication session and cookies. To secure the production server environment, regenerate the LTPA key using the WebSphere Integrated Solutions Console. If you plan to enable single sign-on at a later time, you must first disable the automatic key generation.

About this task

Complete the following steps to secure LTPA keys on a production environment:

Procedure

1. Complete the following steps to regenerate the LTPA keys:

   Note: These steps only need to be completed once in a clustered environment.

   a. Log on to the WebSphere Integrated Solutions Console.
   b. Navigate to **Security** > **Secure administration, applications, and infrastructure**.
   c. Click **Authentication mechanisms and expiration**.
   d. Click **NodeLTPAKeySetGroup** under Key Generation and then click **Generate Keys**.
   e. Click **Save** to save the changes to the master configuration.

   Restriction: By default, WebSphere Application Server is configured to automatically regenerate the LTPA keys every 90 days. If you setup single sign-on to export the LTPA key and then import it on another server, disable the automatic key generation; otherwise, single sign-on fails after 90 or 180 days because of regenerated keys.
2. Complete the following steps to disable automatic LTPA key generation on all servers of the single sign-on domain:
   a. Log on to the WebSphere Integrated Solutions Console.
   b. Navigate to **Security** > **Secure administration, applications, and infrastructure**.
   c. Click **Authentication mechanisms and expiration**.
   d. Click **Key generation - Key set groups**.
   e. Click **NodeLTPAKeySetGroup**.
   f. Disable the **Key generation - Automatically generate keys** checkbox.
   g. Click **OK**.
   h. Click **Save** to save the changes to the master configuration.

Configuring SSL

Secure socket layers (SSL) encrypt traffic between the client browser and the server to secure information exchanged over the network between the browser and IBM WebSphere Portal Express. You will need to configure your environment for SSL to activate this additional security feature.

About this task

Perform the following tasks to configure SSL:

1. "Setting up SSL" on page 1596
   Get an overview of the tasks that are required to configure SSL for IBM WebSphere Portal Express. Some of these tasks are completed on the IBM WebSphere Application Server and the web server. The steps that refer to the WebSphere Application Server and the web server are summarized here; refer to the WebSphere Application Server and the web server documentation for detailed information. Steps that are unique to WebSphere Portal Express are described in detail here.

2. “Configuring SSL only for the login process” on page 1601
   You can encrypt only the login process to IBM WebSphere Portal Express and then allow subsequent requests through HTTP.
3. “Setting up Client Certificate Authentication” on page 1603
   View the steps required to configure IBM WebSphere Portal Express for SSL client certificate authentication. The supported scenario is a "client certificate only" setup that switches completely to this authentication method and does not allow form-based login via username and password. Other configuration scenarios are possible, but are neither recommended nor supported.
4. “Cryptographic hardware for SSL acceleration” on page 1607
   If your portal environment makes extensive use of SSL, you might choose to use cryptographic hardware to offload encryption and improve performance. WebSphere Portal Express tolerates interfacing through WebSphere Application Server with cryptographic hardware for SSL acceleration. However, the tasks that are involved in setting up and configuring cryptographic hardware are specific to web servers or WebSphere Application Server and do not necessarily involve configuring WebSphere Portal Express.

Setting up SSL

Get an overview of the tasks that are required to configure SSL for IBM WebSphere Portal Express. Some of these tasks are completed on the IBM WebSphere Application Server and the web server. The steps that refer to the WebSphere Application Server and the web server are summarized here; refer to the WebSphere Application Server and the web server documentation for detailed information. Steps that are unique to WebSphere Portal Express are described in detail here.

About this task

Note: This procedure might be slightly different if a front-end security proxy server such as IBM Security Access Manager WebSEAL is used. In that case, the front-end security server handles the client SSL connections. The web server receives connections from the front-end security proxy server. Mutually authenticated SSL can be configured between the web server and the front-end security proxy server if needed. It is highly dependent on the security requirements of each deployment.

If you plan to use a Security Access Manager WebSEAL TAI with an SSL junction, complete only steps 1-3 of this procedure.

Important: If only the login process is secure over SSL, complete the first three steps and then go to Configuring SSL only for the login process.

Procedure

1. Configure the web server to support HTTPS. This configuration involves setting up the web server to accept inbound connections from client browsers over SSL.
2. Depending on the web server that you want to use, other software must be installed on the web Server. For example: instance Microsoft Internet Information Server and Microsoft Certificate Service.
3. The web server must have a port that is defined (usually 443), and the necessary certificates and keys must be installed.
   - Go to Securing with SSL communications in the related links section for information about how to enable SSL on an IBM HTTP Server.

4. In a production environment, you must obtain a certificate from a certificate authority. For testing purposes, you can use iKeyman to generate a self-signed certificate. For Internet Information Server, use the web server's resource toolkit to create SSL keys. Refer to the related links section for information about iKeyman and creating Secure Sockets Layer digital certificates.
5. Configure the WebSphere Application Server plug-in for the web server to forward WebSphere Portal Express traffic that is received over SSL to WebSphere Application Server (which then forwards the traffic to WebSphere Portal Express). Refer to the related links section for information about how to configure the plug-in.
6. In configurations where the web server and WebSphere Portal Express are on separate servers, requests are rerouted to the application server. Under these circumstances, you can also configure SSL between the web server and the application server to provide complete security. This configuration requires that you create extra keyfiles for the web server plug-in and for the embedded HTTPS of WebSphere Application Server.
   - For information about configuring SSL between the web server and the application server, use the IBM Redbooks called WebSphere Application Server V8.5 Security Guide, found in the related links section. Use the section that is called *Application server configuration: Web container configuration of the IBM WebSphere Application Server*.

   Note: Always create a new SSL keystore and truststore for the external web server and change the WebSphere_Portal server's secure transport channel to use the new SSL repository.
   CAUTION:
   Do not modify the default SSL key and truststore.
7. Required: Complete the following steps to create or modify the following two properties in the configuration services:
   a. Log on to the WebSphere Integrated Solutions Console.
   b. Go to **Resources** > **Resource Environment** > **Resource Environment Providers**.
   c. Click **WP ConfigService**.
   d. Click **Custom Properties** under the Additional Properties heading.
   e. Locate the **redirect.login.ssl** property and complete one of the following options:

      Parameter values: The **redirect.login.ssl** determines the protocol to use after login completes. Specify one of the following values:
      - Set to `true` to use HTTPS.
      - Set to `false` to use HTTP.
      - If the property exists, click the property to modify it and change the value to `true`.
      - If the property does not exist, click **New** to create the property and enter the following information:
        - **Name**: `redirect.login.ssl`
        - **Value**: `true`
        - **Type**: `java.lang.String`

   f. Locate the **host.port.https** property and complete one of the following options:

- If the property exists, click the property to modify it and change the value to *alias_port*.

  Note: The *alias_port* is the port number that is used for the virtual host alias that is specified in a previous step (usually 443).
- If the property does not exist, click **New** to create the property and enter the following information:
  - **Name**: host.port.https
  - **Value**: 443
  - **Type**: java.lang.String

g. Click **Save** to save the changes to the master configuration.

h. Log out of the WebSphere Integrated Solutions Console.

8. Update the Transport Security Constraint in wps.ear. You can modify the transport so that WebSphere Application Server enforces the use of SSL for all pages under the /myportal/ URL. Use this step to completely secure the protected area over HTTPS.

   Clustered environments: Complete this step on the primary node, then complete a full resynchronize to propagate the changes to all nodes.

   a. Export wps.ear. See the related tasks section for instructions.

   b. Go to the directory where you exported wps.ear: *path_to_exported_EAR*/installedApps/*node_name*/wps.ear/wps.war/WEB-INF

      Note: You might need to extract the exported EAR before you can edit any files.

   c. Locate and open web.xml with any text editor.

   d. Set the value of the <transport-guarantee> element to CONFIDENTIAL under the <security-constraint> element for the /myportal/* URL. Do not change the values for the other <transport-guarantee> elements. Use the following information to update the XML file:

```
<security-constraint id="SecurityConstraint_1">
   <web-resource-collection id="WebResourceCollection_1">
      <web-resource-name></web-resource-name>
   <url-pattern>/myportal/*</url-pattern>
       <http-method>DELETE</http-method>
      <http-method>POST</http-method>
      <http-method>GET</http-method>
      <http-method>PUT</http-method>
   </web-resource-collection>
   <auth-constraint id="AuthConstraint_1">
      <description></description>
      <role-name>All Role</role-name>
   </auth-constraint>
   <user-data-constraint id="UserDataConstraint_4">
      <transport-guarantee>CONFIDENTIAL</transport-guarantee>
   </user-data-constraint>
</security-constraint>
```

   e. Save and close web.xml.

   f. Redeploy wps.ear. See the related tasks section for instructions.

   g. Synchronize the nodes in your clustered environment.

      1) Log in to the Deployment Manager.
      2) Select **System Administration** > **Nodes**.
      3) Select the nodes to synchronize from the list.
      4) Click **Full Resynchronize**.

9. Complete the following steps to update theme links:
   a. Edit the JSP and JSPF files that provide the login link. Locate the JSP and JSPF files that include the "wps.Login" string: Do not edit any of the themes that are included with WebSphere Portal Express because these themes are updated with fixes. Instead, copy the theme and make your changes to the copy.

      Finding theme resources: See the *Location of theme resources* link in the Related section.

      This attribute must appear in a tag similar to the following code:

      ```
      <portal-navigation:urlGeneration contentNode="wps.Login"
           portletWindowState="Normal">
      ```

      The exact structure of this tag can vary depending on how it was constructed by the page designer. The JSP comments might also be used to indicate where the login link is located:

      ```
      <%-- login button --%>
      ```

   b. After you find the login link, change or add the ssl="true" attribute to the <portal-navigation:urlGeneration> tag of the anchor. For example:

      ```
      <wps:if loggedIn="no" notSelection="wps.Login">
       <wps:urlGeneration contentNode="wps.Login"
                  portletWindowState="Normal" ssl="true">
        <td class="wpsToolBar" valign="middle" nowrap>
         <a href="<% wpsURL.write(escapeXmlWriter); %>" class="wpsToolBarLink">
         <wps:text key="link.login" bundle="nls.engine"/>
         </a>
        </td>
       </wps:urlGeneration>
      </wps:if>
      ```

      Note: The previous examples use the portal-navigation: prefix to designate JSP tags from the portal navigation tag library. Your custom JSP tags might use a different tag prefix.

10. Optional: Complete the following steps when you use a remote web server if you must allow direct access to the WebSphere_Portal node on the internal port. For example, http://*hostname.example.com*:*10039*/wps/portal, where *hostname.example.com* is the fully qualified host name of the server where Portal is running and *10039* is the default transport port that is created by WebSphere Application Server. The port number might be different for your environment.:
    a. From the WebSphere Integrated Solutions Console, go to **Servers** > **Server Types** > **WebSphere application servers** > **WebSphere_Portal** > **Web Container Settings** > **Web Container Transport Chains**.
    b. Click **New**.
    c. Select a name for the transport chain.
    d. Select the **WebContainer-Secure** template (templates/chains|webcontainer-chains.xml).
    e. Select **Next**.
    f. Specify the **Port name**. For example, port 443.
    g. Click **Next**.
    h. Click **Finish** to confirm the creation of the transport chain.
    i. Click **Save**.

j. In a clustered environment, repeat the previous steps for each node in the cluster. For example, WebSphere_Portal2, and then synchronize the changes to all nodes.

11. Optional: Complete the following steps only if you use the **Login** portlet:
    a. Log in to WebSphere Portal Express.
    b. Click the **Administration menu** icon. Then, click **Portlet Management** > **Portlets**.
    c. Locate the Login portlet and click the **Configure portlet** icon.
    d. Locate the **UseSecureLoginActionUrl** parameter and click the **Edit value** icon.
    e. Type `true` in the **Value** field and click **OK** to save your changes.
    f. Click **OK** to return to the **Manage Portlets** portlet.
12. In a stand-alone environment, stop and restart the WebSphere_Portal server. In a clustered environment, stop and restart the Deployment Manager and the WebSphere_Portal servers.

    Clustered environments: In the Deployment Manager, verify that the EAR changes were successfully synchronized to all nodes. Stop and restart the servers on all nodes.
13. Complete the following steps to test your changes:
    a. Start the home page in a web browser through an HTTP URL that is not secure. For example, `http://hostname.example.com:10039/wps/portal`, where *hostname.example.com* is the fully qualified host name of the server where Portal is running and *10039* is the default transport port that is created by WebSphere Application Server. The port number might be different for your environment..
    b. Verify that the login link in the banner area uses the HTTPS schema for the link to the login page.
    c. Enter your user name and password. Then, click the login link to verify that the page is protected. The URL must be HTTPS and the browser must indicate that the page is protected.

       Browser security prompt: After you click the login link to accept the server certificate, a browser security prompt might appear.
    d. Log off.
    e. Log in using an HTTP URL that is not secure and that points directly to the protected area. For example, `http://hostname.example.com:10039/wps/portal`, where *hostname.example.com* is the fully qualified host name of the server where Portal is running and *10039* is the default transport port that is created by WebSphere Application Server. The port number might be different for your environment.
    f. Verify that you are requested to log in and that the login page and the portal page are protected through SSL.

       Note: If the security-constraint was not modified to **CONFIDENTIAL**, SSL does not protect the login page and the portal pages.

Related concepts:

"Understanding the Portal Version 8.5 modularized theme" on page 2521
Modern websites and browsers enable incredible new capabilities that can greatly enhance your user's web experiences. However, these capabilities are not without cost in terms of large page sizes and more processing in the browser when each page is rendered. These capabilities are worth it when you need them, but removing them for an entire site or including them only on pages that take advantage of these capabilities provides for more flexibility.

Related tasks:

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

Related information:

z/OS HTTP Server Planning, Installing, and Using

Securing with SSL communications

Creating Secure Sockets Layer digital certificates and System Authorization Facility keyrings that applications can use to initiate HTTPS requests

Configuring the web server plug-in for Secure Sockets Layer

Installing and configuring the plug-in for V5.3 HTTP Server for z/OS

WebSphere Application Server V7.0 Security Guide

Redbooks logo

Configuring SSL only for the login process

You can encrypt only the login process to IBM WebSphere Portal Express and then allow subsequent requests through HTTP.

About this task

Complete the following steps to configure SSL only for the login process:

Remember: These steps configure SSL only for the login; if you want to configure SSL for other features such as themes and skins, complete the steps in *Setting up SSL*.

Procedure

1. Configure SSL for the webserver plug-in if you have an external webserver that is configured for SSL. Consult with your webserver vendor for more details on how to configure SSL for your webserver. For more information, go to *Guide to properly setting up SSL within the IBM HTTP Server* topic in the related information section.

   Note: Proceeding with this task without configuring SSL for the webserver plug-in causes the login to fail.
2. Verify that the following parameters exist and are correctly set for your installation in the **WP ConfigService** application:

a. Log on to the WebSphere Integrated Solutions Console in a stand-alone environment or on the Deployment Manager WebSphere Integrated Solutions Console in a clustered environment.
b. Go to **Resources** > **Resource Environment** > **Resource Environment Providers**.
c. Click **WP ConfigService**.
d. Click **Custom Properties** under the Additional Properties heading.
e. Locate the **redirect.login.ssl** property and do one of the following options:

   Note: The **redirect.login.ssl** parameter determines the protocol to use after login completes. If this parameter is set to `true`, https is used. If this parameter is set to `false`, http is used. This setting is not affected by the protocol that is used to access the main page.
   - If the property exists, click the property to modify it and change the value to `true`.
   - If the property does not exist, click **New** to create the property and enter the following information:
     - **Name**: `redirect.login.ssl`
     - **Value**: `true`
     - **Type**: `java.lang.String`

f. Locate the **host.port.https** property and do one of the following options:
   - If the property exists, click the property to modify it and change the value to *alias_port_for_HTTPS*.

     Note: The *alias_port_for_HTTPS* is the port number that is used for the virtual host alias (usually 443).
   - If the property does not exist, click **New** to create the property and enter the following information:
     - **Name**: `host.port.https`
     - **Value**: `443`
     - **Type**: `java.lang.String`

g. Locate the **host.port.http** property and do one of the following options:

   Note: Set the **host.port.http** if you are using a port other than the default 80.
   - If the property exists, click the property to modify it and change the value to *alias_port_for_HTTP*.

     Note: The *alias_port_for_HTTP* is the port number that is used for the virtual host alias (usually 80).
   - If the property does not exist, click **New** to create the property and enter the following information:
     - **Name**: `host.port.http`
     - **Value**: `80`
     - **Type**: `java.lang.String`

h. Click **Save** to save the changes to the master configuration.
i. Log out of the WebSphere Integrated Solutions Console.

3. Complete the following steps to encrypt the login process to WebSphere Portal Express and allow subsequent requests through HTTP:

The Login portlet uses the **UseSecureLoginActionUrl** parameter to control the generation of the login action URL. Set this parameter to `true` to use a secure URL for login.

a. Click the **Administration menu** icon. Then, click **Portlet Management** > **Portlets**.
b. Search for **Title start with = "Login"**.
c. Select the **Configure portlet** icon.
d. Edit the **UseSecureLoginActionUrl** parameter and set the parameter to `true`.

For more information about setting up SSL within the IBM HTTP Server, see *Guide to properly setting up SSL within the IBM HTTP Server*.

Results

You can test the SSL login by using the following unprotected URL: `http://portalserver.com/wps/myportal` and submitting your credentials. You notice that the URL does not change to https.

Note: Confirm that the login was encrypted by monitoring the packets through a network utility such as Ethereal or by reviewing the source code of the login form when accessed through an unprotected HTTP URL. The login form must have an action URL that is secured, for example `<form method="post" action="https://....">`. Set your browser to warn you when you change between secure and insecure modes to see the behavior on the client-side.

Related information:

Guide to properly setting up SSL within the IBM HTTP Server

Setting up Client Certificate Authentication

View the steps required to configure IBM WebSphere Portal Express for SSL client certificate authentication. The supported scenario is a "client certificate only" setup that switches completely to this authentication method and does not allow form-based login via username and password. Other configuration scenarios are possible, but are neither recommended nor supported.

About this task

Complete the following steps to configure WebSphere Portal Express for SSL client certificate authentication:

Note: See related references section on how to setup WAS for SSL support with client certification.

Procedure

1. Ensure you complete the following steps when you configure IBM WebSphere Application Server for SSL support with client certificates:
   a. When you define the Quality of protection (QoP) settings for the new SSL Repertoire, do the following:
      1) Choose **Required** from the **Client Authentication** list
      2) Choose **SSL_TLS** from the **Protocol** list.
      3) In the **Provider** section, select **Predefined JSSE provider** then choose **IBMJSSE** from the **Select provider** list.

For more information, see the *Quality of Protection* topic in the WebSphere Application Server Information Center.

b. Ensure you reference the correct key and trust files. You should create new key and trust files using the IKEYMAN tool and the PKCS12 format for maximum browser compatibility.

   Note: The key file must contain the server certificate. The trust file must contain either all the client certificates of users that will be authenticated or a certification authority certificate (CA key) that can be used to verify the client certificates of users.

c. Associate the secure transport chain with the new SSL Repertoire.

d. Configure your advanced LDAP security settings. Certificate-based authentication requires that you configure the authentication mechanism so that one of the following conditions apply:

   - WebSphere Application Server maps the entire Distinguished Name (DN) from the subject field of the certificate to a corresponding Distinguished Name in your LDAP. To use this option, set the mapping technique in the LDAP configuration panel to `exact`.
   - WebSphere Application Server maps the entry in the subject field to a different attribute than the Distinguished Name in your user registry. To use this option, set up the mapping technique in the LDAP configuration panel to use the certificate filter option. Using the certificate filter option allows you more flexibility in using attributes other than the Distinguished Name to identify the users. For example, the filter `uid=${SubjectCN}` maps the `SubjectCN` field of the client certificate to the uid attribute in your LDAP.

2. Complete the following steps if you use an external HTTP server:

   a. Regenerate the plug-in. Go to **Servers** > **Web Servers**. Select the **Web server** and click **Generate Plug-in**. Update the HTTP server with the generated plug-in.

   b. Restart the HTTP server for the changes to take effect.

   c. Enable client certificate authentication in your Web server. For IBM HTTP Server (IHS), refer to http://www.redbooks.ibm.com/ and search for security handbook for the latest information about WebSphere Application Server.

3. Update `wps.ear` to change the authentication method and transport guarantee setting to support client certificate authentication.

   Clustered environments: Complete this step on the primary node, then complete a full resynchronize to propagate the changes to all nodes.

   a. Export `wps.ear`. See the following topic title in the Related task section for instructions: *Exporting the portal EAR file*.

   b. Go to the directory where you exported `wps.ear`: *`path_to_exported_EAR`*`/installedApps/`*`node_name`*`/wps.ear/wps.war/WEB-INF`

      Note: You might need to extract the exported EAR before you can edit any files.

   c. Edit the `web.xml` file located in the exported ear directory under `/wps.war/WEB-INF`.

   d. Change the login-config tag to the client certificate authentication method.

```
<login-config id="LoginConfig_1">
  <auth-method>CLIENT-CERT</auth-method>
  <realm-name>WPS</realm-name>
```

```
  <!--<form-login-config id="FormLoginConfig_1">
   <form-login-page>/redirect</form-login-page>
   <form-error-page>/error.html</form-error-page>
  </form-login-config> -->
</login-config>
```

e. Change the transport guarantee setting in the security constraint for the protected area to `CONFIDENTIAL`:

```
<security-constraint id="SecurityConstraint_1">
  <web-resource-collection id="WebResourceCollection_1">
   <web-resource-name/>
   <url-pattern>/myportal/*</url-pattern>
   <http-method>DELETE</http-method>
   <http-method>GET</http-method>
   <http-method>POST</http-method>
   <http-method>PUT</http-method>
   <http-method>HEAD</http-method>
  </web-resource-collection>
  <auth-constraint id="AuthConstraint_1">
   <description/>
   <role-name>All Role</role-name>
  </auth-constraint>
  <user-data-constraint id="UserDataConstraint_4">
   <transport-guarantee>CONFIDENTIAL</transport-guarantee>
  </user-data-constraint>
</security-constraint>
```

f. Save and close `web.xml`.

g. Redeploy `wps.ear`. See the following topic title in the Related task section for instructions: *Redeploying the portal EAR file*.

h. Clustered environments: Synchronize the nodes.

1) Log in to the Deployment Manager.

2) Select **System Administration** > **Nodes**.

3) Select the nodes to synchronize from the list.

4) Click **Full Resynchronize**.

4. Update the themes and settings.

a. Modify the URLs for login and logout in the themes that are used in your scenario. The files that contain the login and logout links can be different, depending on the theme. In more recent themes, these links might be located in `Default.jsp`. In older themes, these links might be located in either `banner.jspf` or `mainMenu.jsp`.

Finding theme resources: See the *Location of theme resources* link in the Related section.

Clustered environments: Complete the following steps. Notice that in a clustered environment, the steps must be completed on the Deployment Manager.

- For the login link, use an arbitrary protected page. The login link will then implicitly trigger the SSL handshake in WebSphere Application Server due to the security constraint. For example, you can generate the URL to point to the protected welcome page:

```
<%-- Login button --%>
<%-- comment this to enable screen login --%>
<portal-logic:if loggedIn="no">
<portal-navigation:urlGeneration
contentNode="ibm.portal.Home.Welcome" home="protected">
    <a tabIndex="7" class="toolbarLink" href='<%
```

```
wpsURL.write(escapeXmlWriter); %>'>
    <portal-fmt:text key="link.login" bundle="nls.engine"/>
</a></portal-navigation:urlGeneration>
</portal-logic:if>
```

- For the logout, you need to consider whether or not a logout should redirect you back to HTTP. If so, you need to set the property **redirect.logout.ssl** in the configuration service to `true`. Also, set the **host.port.http** in the same service to the correct port. If you want to stay in the HTTPS protocol after the logout, you do not need to complete any configuration steps here.

b. Remove the login portlet from all pages where it is placed; for example, the welcome and the login page.

c. If you want to completely disable the entry points 'login portlet' and 'login URL' to WebSphere Portal Express, complete the following steps: set the **command.login** property in the configuration service to the value `LoginUserBlocked`. This ensures that a login can only be triggered after being authenticated by WebSphere Application Server, in this case by the client certificate handshake.

1) Log on to the WebSphere Integrated Solutions Console.
2) Go to **Resources** > **Resource Environment** > **Resource Environment Providers**.
3) Click **WP ConfigService**.
4) Click **Custom Properties** under the Additional Properties heading.
5) Click **command.login** and change the value from `LoginUserAuth` to `LoginUserBlocked`.
6) Click **Save** to save the changes to the master configuration.
7) Log out of the WebSphere Integrated Solutions Console.

5. Stop and restart the appropriate servers to propagate the changes. For specific instructions, see “Starting and stopping servers, deployment managers, and node agents” on page 1216.
6. Verify your setup.
   a. Import one of the client certificates that are accepted by the server to your browser.
   b. Launch the home page in this browser through an HTTP URL that is not secure; for example, `http://`*`hostname.example.com`*`:`*`10039`*`/wps/portal`, where *hostname.example.com* is the fully qualified host name of the server where Portal is running and *10039* is the default transport port that is created by WebSphere Application Server. The port number might be different for your environment..
   c. Click the login link.
   d. Verify that the server switches to HTTPS and you are prompted for the client certificate.
   e. After selecting and confirming the correct client certificate, you are redirected to the protected area served with HTTPS.

Related concepts:

“Understanding the Portal Version 8.5 modularized theme” on page 2521
Modern websites and browsers enable incredible new capabilities that can greatly enhance your user's web experiences. However, these capabilities are not without cost in terms of large page sizes and more processing in the browser when each page is rendered. These capabilities are worth it when you need them, but removing them for an entire site or including them only on pages that take advantage of these capabilities provides for more flexibility.

Related information:

WebSphere Application Server Network Deployment Version 8.5: Creating a Secure Sockets Layer configuration

WebSphere Application Server Network Deployment Version 8.5: Quality of protection (QoP) settings

Cryptographic hardware for SSL acceleration

If your portal environment makes extensive use of SSL, you might choose to use cryptographic hardware to offload encryption and improve performance. WebSphere Portal Express tolerates interfacing through WebSphere Application Server with cryptographic hardware for SSL acceleration. However, the tasks that are involved in setting up and configuring cryptographic hardware are specific to web servers or WebSphere Application Server and do not necessarily involve configuring WebSphere Portal Express.

The WebSphere Application Server Information Center contains several topics for setting up and configuring password encryption with cryptographic hardware. Refer to these topics to get started with password encryption and learn more about available encryption features.

Most cryptographic hardware requires the PKCS11 support software for the host machine and internal firmware. To get started with cryptographic hardware, you must install the required support software, configure IBM HTTP Server, then install the necessary devices. Refer to *Getting started with the cryptographic hardware for SSL* at: http://www-01.ibm.com/support/knowledgecenter/SSEQTJ_8.5.5/com.ibm.websphere.ihs.doc/ihs/tihs_cryptossl.html

You can create a plug point to encrypt and decrypt all passwords in WebSphere Application Server that are currently encoded or decoded by using Base64-encoding. Refer to *Plug point for custom password encryption* at:

Stand-alone environments: http://www-01.ibm.com/support/knowledgecenter/SSEQTP_8.5.5/com.ibm.websphere.base.doc/ae/csec_plugpoint_custpass_encrypt.html

Clustered environments: http://www-01.ibm.com/support/knowledgecenter/SSAW57_8.5.5/com.ibm.websphere.nd.multiplatform.doc/ae/csec_plugpoint_custpass_encrypt.html

Create a custom class to encrypt passwords after you create your server profile. Refer to *Enabling custom password encryption* at:

Stand-alone environments:http://www-01.ibm.com/support/knowledgecenter/SSEQTP_8.5.5/com.ibm.websphere.base.doc/ae/tsec_enable_custpass_encrypt.html

Clustered environments: http://www-01.ibm.com/support/knowledgecenter/SSAW57_8.5.5/com.ibm.websphere.nd.multiplatform.doc/ae/tsec_enable_custpass_encrypt.html

In stand-alone environments, administrative functions such as installing `WAR` files or adding trace settings can fail when you meet both of the following conditions:

- Your WebSphere Portal Express server uses the **`RSA_token`** value for security.
- You enable cryptographic offloading of SSL decryption and encryption through an implementation of PKCS11.

If your stand-alone environment meets both of the preceding conditions, complete the following steps:

1. Log in to the WebSphere Integrated Solutions Console.
2. Go to **Security** > **Global Security** > **Administrative security** > **Administrative authentication**
3. Select **Only use the active application authentication mechanism**.
4. Click **Apply** then **OK** and save the changes to the master configuration.
5. Log out of the WebSphere Integrated Solutions Console.
6. Restart the WebSphere_Portal server.

Related tasks:

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

Enabling FIPS and (NIST) SP800-131a

IBM WebSphere Portal Express tolerates IBM WebSphere Application Server support of Federal Information Processing Standards (FIPS) and National Institute of Standards and Technology (NIST) SP800-131a. You can configure WebSphere Application Server to activate FIPS 140-2 compliant security modules. When you enable FIPS, you can use only FIPS to securely encrypt data. For this reason, you must also configure FIPS for systems that require secure transactions, which can include HTTP servers and LDAP servers.

Before you begin

You must install WebSphere Portal Express before enabling FIPS.

If your portal environment includes an HTTP server or LDAP server or any other components that use secure connections, consult the related links section to determine the level of support for FIPS 140-2 and SP800-131a. However, your environment does not need to include an HTTP server or LDAP server. You can enable FIPS on an out-of-box WebSphere Portal Express installation. Likewise, you do not have to enable FIPS for systems that do not require secure transactions. For example, if your LDAP server is accessed via the LDAP protocol, rather than the secure LDAPS protocol, you do not need to enable FIPS for that LDAP server.

About this task

The tasks involved in enabling FIPS are specific to Web servers and WebSphere Application Server and do not involve any configuration steps in WebSphere Portal Express. The WebSphere Application Server Information Center contains several topics with information and instructions for enabling FIPS for HTTP servers. Refer to these topics as appropriate to learn whether you should enable FIPS and, if necessary, what steps to perform.

HTTP servers

See Securing applications at the transport level for Web services in the WebSphere Application Server Information Center for instructions. Configure your HTTP server to support TLS with FIPS enabled. Refer to the appropriate documentation for instructions.

LDAP servers

Refer to the appropriate documentation to configure your LDAP over SSL and to enable FIPS.

Remember: Enable FIPS for your LDAP server only if it requires a secure connection. If you do not use an LDAP server or you do not connect to your LDAP server over a secure connection, you do not need to enable FIPS for an LDAP server.

Related concepts:

"Federal Information Processing Standards and and (NIST) SP800-131a" on page 1520

Federal Information Processing Standards (FIPS) and NIST SP800-131a are standards and guidelines issued by the United States National Institute of Standards and Technology (NIST) for federal government computer systems. FIPS and SP800-131a are developed when there are compelling federal government requirements for standards, such as for security and interoperability, but acceptable industry standards or solutions do not exist.

Related information:

Securing web services applications at the transport level

Configuring Session Security Integration

IBM WebSphere Application Server protects your session from access by other users.

About this task

To use Session Security Integration, you must enable it on both IBM WebSphere Portal Express and WebSphere Application Server functionality as well. WebSphere Application Server will preserve an available security context even if an unprotected URI is accessed.

Procedure

1. In the WebSphere Integrated Solutions Console, click **ServersServer TypesWebSphere Application Servers**.
2. Select **WebSphere Portal**.
3. Expand the **Web Container Settings**. Then click **Web Container**.
4. Click **Session Management**.
5. Select **Security Integration**.
6. Save your changes.

Enable WebSphere Portal Express to create Security context on the unprotected URI

7. Click **Security** > **Global Security**.
8. Expand **Web and SIP security**. Then click **General Settings**.
9. Select **Use available authentication data when an unprotected URI is accessed**.
10. Save your changes.

Enabling HTTP Basic Authentication for simple clients

IBM WebSphere Portal Express provides an HTTP Basic Authentication Trust Association Interceptor that can be enabled to allow specific clients to log into the portal by using HTTP Basic Authentication instead of HTTP Form Based Authentication.

"The HTTP Basic Authentication Trust Association Interceptor"
The HTTP Basic Authentication Trust Association Interceptor (TAI) can be used to authenticate incoming requests using the HTTP Basic Authentication Protocol described in RFC 2617. This can be useful for clients that are not capable of doing HTTP FORM based authentication.

"Configuring the HTTP Basic Authentication Trust Association Interceptor" on page 1611
To configure the HTTP Basic Authentication Trust Association Interceptor according to your requirements, you set its properties.

"Reference: Properties for the Trust Association Interceptor" on page 1611
The HTTP Basic Authentication Trust Association Interceptor has several configuration properties.

"HTTP Basic Authentication Trust Association Interceptor in combination with external authentication servers" on page 1614
When you use the HTTP Basic Authentication Trust Association Interceptor (TAI) with external authentication servers, you can configure the TAI to work for a specific set of requests.

The HTTP Basic Authentication Trust Association Interceptor

The HTTP Basic Authentication Trust Association Interceptor (TAI) can be used to authenticate incoming requests using the HTTP Basic Authentication Protocol described in RFC 2617. This can be useful for clients that are not capable of doing HTTP FORM based authentication.

In general, HTTP Basic Authentication has the following two main disadvantages compared to HTTP Form based authentication:

- With HTTP Basic Authentication, the Web client sends the user ID and password information used for authentication with each individual request to the IBM WebSphere Application Server. This typically requires using transport layer security (SSL) for the complete portal related network traffic. Otherwise the user password is exposed on the network. Compared to this, when you use HTTP form based authentication, it can be sufficient that you use the transport layer security to cover only the user login flow.
- With HTTP Basic authentication the Web client sends the user credentials with each request, therefore, users cannot log out from the portal except by completely closing the Web client. For example, if a user logs out of the portal and leave the browser open, another user might be able to access pages that the first user visited previously.

If the HTTP Basic Authentication TAI is enabled, it decides on every incoming request whether it is responsible for the authentication of that request or not. This decision is based on black and white lists for the requested URL and the client's user agent. The TAI is responsible only if none of the patterns in the black lists match and at least one of the patterns in one of the white lists match. Therefore, if the TAI is configured with empty white lists, it will never authenticate a request.

If the TAI decides to authenticate the request and that request contains an authorization header that contains a user ID and password, the TAI tries to log on

with that credential. If no user ID and password is provided, the TAI will challenge the client according to RFC 2617.

Related information:

RFC 2617

Configuring the HTTP Basic Authentication Trust Association Interceptor

To configure the HTTP Basic Authentication Trust Association Interceptor according to your requirements, you set its properties.

About this task

Complete the following steps to set the properties of the HTTP Basic Authentication Trust Association Interceptor:

Procedure

1. Log on to the WebSphere Integrated Solutions Console.
2. Click **Security** > **Global security**.
3. Click **Web and SIP security** > **Trust association**.
4. Make sure that the **Enable trust association** check box is checked.
5. Click **OK** and then click **Save** if you checked the **Enable trust association** check box.
6. Click **Web and SIP security** > **Trust association** again.
7. Click **Interceptors** > **com.ibm.portal.auth.tai.HTTPBasicAuthTAI**.

   Tip: If `com.ibm.portal.auth.tai.HTTPBasicAuthTAI` is not available, run the following task to enable the interceptor: `enable-http-basic-auth-tai-sitemgmt`. See the related links section for instructions.
8. Click **Custom properties**.
9. Click the property you that want to change.
10. Change the value for the property according to your requirements.
11. Click **OK**.
12. Select other properties and change their values as required.
13. Click **OK**.
14. Save your changes.
15. Restart the WebSphere Application Server.

Reference: Properties for the Trust Association Interceptor

The HTTP Basic Authentication Trust Association Interceptor has several configuration properties.

You can configure the following properties for the HTTP Basic Authentication Trust Association Interceptor.

Notes:

1. The default value for each parameter is given in parentheses.
2. In the descriptions, **TAI** refers to the portal HTTP Basic Authentication Trust Association Interceptor.

enabled = (true)

Use this property to determine whether the TAI is active or not. Possible values are `true` and `false`. The default is `true`. If you set this property to `true`, the TAI authenticates requests. If you set this property to `false`, the TAI does not authenticate requests.

loginTarget = (Portal_LTPA)

Use this property to specify the alias of the JAAS login configuration that is used by the TAI. The default value is `Portal_LTPA`. By this default, the TAI uses the same JAAS login configuration as the one that is used by portal HTTP form based log in.

authenticationRealm = (WPS)

Use this property to specify the name of an authentication realm as defined in RFC 2617. The TAI challenges the client to authenticate against this realm. The default is `WPS`. By this default, the TAI uses the same authentication realm name as the one that is used by portal HTTP form based login.

userAgentBlackList = (AllAgentsAllowed)

Use this property to specify a list of patterns for which you do not want the TAI to handle the requests. Separate the patterns by whitespaces.

Every product name in the HTTP header field `User-Agent` of incoming requests is compared with each of the patterns specified for this parameter. If the TAI is enabled and the URL matches at least one of the patterns specified for the `userAgentBlackList` property, the TAI will not handle the request.

The default value is `AllAgentsAllowed`. This default value means that the user agent black list is not active.

You can specify the patterns with an asterisk (`*`) as a wild card character. You can also define the patterns as Java regular expressions. In this case set the property `useRegExp` to `true`.

urlBlackList = (/wps/myportal*)

Use this property to specify a list of patterns for which you do not want the TAI to handle the requests. Separate the patterns by whitespaces.

The full path information of the URL of the incoming request is compared with each of the patterns specified for this parameter. Before comparing the URL to the patterns, the protocol, server, port, and query information is removed from the URL. If the TAI is enabled and the URL matches at least one of the patterns specified for the `urlBlackList` property, the TAI will not handle the request. The default value is `/wps/myportal*`.

Use the following syntax rules for specifying the patterns:

- You can use URI encoded patterns. For example, if you want to use the blank character as part of a pattern, you can encode it as `%20` . It is then interpreted as part of the pattern and not as a pattern separator. Make sure that you use only characters that are valid within a URI, and encode all other characters.
- You can use an asterisk (`*`) as a wild card character.
- You can define the patterns as Java regular expressions. In this case set the property `useRegExp` to `true`.

userAgentWhiteList = (NoAgentSpecified)

Use this property to specify a list of patterns for which you want the TAI to handle the requests. Separate the patterns by whitespaces. Every

product name in the HTTP header field User-Agent of the incoming request is compared with each of the patterns specified for this parameter.

If the TAI is enabled and the pattern specified for this property has at least one match and neither of the userAgentBlackList or the urlBlackList have a match, then the TAI handles the request.

The default is NoAgentSpecified. This default value means that the user agent white list is not active.

You can specify the patterns with an asterisk (*) as a wild card character. You can also define the patterns as Java regular expressions. In this case set the property useRegExp to true.

urlWhiteList = (/wps/mycontenthander*)

Use this property to specify a list of patterns for which you want the TAI to handle the requests. Separate the patterns by whitespaces. The full path information of the URL of the incoming request is compared with each of the patterns specified for this parameter. Before comparing the URL to the patterns, the protocol, server, port, and query information is removed from the URL.

If the TAI is enabled and the pattern specified for this property has at least one match and neither of the userAgentBlackList or the urlBlackList have a match, then the TAI handles the request.

The default value is /wps/mycontenthander*.

Use the following syntax rules for specifying the patterns:

- You can use URI encoded patterns. For example, if you want to use the blank character as part of a pattern, you can encode it as %20 . It is then interpreted as part of the pattern and not as a pattern separator. Make sure that you use only characters that are valid within a URI, and encode all other characters.
- You can use an asterisk (*) as a wild card character.
- You can define the patterns as Java regular expressions. In this case set the property useRegExp to true.

Note: Values that you specify for the userAgentWhiteList or urlWhiteList properties come into effect only if all of the following conditions apply:

- The TAI is enabled by specifying enabled = true.
- Neither of the properties userAgentBlackList or urlBlackList has the default value asterisk specified. To enable the values specified for the white list properties, you can remove the asterisk from the black list properties and leave them without a specified value.

useRegExp = (false)

Use this property to determine whether or not the patterns that you specified for the black list and white list the previous properties are to be interpreted as Java regular expressions. Possible values are true or false. The default value is false. The values have the following meanings and syntax rules:

true If you set the value for this property to true, all the patterns in the black and white lists are interpreted as Java regular expressions (RegExp). For more information about Java regular expressions syntax and usage refer to http://docs.oracle.com/javase/tutorial/essential/regex/. Examples:

- `[^X]*` will match every user agent that does not contain an uppercase `X` in its product name.
- `.*my_browser.*` will match every user agent that contains `my_browser` in its product name.
- `.*%5bX%5d` is URL encoded for `.*[X]` and will match every URL that ends with `X`.

false This is the default. If the value for this property is set to `false`, all patterns support only the asterisk (*) as a wildcard character that matches against any string. The asterisk (*) wildcard can appear anywhere in the pattern. You can use multiple asterisk (*) wildcards within the same pattern.

If you want to represent an asterisk as an actual character for matches in the pattern instead of using it as a wildcard, prefix it with a backslash like this: `\*` . To represent the backslash as a character for matching, code it by using a double backslash: `\\` .

Examples:

- `*my_browser*` will match every user agent that contains `my_browser` in its product name.
- `/myprefix*mysuffix` will match every URL that starts with `/myprefix` and ends with `mysuffix`.
- `Fun\* Ag\\ent` will only match a user agent hat has `Fun* Ag\ent` as product name.

HTTP Basic Authentication Trust Association Interceptor in combination with external authentication servers

When you use the HTTP Basic Authentication Trust Association Interceptor (TAI) with external authentication servers, you can configure the TAI to work for a specific set of requests.

IBM WebSphere Application Server can have multiple TAIs registered for authentication handling. This way you can use the HTTP Basic Authentication TAI together with other TAIs, for example, the WebSEAL TAI. However, you cannot configure an invocation sequence for the TAIs that you installed. You must ensure that the TAIs handle disjoint sets of requests. The HTTP Basic Authentication TAI is configured this way by default. For an alternative HTTP Basic Authentication TAI, you can configure the TAI properties for URL pattern filtering by black and white lists. Conflicts can arise if the security server relies on Basic Authentication or if other non-standard configurations are wanted.

Normal configuration of an external authentication server protects the HTML-based entry points into the portal, for example, /wps/myportal. When the authentication server handles the initial request, shared tokens known to the authentication server are exchanged and used for subsequent requests through the authentication server. Similarly, when the request makes it to the Portal Java virtual machine, the TAIs handle the initial authentication trust request. Then, they generate an LtpaToken or LtpaToken2 shared token for subsequent requests, including REST or XML-based requests.

Setting up custom user repositories

A custom user repository is any repository that WebSphere Portal Express does not support out-of-box. However, you can configure WebSphere Portal Express to support any type of repository in a federated or stand-alone user registry, whether an LDAP directory, database, file system, and so on. Setting up custom user repositories involves tasks such as defining additional repositories to the default federated user registry, creating a custom stand-alone user repository, and updating your user repository to reflect changes in your environment. Learn what steps are required to create and update custom user repositories and what specific interfaces you must implement to enable communication between WebSphere Portal Express and a repository.

About this task

A user registry is an implementation of the `UserRegistry` interface in WebSphere Application Server. The following user registries are available out-of-box:

Federated Repositories

An implementation of the `UserRegistry` interface that supports multiple repositories. To communicate with the federated repositories, both WebSphere Application Server and WebSphere Portal Express dispatch all operations to VMM.

WebSphere Portal Express accesses all user repositories through VMM. WebSphere Portal Express uses the Portal User Management Architecture (PUMA) System Programming Interface (SPI) to retrieve and set attributes on user objects. PUMA passes these requests to VMM, which then passes the requests on to a corresponding registry adapter that connects VMM to the repository. For this reason, registry adapters are required to enable communication between WebSphere Portal Express and any repository.

Important: You must create a user registry adapter if you plan to use a custom user repository. To create a user registry adapter, implement the `com.ibm.wsspi.wim.Repository` interface. Refer to the following topics in the WebSphere Application Server documentation for information and instructions:

Repository SPI (System programming interfaces for virtual member manager adapters)

Sample custom adapters for federated repositories examples

"Creating and updating federated repositories" on page 1616
You can define additional repositories as required for the out-of-box federated repositories user registry. For example, you can define one or more databases and/or LDAP directories for the user registry. Federated repositories also let you implement multiple realms. Realms define subsets of users and are spread across multiple repositories. For example, you can define one realm in a file-based repository and another realm in an LDAP directory. Because WebSphere Application Server provides an implementation of the `UserRegistry` interface for federated repositories out-of-box, you do not need to create a custom implementation of this interface.

Related concepts:

"Virtual Member Manager integration" on page 132
IBM WebSphere Application Server includes the Virtual Member Manager (VMM), which IBM WebSphere Portal Express uses to access user and group information. VMM provides an interface that enables communication between WebSphere Portal Express and any repository, whether federated repositories or your own custom user registry.

Related information:

Websphere Application Server Information center: Sample custom adapters for federated repositories examples

WebSphere Application Server Information Center: Repository SPI

WebSphere Application Server Information Center: Virtual member manager API

Setting up a custom user repository with Virtual Member Manager for IBM WebSphere Application Server and IBM WebSphere Portal

IBM WebSphere Developer Technical Journal: Expand your user registry options with a federated repository in WebSphere Application Server, Using the Virtual Member Manager

Creating and updating federated repositories

You can define additional repositories as required for the out-of-box federated repositories user registry. For example, you can define one or more databases and/or LDAP directories for the user registry. Federated repositories also let you implement multiple realms. Realms define subsets of users and are spread across multiple repositories. For example, you can define one realm in a file-based repository and another realm in an LDAP directory. Because WebSphere Application Server provides an implementation of the `UserRegistry` interface for federated repositories out-of-box, you do not need to create a custom implementation of this interface.

"Creating and configuring federated repositories"
Add your custom repository to the default federated repositories. You must implement a custom registry adapter so that WebSphere Portal Express can access the repository. After you implement your custom registry adapter, you must define several parameters for your environment and run a task to add your repository to the federated repositories.

"Updating federated repositories" on page 1618
You can update the environment parameters you defined when you created the federated repository. You should run the task to update your custom repository if you any environment settings have changed such as the base DN. You can also run the update task if you want to change any settings you specified when you created the repository. For example, run the update task if you want to change support for paging, sorting, and transactions.

Creating and configuring federated repositories

Add your custom repository to the default federated repositories. You must implement a custom registry adapter so that WebSphere Portal Express can access the repository. After you implement your custom registry adapter, you must define several parameters for your environment and run a task to add your repository to the federated repositories.

Before you begin

You must create a custom user registry adapter before you create your repository. To create the user registry adapter, implement the `wim.Repository` interface. Refer to the WebSphere Application Server documentation for information and instructions.

About this task

Procedure

1. Perform the following steps to create a federated repository:
   a. Open `wkplc.properties` with any text editor from the following directory:
      Windows: *wp_profile_root*`\ConfigEngine\properties`
      Linux: *wp_profile_root*`/ConfigEngine/properties`
      IBM i: *wp_profile_root*`/ConfigEngine/properties`
   b. Specify values for the following parameters under the **VMM Federated CUR Properties** heading:
      federated.cur.id
      federated.cur.adapterClassName
      federated.cur.baseDN
      federated.cur.isExtIdUnique
      federated.cur.supportExternalName
      federated.cur.supportPaging
      federated.cur.supportSorting
      federated.cur.supportTransactions
   c. Save and close `wkplc.properties`.
   d. Run the following task from the *wp_profile_root*`/ConfigEngine` directory:
      Windows: `ConfigEngine.bat wp-create-cur -DWasPassword=`*password*
      Linux: `./ConfigEngine.sh wp-create-cur -DWasPassword=`*password*
      IBM i: `ConfigEngine.sh wp-create-cur -DWasPassword=`*password*
   e. Restart the WebSphere_Portal server.
2. Perform the following steps to create custom properties for your federated repository:
   a. Open `wkplc.properties` with any text editor from the following directory:
      Windows: *wp_profile_root*`\ConfigEngine\properties`
      Linux: *wp_profile_root*`/ConfigEngine/properties`
      IBM i: *wp_profile_root*`/ConfigEngine/properties`
   b. Specify values for the following parameters under the **VMM Federated CUR Properties** heading in `wkplc.properties`:
      cur.id
      cur.name
      cur.value
   c. Save and close `wkplc.properties`.
   d. Run the following task from the *wp_profile_root*`/ConfigEngine` directory:
      Windows: `ConfigEngine.bat wp-create-cur-custom-property -DWasPassword=`*password*
      Linux: `./ConfigEngine.sh wp-create-cur-custom-property -DWasPassword=`*password*
      IBM i: `ConfigEngine.sh wp-create-cur-custom-property -DWasPassword=`*password*
   e. Restart the WebSphere_Portal server.

Related tasks:

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

Related information:

Websphere Application Server Information center: Sample custom adapters for federated repositories examples

WebSphere Application Server Information Center: Repository SPI

WebSphere Application Server Information Center: Virtual member manager API

Setting up a custom user repository with Virtual Member Manager for IBM WebSphere Application Server and IBM WebSphere Portal

Updating federated repositories

You can update the environment parameters you defined when you created the federated repository. You should run the task to update your custom repository if you any environment settings have changed such as the base DN. You can also run the update task if you want to change any settings you specified when you created the repository. For example, run the update task if you want to change support for paging, sorting, and transactions.

About this task

To update a federated repository, do the following:

Procedure

1. Open `wkplc.properties` with any text editor from the following directory:
   Windows: *wp_profile_root*`\ConfigEngine\properties`
   Linux: *wp_profile_root*`/ConfigEngine/properties`
   IBM i: *wp_profile_root*`/ConfigEngine/properties`
2. Specify values for the following parameters under the **VMM Federated CUR Properties** heading:
   `federated.cur.id`
   `federated.cur.adapterClassName`
   `federated.cur.baseDN`
   `federated.cur.isExtIdUnique`
   `federated.cur.supportExternalName`
   `federated.cur.supportPaging`
   `federated.cur.supportSorting`
   `federated.cur.supportTransactions`
3. Save and close `wkplc.properties`.
4. Run the following task from the *wp_profile_root*`/ConfigEngine` directory:
   Windows: `ConfigEngine.bat wp-update-federated-cur -DWasPassword=`*`password`*
   Linux: `./ConfigEngine.sh wp-update-federated-cur -DWasPassword=`*`password`*
   IBM i: `ConfigEngine.sh wp-update-federated-cur -DWasPassword=`*`password`*

5. Restart the WebSphere_Portal server.

Related tasks:

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

External security managers

Use external security managers such as IBM Security Access Manager to perform authentication and authorization for IBM WebSphere Portal Express. You can use an external security manager for authentication only or for both authentication and authorization. Using an external security manager to perform only authorization is not supported at this time.

Perform the following tasks to configure external security managers:

"Planning for external security managers" on page 1620
By default, IBM WebSphere Application Server controls authentication to IBM WebSphere Portal Express and WebSphere Portal Express controls authorization (access control) to resources.

"Enabling and configuring single sign-on for HTTP requests using SPNEGO" on page 1625
You can create single sign-on requests for your HTTP server using the Simple and Protected GSS-API Negotiation Mechanism (SPNEGO) available in IBM WebSphere Application Server. Creating single sign-on requests using SPNEGO allows HTTP users to log in and authenticate only once and receive automatic authentication from WebSphere Application Server.

"Security Access Manager" on page 1626
IBM WebSphere Portal Express supports the use of IBM Security Access Manager. Existing Security Access Manager users can use the Security Access Manager services to assist them in their deployment.

"Configuring eTrust SiteMinder" on page 1657
IBM WebSphere Portal Express supports the use of Computer Associates eTrust SiteMinder for authentication and authorization.

"Verifying Trust Association Interceptors for authentication" on page 1666
After configuring IBM WebSphere Portal Express to use an external security manager for authentication, you should verify that the Trust Association Interceptors (TAI) are working properly before continuing with any additional configuration tasks.

"Changing the login and logout pages" on page 1666
By default, when unauthenticated users attempt to access the `myportal` page, they get redirected to the login page to provide a user name and password. When using a WebSEAL or Computer Associates eTrust SiteMinder TAI for authentication, you no longer need to use the IBM WebSphere Portal Express login page. Instead, the login icon should point to the protected portal page.

"Managing access control with external security managers" on page 1668
IBM WebSphere Portal Express externalizes roles and uses access control to control role membership. From the perspective of the external security manager, these externalized roles contain only one permission: membership in the role. WebSphere Portal Express always determines the permissions associated with each role.

Planning for external security managers

By default, IBM WebSphere Application Server controls authentication to IBM WebSphere Portal Express and WebSphere Portal Express controls authorization (access control) to resources.

Note: When setting up security to use an external security manager in a cluster environment and across mixed nodes, there are additional considerations. For example, you should configure your external security manager after completing all other setup tasks, including ensuring that the cluster is functional.

"WebSphere Trust Association Interceptors"
Security Access Manager and Computer Associates eTrust SiteMinder provide Trust Association Interceptors (TAIs) that are used only as an authentication service.

"External authorization" on page 1621
WebSphere Portal Express always determines the permissions that are associated with each role, whether the role is externalized or not.

"Planning considerations for WebSEAL junctions" on page 1622
A junction acts as a single point of access into a web application network.

"Security Access Manager Permissions" on page 1625
In many installations, WebSEAL is behind Portal between the portlets and the back-end servers they access.

Related concepts:

"WebSphere Portal Express Support Statement" on page 103
This support statement proposes a revision to the definition of "supported" and "unsupported" about the various products of which IBM WebSphere Portal Express depends on for proper operation.

WebSphere Trust Association Interceptors

Security Access Manager and Computer Associates eTrust SiteMinder provide Trust Association Interceptors (TAIs) that are used only as an authentication service.

External authentication

TAIs can be configured with the Portal configuration tasks. The Security Access Manager TAI requires an available Security Access Manager authorization server for successful single sign-on. For information about using TAI with WebSphere Application Server, see the related links section.

Whenever a request attempts to access a secured resource, WebSphere Application Server starts the TAI. The TAI validates that the request comes from a legitimate third-party authentication proxy and returns the user's authenticated identity to WebSphere Application Server. The TAI returns either a distinguished name (DN) or a short name. WebSphere Application Server performs a registry lookup to verify the distinguished name or convert the short name to a distinguished name before searching for group memberships for that user. If the registry lookup fails, WebSphere Application Server refuses to trust the user. If the registry lookup succeeds, WebSphere Application Server generates a Lightweight Third-Party Authentication (LTPA) token for the user. It stores it as a cookie for subsequent authentication during the user's session.

A TAI is not necessary if the third-party authentication proxy provides native WebSphere Application Server identity tokens, such as LTPA tokens. Currently, only Security Access Manager WebSEAL and Security Access Manager plug-in for Edge Server provide native WebSphere Application Server identity tokens. Consult

the WebSEAL Administration Guide for more information about configuring Security Access Manager to provide LTPA tokens. The authentication proxy determines the challenge mechanism.WebSphere Portal Express relies on the authentication proxy to relay success or failure of the user identifier through the TAI or LTPA token. WebSphere Application Server sees all requests from the TAI as authenticated, but WebSphere Application Server and WebSphere Portal Express performs a look up on each user anyway. Depending on the TAI and system configuration, WebSphere Application Server and WebSphere Portal Express can be configured to look up the group also. Even if the authentication proxy has successfully authenticated, WebSphere Application Server and WebSphere Portal Express deny access if they cannot query the user in the registry. For example, a user in an External Security Manager is not accessible from WebSphere Portal Express because it is configured to a different registry. Or that registry does not have the same registry configuration properties as the External Security Manager.

Custom TAIs

TAIs that allow other custom authentication services to interact with WebSphere Application Server can be written. If you use a different security configuration, you must provide and implement a TAI to communicate with the authentication proxy.

Related information:

WebSphere Application Server Library

Single sign on to a IBM WebSphere Portal through IBM Tivoli Access Manager WebSEAL

TAM Trust Association Interceptor (TAI++)

Extended Tivoli Access Manager Trust Association Interceptor Plus (ETAI)

External authorization

WebSphere Portal Express always determines the permissions that are associated with each role, whether the role is externalized or not.

Roles are always associated with a specific resource. Resources can be moved back and forth from internal to external control with the Resource Permissions portlet. Explicit role assignments are preserved when moving in both directions. However, inherited role memberships are blocked for externalized resources. When you externalize access control for a resource, the resource is administered only through the external security manager interface. After externalizing the resource, role membership must be assigned and removed using the external security manager. The Resource Permissions portlet can no longer control user access to the resource; however, the Resource Permissions portlet can move the object back to internal control.

Note the following issues:

- Private pages cannot be externalized.
- When you use the Resource Permissions portlet to externalize or internalize access control for a resource, access control for all of its public child resources moves with it. When you use the XML configuration interface (xmlaccess) to externalize or internalize access control for a resource, access control for public child resources does not change.
- After you externalize access control for a resource, you must use the external security manager to assign users to roles on the resource.

- After access control for a resource is externalized, you can use either the Resource Permissions portlet or the XML configuration interface to create additional role types on the resource. For example, suppose that you create only the Administrator and Manager role types on the Market News Page. Then you externalize access control for the Market News Page. You must use the external security manager to assign users to the Administrator@Market News Page or Manager@Market News Page roles. If you decide that you want to assign users to the Editor@Market News Page role, which is not yet externalized, follow these steps:
  1. Use the Resource Permissions portlet to create the Editor role type for the Market News Page.
  2. Use the external security manager to assign users to the Editor@Market News Page role by editing the ACL.

  Remember that WebSphere Portal Express still determines the permissions that are associated with the externalized Editor role type.
- If a user inherits access to a resource from a parent resource, the user loses the inherited access when the resource is externalized. If the user needs access to that resource, you must assign access through the external security manager.
- The user, who externalizes the resource, automatically receives the Administrator role on the parent resource of the externalized resource tree (if using the Resource Permissions portlet) or the resource (if using the XML configuration interface).

The decision to use an external security manager must be made with the understanding that the external security manager software's ACL semantics override WebSphere Portal Express semantics. For example, if you use Security Access Manager to grant anonymous membership on a role for an externally controlled portlet, you must set the ACL for that portlet to include the Security Access Manager unauthenticated user group.

Note: If you use Security Access Manager for authorization, you must also use it for authentication. Using Security Access Manager to perform only authorization is not supported.

Planning considerations for WebSEAL junctions

A junction acts as a single point of access into a web application network.

A junction is an HTTP or HTTPS connection between a front-end WebSEAL server and a back-end web application server. WebSEAL does authentication checks on all requests for resources before it passes those requests across a junction to the back-end server.

Starting with WebSphere Portal Express version 8.0, the supported junction types between the front-end WebSEAL server and a back-end WebSphere Portal Express server are:

- Virtual host junctions - this junction type is generally supported for all use cases.
- Transparent junctions - support for Transparent junctions with WebSphere Portal Express is limited to a simple use case. For example, one WebSphere Portal Express logical instance that uses the default `/wps` context root. The logical instance might be one server or cluster or set of related farm nodes that run a single logical instance of a Portal website, which might include virtual portals. Anything more complex than this simple use case requires the use of Virtual host junctions.

Prior releases supported the use of traditional non-transparent WebSEAL junctions, but this configuration is no longer supported by WebSphere Portal Express 8.0 and later. With the proliferation of WebSphere Portal Express URLs, the virtual host junction is now the most efficient way to make a WebSphere Portal Express server work behind a WebSEAL proxy. All discussions around encryption, Trust Association Interceptor (TAI) versus LTPA and other setup options continue to be applicable. The difference is the overall junction type, which determines how the junction is visible to the users.

A traditional non-transparent junction has a token in the URL that corresponds to the junction in WebSEAL. For example, the URL might be `http://webseal.hostname.yourco.com/junction1/wps/myportal`. A transparent junction uses an existing token in the URL to identify the junction; for example, it uses the `/wps` token in `/wps/myportal`. The problem with both of these methods is that WebSphere Portal Express has many URLs and not all of them start with `/wps`. They are also difficult to configure to use a consistent prefix.

Virtual host junctions use a virtual host name to identify the junction. To identify the junction, the host name might be `junction1.webseal.hostname.yourco.com`. This junction is only an example; you can use any host name that fits within your domain. The junction is then defined in WebSEAL to use the incoming host name, instead of a URL token, to identify the junction and the corresponding back-end servers.

In the configuration that is described here, the WebSEAL component of Security Access Manager handles the user authentication. A Trust Association Interceptor (TAI) is used by WebSphere Application Server and WebSphere Portal Express to accept the identity of the user as asserted by WebSEAL.

To properly secure the WebSphere Application Server and WebSphere Portal Express system against an attack, the TAI must still authenticate the WebSEAL server. So that only requests that are legitimately presented through that WebSEAL server are accepted. There are different ways to configure this authentication between WebSEAL and the TAI in WebSphere Application Server. You can choose between the different ways to configure depending on how much effort and performance you want to put into securing your network. The decisions that you make determines how you set up the junctions between the WebSEAL server and WebSphere Portal Express.

Note: By default, the XML configuration interface cannot access WebSphere Portal Express through a WebSEAL junction. To enable the XML configuration interface to access WebSphere Portal Express through a WebSEAL junction, use Security Access Manager to define the configuration URL (`/wps/config`) within the junction as unprotected. Use the WebSEAL documentation for specific instructions about defining separate URLs within the junction and assigning separate ACLs to these URLs. After the configuration URL is defined as unprotected, only WebSphere Portal Express enforces access control to this URL. Other resources that are protected within the WebSEAL junction (for example, the `wps/myportal` URL) are still protected by WebSEAL.

Nonencrypted junction using Basic Authentication

The identity of the user must be passed to the TAI in a header that is called `iv-user`. The header is inserted by WebSEAL into the request that is sent from WebSEAL to the WebSphere Application Server and the WebSphere Portal Express servers. The junction creation option to pass the user identity is `-c iv-user`. While

WebSEAL can be configured to pass the user identity in other ways, the `iv-user` header is the only one that is supported by the TAI.

Advanced junction configurations

For more details and options about how to configure junctions between WebSEAL with WebSphere Application Server and WebSphere Portal Express, including other options for specifying the WebSEAL server identity, use the WebSEAL Administration Guide and to the documentation for the HTTP Server that you are using with WebSphere Application Server.

The junctions between WebSEAL and WebSphere Application Server and WebSphere Portal Express can be configured to be encrypted or not. Encrypted junctions enhance security by making sure that no one can eavesdrop on information that is flowing between WebSEAL, WebSphere Application Server, and WebSphere Portal Express. However, encrypted junctions require more administration to move the necessary signing certificates between the systems, and also have a performance cost. If you are not comfortable that your network between the firewalls is secure against unauthorized access and observation, you must use encrypted junctions between WebSEAL and WebSphere Application Server/WebSphere Portal Express. If you are comfortable that your network is secure against unauthorized access and observation, especially for traffic across an inner firewall, you can use unencrypted junctions between WebSEAL and WebSphere Application Server/WebSphere Portal Express.

Setting up the WebSEAL -WebSphere Application Server/WebSphere Portal Express junction over SSL requires that you configure WebSphere Application Server and the HTTP server that is used by WebSphere Application Server to accept inbound SSL traffic and route this traffic correctly to WebSphere Application Server and WebSphere Portal Express. This process includes importing the necessary signing certificates into at least the WebSEAL certificate keystore, and possibly also the HTTP server certificate keystore.

If you choose to use encrypted junctions between WebSEAL and WebSphere Application Server and WebSphere Portal Express, you can also choose to have WebSEAL identify and authenticate itself to WebSphere Application Server and the TAI by using its own client-side certificate. In this case, you can configure the TAI to not further validate the WebSEAL server, relying on the mutual SSL connection to supply a trustable identity for the WebSEAL server.

If you choose not to use client-side certificates to identify the WebSEAL server, or if you choose not to use an SSL junction, you can identify the WebSEAL server to the TAI by using a Basic Authentication (BA) header. In this case, a password is placed into the BA header, and also configured into the TAI. This action represents a "shared secret" that only the TAI and the WebSEAL server know, which allows the TAI to trust that it really is the WebSEAL server that is asserting the user's identity, and the TAI can trust it. In this case, using an SSL junction provides more security by protecting this BA header from observation, but the TAI still relies on the BA header for identifying the WebSEAL server.

To set up the junction to use the Basic Authentication header to identify the WebSEAL server, use the `-b supply` option on the junction creation command. This option causes WebSEAL to build the BA header by using the user's user ID (which is ignored by the TAI, in favor of the `iv-user` header) and the password that is configured into WebSEAL from the `webseald-`*`instance`*`.conf` file, on the `basicauth-dummy-passwd` property. The password in the `webseald-`*`instance`*`.conf`

file must match the password for the ID that is specified on the `com.ibm.websphere.security.webseal.loginid` property of the TAI startup parameters in the WebSphere Integrated Solutions Console. For example, if you specify `com.ibm.websphere.security.webseal.loginid=mistered` on the TAI startup parameters, and the password for `mistered` is `wilbur`, then you must specify `wilbur` on the `basicauth-dummy-passwd` property in `webseald-`*`instance`*`.conf` on the WebSEAL server.

Related information:

WebSEAL Administration Guide

Security Access Manager Permissions

In many installations, WebSEAL is behind Portal between the portlets and the back-end servers they access.

Portlets accessing back-end services through an external security manager

Portlets require some method of getting through the ESM to the back-end, and that the portlets can use the Portal credential vault. The portlets are themselves designed to reuse existing security tokens from the Portal security context.

Enabling and configuring single sign-on for HTTP requests using SPNEGO

You can create single sign-on requests for your HTTP server using the Simple and Protected GSS-API Negotiation Mechanism (SPNEGO) available in IBM WebSphere Application Server. Creating single sign-on requests using SPNEGO allows HTTP users to log in and authenticate only once and receive automatic authentication from WebSphere Application Server.

About this task

To enable SPNEGO Web authentication, complete the steps in the following topic, available in the WebSphere Application Server documentation, *Enabling and configuring SPNEGO Web authentication using the administrative console*.

For more information about SPNEGO Web authentication, go to the related topics in the WebSphere Application Server documentation.

"Re-enabling the SPNEGO TAI" on page 1626
The IBM WebSphere Portal Express installation removes the Simple and Protected GSS-API Negotiation Mechanism trust association interceptor (SPNEGO TAI) from the list of available trust association interceptors. For this reason, you need to re-enable the SPNEGO TAI. You do not need to complete this task if you plan to use SPNEGO Web authentication.

Related information:

WebSphere Application Server Information Center: Configure and enable SPNEGO web authentication using the administrative console on your WebSphere Application Server machine

WebSphere Application Server Information Center: Single sign-on for HTTP requests using SPNEGO web authentication

WebSphere Application Server Information Center:Creating a single sign-on for HTTP requests using the SPNEGO TAI (deprecated)

Re-enabling the SPNEGO TAI

The IBM WebSphere Portal Express installation removes the Simple and Protected GSS-API Negotiation Mechanism trust association interceptor (SPNEGO TAI) from the list of available trust association interceptors. For this reason, you need to re-enable the SPNEGO TAI. You do not need to complete this task if you plan to use SPNEGO Web authentication.

Procedure

1. Log on to the WebSphere Integrated Solutions Console.
2. Click **Security** > **Global security**.
3. Click **Trust association** under **Web and SIP security**.
4. Ensure that the **Enable trust association** check box is checked and then click **Interceptors**.
5. Click **New** and then type `com.ibm.ws.security.spnego.TrustAssociationInterceptorImpl` in the **Interceptor class name** text field.
6. Click **OK** and then click the **Save** link to save changes to the master configuration.

Security Access Manager

IBM WebSphere Portal Express supports the use of IBM Security Access Manager. Existing Security Access Manager users can use the Security Access Manager services to assist them in their deployment.

About this task

You can use the following services:

- WebSEAL Single Signon (SSO) for authentication
- Protected Object Space and Access Control List Management for authorization
- Global Sign-on (GSO) lockbox credential vault integration
- Automatic user provisioning from WebSphere Portal Express self-registration to Security Access Manager

Perform the following tasks to configure Security Access Manager:

"Configuring Security Access Manager" on page 1627
WebSphere Portal Express can be integrated with IBM Security Access Manager to provide authentication services, authorization, and to link WebSphere Portal Express's credential vault to the ISAM GSO Lockbox feature.

"Enabling user provisioning" on page 1653
When users are created in WebSphere Portal Express, they are not automatically imported into Security Access Manager. Enabling automatic user provisioning to Security Access Manager changes this behavior. After this feature is enabled, users are automatically imported into Security Access Manager whenever they are created in WebSphere Portal Express. When user provisioning is enabled, anyone with access to the public URL can become an active user in Security Access Manager if the self-registration feature remains enabled.

"Verifying external authorization to Security Access Manager" on page 1655
After configuring IBM WebSphere Portal Express to use Security Access Manager for externalized authorization, you should verify that it works properly before continuing with any additional configuration tasks.

"Removing the Credential Vault adapter" on page 1656
If you no longer require the use of the credential vault adapter that you created, you can remove it from your configuration.

"Disabling user provisioning" on page 1657
After you enable and use the user provisioning feature within IBM Security Access Manager, you can disable the feature.

Configuring Security Access Manager

WebSphere Portal Express can be integrated with IBM Security Access Manager to provide authentication services, authorization, and to link WebSphere Portal Express's credential vault to the ISAM GSO Lockbox feature.

About this task

Authentication, Authorization and Credential Vault features can be configured in the following combinations:

- Authentication can be configured either with or without the other features
- Credential Vault integration can be configured either with or without the other features
- Authorization cannot be configured without configuring Authentication.

As part of the Authentication integration, you can also configure WebSphere Portal Express user provisioning to fully activate the created users as Security Access Manager users. By default, users that are created in LDAP by WebSphere Portal Express are not Security Access Manager users. This configuration is necessary only if you do not have an enterprise Identity Management system and provisioning process that is integrated with IBM Security Access Manager, and are using WebSphere Portal Express as your user creation tool.

Important information about authentication: To integrate WebSphere Portal Express and IBM Security Access Manager for authentication, you must create one or more junctions in WebSEAL that points to WebSphere Portal Express. Starting with WebSphere Portal Express Version 8.0, the type of junction that is supported depends on your specific use case:

Table 230. Use cases for junction type

Use case	Supported junction type
Simple use case A single, logical WebSphere Portal Express instance behind the WebSEAL layer, by using the default `/wps` context root. The WebSphere Portal Express instance can be one of the following deployments: • A stand-alone server • A single cluster • A common set of Portal instances in a farm	Either a transparent junction or a virtual host junction can be used. The junctions can be either TCP or SSL. They can use a TAI in WebSphere, or generate LTPA tokens in WebSEAL for identity assertion. Not all WebSphere Portal Express URLs start with `/wps`. Therefore, if you use transparent junctions, you must configure multiple transparent junctions to get all requests passed back to WebSphere Portal Express from WebSEAL. To avoid this complication, use a single virtual host junction. **Tip:** If you plan to change the WebSphere Portal Express context root, use virtual host junctions.

Table 230. Use cases for junction type (continued)

Use case	Supported junction type
Other use cases Anything other than a simple use case.	The supported junction type for the general case is virtual host junctions. The virtual host junctions can be either TCP or SSL. They can use a TAI in WebSphere, or generate LTPA tokens in WebSEAL for identity assertion.

Integrating WebSEAL and WebSphere Portal by using virtual portals:

Virtual Portals can be defined and identified in an incoming request by using either a token in the URL, or a virtual host name. If the URL token is used, it comes immediately after the servlet mapping of the URL, for example the `portal` or `myportal` token. If a virtual host name is used, then the host name for a request that targets the virtual portal has a different host name than requests to other virtual portals or the base portal.

When WebSphere Portal Express , using the virtual hostname-defined virtual portals, is integrated behind WebSEAL as a proxy, the configuration is to have one virtual host junction in WebSEAL, per virtual portal in WebSphere Portal Express (one to one in both directions). In addition, the virtual host junction host name in WebSEAL must be identical to the corresponding virtual portal host name on WebSphere Portal. The virtual host junction itself can be defined by using either the virtual portal host name identical to the virtual host junction host name, or the real portal or HTTP server host name, as the backend server host (the value of the **-h** parameter on the junction definition). It is better to use the virtual portal host name because some operations (such as redirect calculations) depend on the value of the HOST header, and the **-h** parameter on the junction definition causes WebSEAL to set the HOST header to this value. If the virtual portal host name is used, then either a secondary, internal DNS resolution, or manipulation of the hosts file, must be used by WebSEAL to resolve that host name to the IP address of the HTTP Server or Portal host.

Choose the appropriate tasks to configure IBM Security Access Manager:

"Security Access Manager prerequisites" on page 1629
Complete the prerequisite tasks before you configure IBM Security Access Manager.

"Creating the `PdPerm.properties` file" on page 1629
The `PdPerm.properties` file configures the Access Manager Java Run Time Environment (AMJRTE). You must create the `PdPerm.properties` file before you configure IBM Security Access Manager for authentication, authorization, Credential Vault, or user provisioning. Run the **`run-svrssl-config`** task to create the files. This task also creates the keystore file that is used to encrypt communication with Security Access Manager.

"Configuring Security Access Manager for authentication only" on page 1631
IBM WebSphere Portal Express and IBM WebSphere Application Server support the Trust Association Interceptors (TAI) that IBM Security Access Manager provides. If you use Security Access Manager for authorization, you must also use Security Access Manager for authentication. Using Security Access Manager only for authorization is not supported.

"Configuring Security Access Manager for authorization" on page 1637
You can configure IBM Security Access Manager for both authentication and authorization for IBM WebSphere Portal Express. If you configure these

functions at different times as independent tasks, configure Security Access Manager for authentication first. Using Security Access Manager only for authorization is not supported.

"Configuring the Credential Vault adapter for Security Access Manager" on page 1642
You can use IBM Security Access Manager in the IBM WebSphere Portal Express Credential Vault service. WebSphere Portal Express includes a vault adapter to access the Security Access Manager Global Sign-on (GSO) lockbox. Any existing Tivoli resource or resource credentials can be used in your portlets that access the credential vault service without any additional configuration. In addition, the credential vault service and credential vault management portlet can create or update an existing GSO lockbox entry.

"Configuring Security Access Manager for authentication, authorization, and the Credential Vault" on page 1644
You can configure Security Access Manager for authentication, authorization, and the vault adapter with one task.

"Removing Security Access Manager" on page 1652
After you install and use IBM Security Access Manager, you might find that you no longer require its use. You can then remove it from the IBM WebSphere Portal Express environment and restore authentication capabilities to IBM WebSphere Application Server and authorization capabilities to WebSphere Portal Express.

Security Access Manager prerequisites:

Complete the prerequisite tasks before you configure IBM Security Access Manager.

Complete the following task before you configure Security Access Manager:

1. Install IBM WebSphere Portal Express.
2. Run the appropriate configuration wizard option to configure WebSphere Portal Express. The stand-alone server and production server options include a database transfer and LDAP server configuration.
3. Install and configure Security Access Manager.
4. Configure the WebSEAL security proxy. Refer to the WebSEAL installation guide for information.

Tip: IBM WebSphere Application Server ships a PD.jar file that WebSphere Portal Express uses to configure Security Access Manager. The file is in the *AppServer_root*/tivoli/tam directory. WebSphere Application Server maintenance packages contain fixes for the PD.jar file. However, the Security Access Manager typically contains a newer version of the PD.jar file. Therefore, if you have access to the Security Access Manager server, use that version. After you apply the WebSphere Application Server maintenance, create a backup of the PD.jar file. Then, replace the original file with a copy of the newer version from the Security Access Manager server.

Creating the `PdPerm.properties` file:

The `PdPerm.properties` file configures the Access Manager Java Run Time Environment (AMJRTE). You must create the `PdPerm.properties` file before you configure IBM Security Access Manager for authentication, authorization, Credential Vault, or user provisioning. Run the **`run-svrssl-config`** task to create

the files. This task also creates the keystore file that is used to encrypt communication with Security Access Manager.

About this task

Cluster note: Complete these steps on every node in the cluster.

Procedure

1. Use a text editor to open the `wkplc_comp.properties` file in the *wp_profile_root*`/ConfigEngine/properties` directory.
2. Enter the following parameters in the `wkplc_comp.properties` file; go to the AMJRTE connection parameters heading:

   Cluster note: Complete this step on all nodes in the cluster. The following parameters must match on all nodes in the clustered environment. The one exception is the **wp.ac.impl.PDServerName** parameter.

   a. For **wp.ac.impl.PDAdminId**, type the user ID for the administrative Security Access Manager user.
   b. For **wp.ac.impl.PDPermPath**, type the fully qualified path and file name where the `PdPerm.properties` file is created.
   c. For **wp.ac.impl.PDServerName**, type the unique application name that is used to create a server in the Security Access Manager Policy server. The application name is an arbitrary name but must be unique for this server instance. You might want to use the node name for this WebSphere Portal Express server instance.

      Cluster note: The **wp.ac.impl.PDServerName** parameter represents an individually configured AMJRTE connection to Security Access Manager. Therefore, each node in the cluster must specify a unique value for the **wp.ac.impl.PDServerName** parameter before you run the **run-svrssl-config** task. If the cluster has four nodes, set this parameter differently on each node; for example, `amwp81`, `amwp82`, `amwp83`, and `amwp84`.
   d. For **wp.ac.impl.SvrSslCfgPort**, type the configuration port for the application name. The property is ignored by the SvrSslCfgPort.
   e. For **wp.ac.impl.SvrSslCfgMode**, type the configuration mode of the SvrSslCfg command. The only valid value is remote.
   f. For **wp.ac.impl.TamHost**, type the host name of the Security Access Manager Policy server that is used when you run **PDJrteCfg**.
   g. For **wp.ac.impl.PDPolicyServerList**, type the host name, port, and priority combinations for your Security Access Manager Policy servers that are used when you run **SvrSslCfg**.
   h. For **wp.ac.impl.PDAuthzServerList**, type the host name, port, and priority combination for your Security Access Manager authorization servers.
   i. For the **wp.ac.impl.PDKeyPath**, type the fully qualified path and file name of the location for the keystore file. This file is created when you run the **run-svrssl-config** task. The keystore file holds the keys that are used to encrypt communication between the Portal node and the Security Access Manager server.
3. Save your changes to the properties file.
4. Open a command prompt and change to the *wp_profile_root*`/ConfigEngine` directory.
5. Run the following task to create the `PdPerm.properties` file:

- Linux : `./ConfigEngine.sh run-svrssl-config -Dwp.ac.impl.PDAdminPwd=password -DWasPassword=password`
- IBM i: `ConfigEngine.sh run-svrssl-config -Dwp.ac.impl.PDAdminPwd=password -DWasPassword=password`
- Windows: `ConfigEngine.bat run-svrssl-config -Dwp.ac.impl.PDAdminPwd=password -DWasPassword=password`

Note: If the configuration task fails, validate the values in the `wkplc_comp.properties` file.
The following files are created:

- `PdPerm.properties`

  Note: This file is in the directory path you specified for the **wp.ac.impl.PDPermPath** parameter.
- `pdperm.ks`

  Note: This file is in the directory path you specified for the **wp.ac.impl.PDKeyPath** parameter.

Configuring Security Access Manager for authentication only:

IBM WebSphere Portal Express and IBM WebSphere Application Server support the Trust Association Interceptors (TAI) that IBM Security Access Manager provides. If you use Security Access Manager for authorization, you must also use Security Access Manager for authentication. Using Security Access Manager only for authorization is not supported.

About this task

Important information:

- The `pdadmin` command is a utility that supports Security Access Manager administrative functions.
- This procedure requires that you are familiar with WebSEAL administration concepts as presented in the WebSEAL Administrator Guide. For complete descriptions of all the `pdadmin` command options to create junctions, refer to the *Security Access Manager* documentation, particularly the *WebSEAL Administration Guide*.
- The following example assumes that a web server is located between the WebSEAL and WebSphere Portal Express in the request flow. Thus, the junctions that are defined in the following instructions are configured for WebSEAL to forward requests to the HTTP server and then to WebSphere Portal Express. If there is no HTTP server, modify the junction target host name and port values to enable direct communication from WebSEAL to WebSphere Portal Express.
- The following examples do not show any load balancing or other performance-related request features in WebSEAL. For more information about these advanced options, consult the *Security Access Manager* documentation.
- The following examples show simple junction creation cases. Refer to the appropriate *WebSEAL Administration Guide* and *WebSphere Application Server* documentation for information about advanced options, including generating WebSEAL LTPA Tokens in WebSEAL for SSO to WebSphere Application Server.

Clustered environments: Complete the **validate-pdadmin-connection** task on all nodes in the cluster. Complete all other steps on the primary node.

Procedure

1. Start the Security Access Manager policy and authorization servers, which are mandatory for successful configuration and for single sign-on (SSO) to occur.
2. Create your junctions on the WebSEAL server. Refer to the *IBM Security Access Manager for e-business* documentation for guidance on junction creation. Complete the following steps to create a virtual host TCP junction:
   a. Open a `pdadmin` command from any node that has a Security Access Manager run time component installed. You can use the Security Access Manager Server node, WebSEAL node, or the WebSphere Portal Express node.
   b. The general format for the `pdadmin` command to create a virtual host junction is

      ```
      pdadmin> server task WebSEAL-instance_name-webseald-WebSEAL-HostName virtualhost create -t t
      ```

      The following information describes the mandatory parameters in the `pdadmin` command:

      - The *WebSEAL-instance_name*-webseald-*WebSEAL-HostName* has three parts, as documented in the WebSEAL Administration Guide:
        1) The configured name of a single WebSEAL instance, for example *web 1*
        2) The literal string `-websealed-`
        3) The host name, for example, *webseal.yourco.com*

        The resulting combination would be *web 1*-websealed-*webseal.yourco.com*. You can use the `pdadmin server list` command to display the correct format of the server name.
      - The virtual host label (`vhost-label`) is the name for the virtual host junction.
        - Virtual host junctions are always mounted at the root of the WebSEAL object space.
        - You can refer to a junction in the `pdadmin` utility with this label.
        - The virtual host junction label must be unique within each instance of WebSEAL.
        - Because the label represents virtual host junctions in the protected object space, the label name must not contain the forward slash character (/).
      - **`-t type`**: This parameter defines whether the junction is encrypted (`-t ssl`) or not encrypted (`-t tcp`). This parameter is mandatory when you create a virtual host junction. For more information about other possible values, see the *WebSEAL Administration Guide*.
      - **`-h hostname`**: This parameter defines the backend server to which the junction connects. In most situations, the host name is the HTTP server that sits in front of WebSphere Portal Express. This parameter is mandatory when you create a virtual host junction.

      The *[options]* includes the following parameters:

      - **`-p port`**: This parameter defines the port number for the backend server to which the junction connects. If not specified, the default value is `80` for HTTP or `443` for HTTPS. It is best to specify this value explicitly in the junction creation command even if the default values are in use.
      - **`-v vhost_name[:port]`**: This parameter is the virtual host name and port number that defines the junction. WebSEAL maps incoming requests to this host name and port to this junction. If not specified, the values default to the **`-h hostname`** and **`-p port`** values.

- **-c header_type**: This parameter inserts the Security Access Manager client identity in HTTP headers across the junction. The **header_type** argument can include any combination of the following Security Access Manager HTTP header types:
  - {iv_user|iv_user-l}
  - iv_groups
  - iv_creds
  - all

  The header types must be comma-separated, and cannot have a space between the types. For example: **-c iv_user,iv_groups**. Specifying **-c all** is the same as specifying **-c iv_user,iv_groups,iv_creds**. This parameter is valid for all junctions except for the type of local. The setting here depends on how you want your TAI running within WebSphere Application Server to operate. In certain modes, the TAI might be looking for the presence of one or more of these headers. The TAI looks for these headers to know that it must claim the request when interrogated by WebSphere Application Server security. This setting must be set to match what the TAI is looking for. Consult your WebSphere system administrator if you are in doubt as to how the TAI is configured.
- **-b**: This option controls how WebSEAL passes authentication information to the backend server. Usually this setting depends on how you want the TAI to be configured in WebSphere to validate a trust relationship with WebSEAL. The usual option that is chosen is **-b supply**. For more information, see the *WebSEAL Administration Guide* or the *ETAI installation and configuration* documentation.
- **-k**: This option controls whether WebSEAL includes its own session cookie in the request to the backend server. In some situations, sending the WebSEAL session cookie to the backend server is necessary. This action is necessary to support single sign-on from WebSphere Portal Express to other backend services where WebSEAL also protects those backend services.
-

  Note: Junctions to WebSphere Portal Express whether direct or through an HTTP server does not support the **-q option** the query_contents function. Query_contents is not possible on WebSphere Portal Express

The following information is a sample command to create a virtual host TCP junction, on the *web 1* WebSEAL instance that is running on a host *webseal.yourco.com*, for the virtual host name *portalvhost.yourco.com* running on port 80 that requires a TAI in WebSphere Application Server. The virtual host junction is labeled *vhost_junction_portal_1*. The virtual host junction host name must be mapped in DNS to the WebSEAL server. The portal or http server is running on host *portal.yourco.com* and is using port 8080:

```
pdadmin> server task web1-webseald-webseal.yourco.com virtualhost create -t tcp -v portalv
```

3. Optional: If you plan to use an SSL junction, more steps are needed before you can create the junction. The necessary key and truststore must be set up with the correct certificates to enable SSL. Follow the instructions in steps 1 - 3 of the topic about *configuring SSL*. Then, complete the following steps to create the virtual host junction:

a. Use the IBM Key Management utility to load the web server certificate into the key ring for the appropriate instance of WebSEAL. See the *HTTP Server* documentation for more details.
b. Restart WebSEAL.
c. Follow the steps that are mentioned earlier to create the junction. But change the **-t** value to `ssl` and add the appropriate set of options from the Mutually Authenticated SSL junctions portion of the WebSEAL Administration Guide: **-B**, **-D**, **-K**, **-U**, and **-W**.

4. Enter the following tasks on the `pdadmin` command to create the trusted user account:

   Tip: This step is mandatory for TAI junctions only. Skip this step if you created an LTPA junction. An LTPA junction is created when you use the **-A** parameter. Refer to the *Security Access Manager for e-business* documentation for this advanced configuration.

   The trusted user account in the Security Access Manager user registry must be the same as the one that the TAI within WebSphere Application Server is configured to use. It is the ID that WebSEAL uses to identify itself to WebSphere Application Server by using the **-b supply** option, and it is one of the underlying TAI security requirements.

   Note: To prevent potential vulnerabilities, do not use the `sec_master` or *`wpsadmin`* users for the trusted user account. The trusted user account must be a dedicated user account for the purposes of communication between WebSEAL and the TAI.

   a. `pdadmin> user create` *`webseal_userid webseal_userid_DN firstname surname password`*
   b. `pdadmin> user modify` *`webseal_userid`* `account-valid yes`

5. **Clustered environments:** Complete this step on all nodes.
   Run the following task in the *wp_profile_root*`/ConfigEngine` directory to validate that the `PdPerm.properties` file is correct and that communication between WebSphere Portal Express and the Security Access Manager server works:

   Tip: Run the **validate-pdadmin-connection** task on the WebSphere Portal Express node or on each node in a clustered environment. In a clustered environment, **WasPassword** is the Deployment Manager administrator password. The **wp.ac.impl.PDAdminPwd** is the Security Access Manager administrative user password.

Table 231. Task to validate that the `PdPerm.properties` file exists by operating system

Operating system	Task
IBM i	`ConfigEngine.sh validate-pdadmin-connection -DWasPassword=`*`password`* `-Dwp.ac.impl.PDdAdminPwd=`*`password`*
Linux	`./ConfigEngine.sh validate-pdadmin-connection -DWasPassword=`*`password`* `-Dwp.ac.impl.PDAdminPwd=`*`password`*
Windows	`ConfigEngine.bat validate-pdadmin-connection -DWasPassword=`*`password`* `-Dwp.ac.impl.PDAdminPwd=`*`password`*

If the task does not run successfully: Run the **run-svrssl-config** task to create the properties file. For information, refer to *Creating the PdPerm.properties file*. Then, run the **validate-pdadmin-connection** task again. If the task is not successful after a second attempt, do not proceed with any subsequent steps. The fact that the task does not run successfully indicates that your portal cannot connect to the Security Access Manager server. Troubleshoot the connectivity issue between your portal instance and the Security Access Manager server.

6. If you are using junctions that require a Trust Association Interceptor in WebSphere Application Server, you must install and configure the TAI if it is not already been set up. The Extended TAI implementation is new from IBM Security, which must be downloaded and installed from *IBM Security website*. The older TAIs that are included with WebSphere Application Server are deprecated and must not be used.
7. The WebSphere Portal Express ConfigEngine tasks that are used to configure TAIs in previous releases, are no longer supported by the new ETAI, and cannot be used to configure the new ETAI. Follow the directions in the documentation that accompanies the download. Install and configure the Extended TAI implementation into WebSphere Application Server, including specifying the necessary custom properties that control the operation of the ETAI.
8. If for some reason you must use the deprecated TAI implementation, complete the following steps:
   a. Use a text editor to open the `wkplc_comp.properties` file in the *wp_profile_root*`/ConfigEngine/properties` directory. Enter the following parameters under the WebSphere Application Server WebSEAL TAI parameters heading:
   b. Add the **TAMTAIName** parameter to the WebSphere Application Server WebSEAL TAI section.
   c. Enter `com.ibm.ws.security.web.TAMTrustAssociationInterceptorPlus` as the value.
   d. For **wp.ac.impl.TAICreds**, type the headers that are inserted by WebSEAL that the TAI uses to identify the request as originating from WebSEAL. Refer to the values entered for the **-c header_type** parameter. For example, if you entered **-c iv-user**, then the value for **wp.ac.impl.TAICreds** is `iv-user`. If you entered **-c all**, then the value for **wp.ac.impl.TAICreds** is `iv-user,iv-groups,iv-creds`.

      Important: Never specify a header name for **wp.ac.impl.TAICreds** that the WebSEAL server is not sending over the junction.
   e. For **wp.ac.impl.hostnames**, enter the fully qualified URL for WebSphere Portal Express. This value must match the **-h** and **-p** parameters from the junction creation command.
   f. For **wp.ac.impl.ports**, enter the port number that is used to access the host server that is identified in **wp.ac.impl.hostnames**. This value must match the **-p** parameter from the junction creation command.
   g. For **wp.ac.impl.loginId**, enter the reverse proxy identity that is used when you create a TCP junction. This value must match the trusted user account.
   h. For **wp.ac.impl.BaUserName**, enter the reverse proxy identity that is used when you create an SSL junction.
   i. For **wp.ac.impl.BaPassword**, enter the password for the SSL junction reverse proxy ID.

      Save your changes to the properties file.

j. Run the following task to configure TAI for Security Access Manager:

Clustered environments: The **WasPassword** is the Deployment Manager administrative password. The `wp.ac.impl.PDAdminPWD` is the Security Access Manager administrative password.

Table 232. Task to configure TAI for Security Access Manager by operating system

Operating system	Task
IBM i	`ConfigEngine.sh enable-tam-tai -DWasPassword=`*password* `-Dwp.ac.impl.PDAdminPwd=`*password*
Linux	`./ConfigEngine.sh enable-tam-tai -DWasPassword=`*password* `-Dwp.ac.impl.PDAdminPwd=`*password*
Windows	`ConfigEngine.bat enable-tam-tai -DWasPassword=`*password* `-Dwp.ac.impl.PDAdminPwd=`*password*

9. Optional: Enable user provisioning. You must do this task only if you are using WebSphere Portal Express to create and provision new users directly in LDAP, and you need these users to also be recognized by Security Access Manager. In an enterprise deployment of WebSphere Portal Express this task would be unusual, as most large deployments have a separate user provisioning process, perhaps by using IBM Security Identity Manager. WebSphere Portal Express reads from LDAP but does not create new users. For more information, see the related links section.
10. If you are using Security Access Manager integrated with WebSphere Portal Express in a stand-alone environment that does not include a web server between WebSEAL and Portal, complete the following steps:
    a. Log on to the WebSphere Integrated Solutions Console.
    b. Go to **Servers** > **Server Types** > **Web application servers** > **WebSphere_Portal** > **Web container settings** > **Web Container** and then click **Additional Properties** > **Custom properties**.
    c. Click **New** and then add the **com.ibm.ws.webcontainer.extracthostheaderport** custom property with a value of `true`.
    d. Click **OK**.
    e. Click **New** and add the **trusthostheaderport** custom property with a value of `true`.
    f. Click **OK**.
    g. Click **Save** to save your changes.
    h. Log out of the WebSphere Integrated Solutions Console.
11. Stop and restart the appropriate servers to propagate the changes. For specific instructions, see “Starting and stopping servers, deployment managers, and node agents” on page 1216.
12. Go to the WebSEAL node and edit the `webseald-`*instance*`.conf` file for the appropriate WebSEAL instance. An example is `webseald-default.conf`. This file sets the `basicauth-dummy-passwd` value to the password for the ID that WebSEAL uses to identify itself to WebSphere Application Server. This password is the trusted user ID and password that were created in an earlier step. Stop and start the WebSEAL server before you continue.

13. If your WebSEAL instance is on the Windows operating system, limit the length of the generated URLs. Edit the `webseald-instance.conf` file and change the **process-root-requests** property value to `filter` to avoid problems with WebSEAL processing.
14. Import WebSphere Portal Express users and groups into Security Access Manager. Enter the following commands on the Security Access Manager administrative command, where *wpsadmin* is the user ID for the administrator, and *wpsadmins* is the administrators group name. The fully distinguished names of the user and group IDs vary depending on your LDAP settings.

```
user import wpsadmin uid=wpsadmin,cn=users,dc=ibm,dc=com
user modify wpsadmin account-valid yes
group import wpsadmins cn=wpsadmins,cn=groups,dc=ibm,dc=com
```

15. Some functions of WebSphere Portal Express require the use of the PUT, and DELETE HTTP method. By default, WebSEAL does not allow these requests. You must either allow this method at the applicable WebSEAL ACL and web server, or change the HTTP methods in the x-method-override configuration in the WebSEAL config file `webseald-instance.conf`.

Related tasks:

"Migrating Security Access Manager" on page 899
The IBM WebSphere Portal Express migration process migrates the security configurations. However, there is no provision for the automatic migration of any junction definitions that exist for the previous version of WebSphere Portal Express in WebSEAL. You must replace the old junction definitions with the new virtual host junction definitions.

"Creating the `PdPerm.properties` file" on page 1629
The `PdPerm.properties` file configures the Access Manager Java Run Time Environment (AMJRTE). You must create the `PdPerm.properties` file before you configure IBM Security Access Manager for authentication, authorization, Credential Vault, or user provisioning. Run the **run-svrssl-config** task to create the files. This task also creates the keystore file that is used to encrypt communication with Security Access Manager.

"Setting up SSL" on page 1596
Get an overview of the tasks that are required to configure SSL for IBM WebSphere Portal Express. Some of these tasks are completed on the IBM WebSphere Application Server and the web server. The steps that refer to the WebSphere Application Server and the web server are summarized here; refer to the WebSphere Application Server and the web server documentation for detailed information. Steps that are unique to WebSphere Portal Express are described in detail here.

Related information:

Extended Tivoli Access Manager Trust Association Interceptor Plus (ETAI)

WebSEAL Administration Guide

ETAI Download

Configuring Security Access Manager for authorization:

You can configure IBM Security Access Manager for both authentication and authorization for IBM WebSphere Portal Express. If you configure these functions at different times as independent tasks, configure Security Access Manager for authentication first. Using Security Access Manager only for authorization is not supported.

Before you begin

Complete the steps in "Configuring Security Access Manager for authentication only" on page 1631 before you configure Security Access Manager for authorization.

About this task

You can configure WebSphere Portal to delegate the decisions about what users or groups are granted access to Portal resources, to IBM Security Access Manager. This action is also called externalizing the access control for Portal resources. Normally these decisions are made by consulting the principal-to-role mappings that are stored in the Portal database. The following task configures the Portal code that is used to obtain access control decisions for Portal resources from Security Access Manager instead of from the Portal database. It includes configuration of properties that determine how the Portal resources are represented in the Security Access Manager protected object namespace. It also configures how permissions are represented in the Security Access Manager access control lists. After this task is run, you can use the Resource Permissions portlet or XMLAccess to place portal resources such as pages and portlets under Security Access Manager control.

Important: There are more considerations when you set up security to use an external security manager in a cluster environment and across mixed nodes. For instance, complete any configuration for an external security manager after you completed all other configuration tasks, including ensuring that the cluster is functional.

Implementation details of externalized access control:

When Portal resources are moved to Security Access Manager access control, WebSphere Portal creates entries corresponding to individual roles on the externalized resources in the Security Access Manager protected object space. The roles in this case are the Portal roles on Portal resources; for example, in simplified form, User@Welcome page or Administrator@Some Portlet.

Access Control Lists (ACLs) are attached to these Security Access Manager objects. The ACLs use the *PDAction* and *PDActionGroup* property values to determine what users are granted the various roles. WebSphere Portal security code queries Security Access Manager for the users that have the <PDAction> within <PDActionGroup> permission on entries in the Security Access Manager object space, and interprets that as granting the user the corresponding Portal role on the resource.

Any subset of Portal resources can be placed under external access control. WebSphere Portal can maintain internal control of other resources.

There are multiple entries in the Security Access Manager object space for every externalized resource, one entry per existing Portal role on that resource. Recall that in WebSphere Portal there are multiple different role types; for example, User, Privileged User, Editor, Manager, Administrator. Not every Portal role might be instantiated for every resource instance, and entries in Security Access Manager exist only if the corresponding actual role on that Portal resource exists.

Format of the entries in Security Access Manager

By default, the entries in the Security Access Manager object space have the following format:

```
<PDRoot_Value>/<Portal Rolename-on-resource>[/<EACappname_value>/<EACserverName_value>/<EACcellNam
```

By default, the `Portal Rolename-on-resource` is in the form of `<Portal_RoleType>@<Portal_Resource_Identifier>`. For example: `Administrator@VIRTUAL_EXTERNAL_ACCESS_CONTROL`.

The Security Access Manager object space entries might be different from the default, depending on the following properties:

- If the reorderRoles property is set to `true`, the `Portal Rolename-on-resource` displays as `Portal_Resource_Identifier@Portal_Roletype`. For example, `VIRTUAL_EXTERNAL_ACCESS_CONTROL@Administrator`.
- Set all three of the `EACserverName`, `EACcellName`, and `EACappname` properties; otherwise, they are not included in the object space entries.

Procedure

1.

Clustered environments: Complete this step on all nodes.
Run the following task in the *wp_profile_root*`/ConfigEngine` directory to validate that the `PdPerm.properties` file is correct and that communication between WebSphere Portal Express and the Security Access Manager server works:

Tip: Run the **`validate-pdadmin-connection`** task on the WebSphere Portal Express node or on each node in a clustered environment. In a clustered environment, **`WasPassword`** is the Deployment Manager administrator password. The **`wp.ac.impl.PDAdminPwd`** is the Security Access Manager administrative user password.

Table 233. Task to validate that the `PdPerm.properties` file exists by operating system

Operating system	Task
IBM i	`ConfigEngine.sh validate-pdadmin-connection -DWasPassword=`*`password`* `-Dwp.ac.impl.PDdAdminPwd=`*`password`*
Linux	`./ConfigEngine.sh validate-pdadmin-connection -DWasPassword=`*`password`* `-Dwp.ac.impl.PDAdminPwd=`*`password`*
Windows	`ConfigEngine.bat validate-pdadmin-connection -DWasPassword=`*`password`* `-Dwp.ac.impl.PDAdminPwd=`*`password`*

If the task does not run successfully: Run the **`run-svrssl-config`** task to create the properties file. For information, refer to *Creating the `PdPerm.properties` file*. Then, run the **`validate-pdadmin-connection`** task again. If the task is not successful after a second attempt, do not proceed with any subsequent steps. The fact that the task does not run successfully indicates that your portal cannot connect to the Security Access Manager server. Troubleshoot the connectivity issue between your portal instance and the Security Access Manager server.

2. Update the Namespace management parameters in the `wkplc_comp.properties` file

a. For **wp.ac.impl.EACserverName**, type the Namespace context information to further distinguish externalized portal role names from other role names in the namespace.

   Note: If set, **wp.ac.impl.EACcellName** and **wp.ac.impl.EACappname** must also be set. All three parameters must be set or none of them.

b. For **wp.ac.impl.EACcellName**, type the Namespace context information to further distinguish externalized portal role names from other role names in the namespace.

   Note: If set, **wp.ac.impl.EACserverName** and **wp.ac.impl.EACappname** must also be set.

c. For **wp.ac.impl.EACappname**, type the Namespace context information to further distinguish externalized portal role names from other role names in the namespace.

   Note: If set, **wp.ac.impl.EACcellName** and **wp.ac.impl.EACserverName** must also be set.

d. For **wp.ac.impl.reorderRoles**, type `false` to keep the role order or `true` to reorder the roles by resource type first.

3. Update the following parameters in the `wkplc_comp.properties` file; go to the Portal authorization parameters heading:

   a. For **wp.ac.impl.PDRoot**, type the root object space name in the Security Access Manager namespace for the resource entries for this portal. All Portal roles are installed with this entry. For multiple profiles and portal instances that all share a common Security Access Manager instance, choose a unique name for each root object space entry. This unique name helps to easily distinguish the resources for different instances. Or use a common *PDRoot* value for all Portal instances so that all Portal roles from any instance have a common parent. You can then use the **EACappname** parameter to distinguish between instances. If it better suits your administration models, you can also mix these two approaches, by using a common *PDRoot* value for some instances, and unique *PDRoot* values for others.

   b. For **wp.ac.impl.PDAction**, type the Custom Action created by the Security Access Manager external authorization plug-in. The combination of the action group and the action determines the Security Access Manager permission string. The permission string is used to assign membership to externalized portal roles. You might want to check with your Security Access Manager administrator to determine what they want the *PDActionGroup* and *PDAction* values to be.

      Note: After you create ACLs in Security Access Manager by using a *PDAction* and *PDActionRoot* value, these values must remain constant. Changing these values after you create ACLs that use the original settings, results in losing permissions.

   c. For **wp.ac.impl.PDActionGroup**, type the Custom Action group that is created by the Security Access Manager external authorization plug-in. The combination of the action group and the action determines the Security Access Manager permission string. The permission string is used to assign membership to externalized portal roles.

Note: After you create ACLs in Security Access Manager by using a *PDAction* and *PDActionRoot* value, these values must remain constant. Changing these values after you create ACLs that use the original settings, results in losing permissions.

d. For **wp.ac.impl.PDCreateAcl**, set the value to `true` to automatically create and attach a Security Access Manager ACL when WebSphere Portal Express externalizes the roles for a resource. Set the value to `false` to not create and attach a Security Access Manager ACL when WebSphere Portal Express externalizes the roles for a resource. In this case, the Security Access Manager Administrator must manually create and attach ACLs to the object space entries for the externalized portal resources and roles. Any ACLs created manually in this way, must use the *PDAction* and *PDActionGroup* values in order for the permissions to be found.

4. Save your changes to the properties file.
5. Run the following task to enable Security Access Manager authorization:

Table 234. Enable Security Access Manager authorization tasks by operating system

Operating system	Task
IBM i	`ConfigEngine.sh enable-tam-authorization -DWasPassword=`*`password`*
Linux	`./ConfigEngine.sh enable-tam-authorization -DWasPassword=`*`password`*
Windows	`ConfigEngine.bat enable-tam-authorization -DWasPassword=`*`password`*

Clustered note: In a clustered environment, complete this step on all nodes. The **WasPassword** value is the Deployment Manager administrative password.

If the task does not run successfully: Ensure the values in the `wkplc_comp.properties` file are valid.

6. Stop and restart the appropriate servers to propagate the changes. For specific instructions, see "Starting and stopping servers, deployment managers, and node agents" on page 1216.

Results

After you complete the authorization procedure, you can then use the WebSphere Portal administration tools (the Resource Permission portlet or XMLAccess scripting) to externalize the access control decisions for Portal resources. For any resources placed under IBM Security Manager access control, the Security Access Manager protected object space contains entries for roles in the following format

PortalServer_root`/role_name/application_name/server_name/cell_name`

For example: If the **wp.ac.impl.PDRoot** value was *Portal_Instance_1* and **wp.ac.impl.EACcellName** was *Cell_A*, **Wp.ac.impl.EACserverName** was *Server_B*, and **wp.ac.impl.EACappName** was *Application_C*, then an object space entry corresponding to a Portal role name approximately looks like

`Portal_Instance_1/Administrator@VIRTUAL_EXTERNAL_ACCESS_CONTROL/Application_C/Server_B/Cell_A`.

Where the Portal role name *Administrator@VIRTUAL_EXTERNAL_ACCESS_CONTROL* is simplified from its actual appearance, which might include a generated UUID value or custom unique name.

Note: The `EACappname`, `EACserverName`, and `EACcellName` must all be specified, or none of them appears in the Security Access Manager object space entries.

Related tasks:

"Creating the `PdPerm.properties` file" on page 1629
The `PdPerm.properties` file configures the Access Manager Java Run Time Environment (AMJRTE). You must create the `PdPerm.properties` file before you configure IBM Security Access Manager for authentication, authorization, Credential Vault, or user provisioning. Run the **`run-svrssl-config`** task to create the files. This task also creates the keystore file that is used to encrypt communication with Security Access Manager.

Configuring the Credential Vault adapter for Security Access Manager:

You can use IBM Security Access Manager in the IBM WebSphere Portal Express Credential Vault service. WebSphere Portal Express includes a vault adapter to access the Security Access Manager Global Sign-on (GSO) lockbox. Any existing Tivoli resource or resource credentials can be used in your portlets that access the credential vault service without any additional configuration. In addition, the credential vault service and credential vault management portlet can create or update an existing GSO lockbox entry.

About this task

Note: Users who are storing credentials in the `accessmanagervault.properties` file must be defined in Security Access Manager as global signon (GSO) users.

Clustered note: In a clustered environment, complete steps 1 and 2 on each node. The **`WasPassword`** value is the Deployment Manager administrative password.

Procedure

1.

Clustered environments: Complete this step on all nodes.
Run the following task in the *wp_profile_root*`/ConfigEngine` directory to validate that the `PdPerm.properties` file is correct and that communication between WebSphere Portal Express and the Security Access Manager server works:

Tip: Run the **`validate-pdadmin-connection`** task on the WebSphere Portal Express node or on each node in a clustered environment. In a clustered environment, **`WasPassword`** is the Deployment Manager administrator password. The **`wp.ac.impl.PDAdminPwd`** is the Security Access Manager administrative user password.

Table 235. Task to validate that the `PdPerm.properties` file exists by operating system

Operating system	Task
IBM i	`ConfigEngine.sh validate-pdadmin-connection -DWasPassword=`*`password`* `-Dwp.ac.impl.PDdAdminPwd=`*`password`*

Table 235. Task to validate that the `PdPerm.properties` file exists by operating system (continued)

Operating system	Task
Linux	`./ConfigEngine.sh validate-pdadmin-connection -DWasPassword=`*`password`* `-Dwp.ac.impl.PDAdminPwd=`*`password`*
Windows	`ConfigEngine.bat validate-pdadmin-connection -DWasPassword=`*`password`* `-Dwp.ac.impl.PDAdminPwd=`*`password`*

If the task does not run successfully: Run the **run-svrssl-config** task to create the properties file. For information, refer to *Creating the `PdPerm.properties` file*. Then, run the **validate-pdadmin-connection** task again. If the task is not successful after a second attempt, do not proceed with any subsequent steps. The fact that the task does not run successfully indicates that your portal cannot connect to the Security Access Manager server. Troubleshoot the connectivity issue between your portal instance and the Security Access Manager server.

2. Run the following task to create and populate the *wp_profile_root*`/PortalServer/config/config/accessmanagervault.properties` file:

Table 236. Task to create and populate the properties file by operating system

Operating system	Task
IBM i	`ConfigEngine.sh enable-tam-vault -DWasPassword=`*`password`* `-Dwp.ac.impl.PDAdminPwd=`*`password`*
Linux	`./ConfigEngine.sh enable-tam-vault -DWasPassword=`*`password`* `-Dwp.ac.impl.PDAdminPwd=`*`password`*
Windows	`ConfigEngine.bat enable-tam-vault -DWasPassword=`*`password`* `-Dwp.ac.impl.PDAdminPwd=`*`password`*

3. Complete the following steps to set the value for the **systemcred.dn** property:

   Note: The **systemcred.dn** property defines the distinguished name of the vault administrative user. All system credentials are stored under the user account. For Security Access Manager, this user must be an existing Security Access Manager user. The Security Access Manager adapter checks if the user exists in Security Access Manager before the slots are accessed.

   a. Log on to the WebSphere Integrated Solutions Console.
   b. Go to **Resources** > **Resource Environment** > **Resource Environment Providers**.
   c. Click **WP CredentialVaultService**.
   d. Under **Additional Properties**, click **Custom properties**.
   e. Edit the **systemcred.dn** property. Set the value to an existing Security Access Manager user.

4. Stop and restart the appropriate servers to propagate the changes. For specific instructions, see "Starting and stopping servers, deployment managers, and node agents" on page 1216.
5. Create a Credential Vault segment and slot to be used by Security Access Manager:

a. Click the **Administration menu** icon. Then, click **Access** > **Credential Vault**. Then, click **Add a vault segment**.
b. Select the **AccessManager** vault from the **Vaults** list, by default it is named `AccessManager`.
c. Enter a **Vault segment name** and click **OK**.
d. Click **Add a vault slot**.
e. Select the `AccessManager` vault from the **Vault** menu.
f. Enter a **Name** for the vault slot and click **OK**.

6. Optional: Use the WebSphere Application Server encoding mechanism to mask the passwords in the `accessmanagervault.properties` file and the Security Access Manager administrative password in the **pdpw** property.

Related tasks:

"Creating the `PdPerm.properties` file" on page 1629
The `PdPerm.properties` file configures the Access Manager Java Run Time Environment (AMJRTE). You must create the `PdPerm.properties` file before you configure IBM Security Access Manager for authentication, authorization, Credential Vault, or user provisioning. Run the **run-svrssl-config** task to create the files. This task also creates the keystore file that is used to encrypt communication with Security Access Manager.

Related information:

Encoding passwords in files

Configuring Security Access Manager for authentication, authorization, and the Credential Vault:

You can configure Security Access Manager for authentication, authorization, and the vault adapter with one task.

About this task

Procedure

1. Start the Security Access Manager policy and authorization servers, which are mandatory for successful configuration and for single sign-on (SSO) to occur.
2. Create your junctions on the WebSEAL server. Refer to the *IBM Security Access Manager for e-business* documentation for guidance on junction creation. Complete the following steps to create a virtual host TCP junction:
   a. Open a `pdadmin` command from any node that has a Security Access Manager run time component installed. You can use the Security Access Manager Server node, WebSEAL node, or the WebSphere Portal Express node.
   b. The general format for the `pdadmin` command to create a virtual host junction is

      ```
      pdadmin> server task WebSEAL-instance_name-webseald-WebSEAL-HostName virtualhost create -t t
      ```

      The following information describes the mandatory parameters in the `pdadmin` command:
      - The *WebSEAL-instance_name*-webseald-*WebSEAL-HostName* has three parts, as documented in the WebSEAL Administration Guide:
        1) The configured name of a single WebSEAL instance, for example *web 1*
        2) The literal string `-websealed-`

3) The host name, for example, *webseal.yourco.com*

The resulting combination would be *web 1*-websealed-*webseal.yourco.com*. You can use the pdadmin server list command to display the correct format of the server name.

- The virtual host label (vhost-label) is the name for the virtual host junction.
  - Virtual host junctions are always mounted at the root of the WebSEAL object space.
  - You can refer to a junction in the pdadmin utility with this label.
  - The virtual host junction label must be unique within each instance of WebSEAL.
  - Because the label represents virtual host junctions in the protected object space, the label name must not contain the forward slash character (/).
- **-t type**: This parameter defines whether the junction is encrypted (-t ssl) or not encrypted (-t tcp). This parameter is mandatory when you create a virtual host junction. For more information about other possible values, see the *WebSEAL Administration Guide*.
- **-h hostname**: This parameter defines the backend server to which the junction connects. In most situations, the host name is the HTTP server that sits in front of WebSphere Portal Express. This parameter is mandatory when you create a virtual host junction.

The *[options]* includes the following parameters:

- **-p port**: This parameter defines the port number for the backend server to which the junction connects. If not specified, the default value is 80 for HTTP or 443 for HTTPS. It is best to specify this value explicitly in the junction creation command even if the default values are in use.
- **-v vhost_name[:port]**: This parameter is the virtual host name and port number that defines the junction. WebSEAL maps incoming requests to this host name and port to this junction. If not specified, the values default to the **-h hostname** and **-p port** values.
- **-c header_type**: This parameter inserts the Security Access Manager client identity in HTTP headers across the junction. The **header_type** argument can include any combination of the following Security Access Manager HTTP header types:
  - {iv_user|iv_user-l}
  - iv_groups
  - iv_creds
  - all

  The header types must be comma-separated, and cannot have a space between the types. For example: **-c iv_user,iv_groups**. Specifying **-c all** is the same as specifying **-c iv_user,iv_groups,iv_creds**. This parameter is valid for all junctions except for the type of local. The setting here depends on how you want your TAI running within WebSphere Application Server to operate. In certain modes, the TAI might be looking for the presence of one or more of these headers. The TAI looks for these headers to know that it must claim the request when interrogated by WebSphere Application Server security. This setting must be set to match what the TAI is looking for. Consult your WebSphere system administrator if you are in doubt as to how the TAI is configured.

- **-b**: This option controls how WebSEAL passes authentication information to the backend server. Usually this setting depends on how you want the TAI to be configured in WebSphere to validate a trust relationship with WebSEAL. The usual option that is chosen is **-b supply**. For more information, see the *WebSEAL Administration Guide* or the *ETAI installation and configuration* documentation.
- **-k**: This option controls whether WebSEAL includes its own session cookie in the request to the backend server. In some situations, sending the WebSEAL session cookie to the backend server is necessary. This action is necessary to support single sign-on from WebSphere Portal Express to other backend services where WebSEAL also protects those backend services.
-

  Note: Junctions to WebSphere Portal Express whether direct or through an HTTP server does not support the **-q option** the query_contents function. Query_contents is not possible on WebSphere Portal Express

The following information is a sample command to create a virtual host TCP junction, on the *web 1* WebSEAL instance that is running on a host *webseal.yourco.com*, for the virtual host name *portalvhost.yourco.com* running on port 80 that requires a TAI in WebSphere Application Server. The virtual host junction is labeled *vhost_junction_portal_1*. The virtual host junction host name must be mapped in DNS to the WebSEAL server. The portal or http server is running on host *portal.yourco.com* and is using port 8080:

```
pdadmin> server task web1-webseald-webseal.yourco.com virtualhost create -t tcp -v portalvho
```

3. Optional: If you plan to use an SSL junction, more steps are needed before you can create the junction. The necessary key and truststore must be set up with the correct certificates to enable SSL. Follow the instructions in steps 1 - 3 of the topic about *configuring SSL*. Then, complete the following steps to create the virtual host junction:
   a. Use the IBM Key Management utility to load the web server certificate into the key ring for the appropriate instance of WebSEAL. See the *HTTP Server* documentation for more details.
   b. Restart WebSEAL.
   c. Follow the steps that are mentioned earlier to create the junction. But change the **-t** value to `ssl` and add the appropriate set of options from the Mutually Authenticated SSL junctions portion of the WebSEAL Administration Guide: **-B**, **-D**, **-K**, **-U**, and **-W**.
4. Enter the following tasks on the `pdadmin` command to create the trusted user account:

   Tip: This step is mandatory for TAI junctions only. Skip this step if you created an LTPA junction. An LTPA junction is created when you use the **-A** parameter. Refer to the *Security Access Manager for e-business* documentation for this advanced configuration.

   The trusted user account in the Security Access Manager user registry must be the same as the one that the TAI within WebSphere Application Server is configured to use. It is the ID that WebSEAL uses to identify itself to WebSphere Application Server by using the **-b supply** option, and it is one of the underlying TAI security requirements.

Note: To prevent potential vulnerabilities, do not use the `sec_master` or *wpsadmin* users for the trusted user account. The trusted user account must be a dedicated user account for the purposes of communication between WebSEAL and the TAI.

a. `pdadmin> user create webseal_userid webseal_userid_DN firstname surname password`

b. `pdadmin> user modify webseal_userid account-valid yes`

5.

Clustered environments: Complete this step on all nodes.
Run the following task in the *wp_profile_root*`/ConfigEngine` directory to validate that the `PdPerm.properties` file is correct and that communication between WebSphere Portal Express and the Security Access Manager server works:

Tip: Run the **validate-pdadmin-connection** task on the WebSphere Portal Express node or on each node in a clustered environment. In a clustered environment, **WasPassword** is the Deployment Manager administrator password. The **wp.ac.impl.PDAdminPwd** is the Security Access Manager administrative user password.

Table 237. Task to validate that the `PdPerm.properties` file exists by operating system

Operating system	Task
IBM i	`ConfigEngine.sh validate-pdadmin-connection -DWasPassword=password -Dwp.ac.impl.PDdAdminPwd=password`
Linux	`./ConfigEngine.sh validate-pdadmin-connection -DWasPassword=password -Dwp.ac.impl.PDAdminPwd=password`
Windows	`ConfigEngine.bat validate-pdadmin-connection -DWasPassword=password -Dwp.ac.impl.PDAdminPwd=password`

If the task does not run successfully: Run the **run-svrssl-config** task to create the properties file. For information, refer to *Creating the `PdPerm.properties` file*. Then, run the **validate-pdadmin-connection** task again. If the task is not successful after a second attempt, do not proceed with any subsequent steps. The fact that the task does not run successfully indicates that your portal cannot connect to the Security Access Manager server. Troubleshoot the connectivity issue between your portal instance and the Security Access Manager server.

6. Use a text editor to open the `wkplc_comp.properties` file in the following directory:
   - Linux : *wp_profile_root*`/ConfigEngine/properties`
   - IBM i: *wp_profile_root*`/ConfigEngine/properties`
   - Windows: *wp_profile_root*`\ConfigEngine\properties`

   Clustered environments: Complete this step on all nodes.

7. Updating properties in the `wkplc_comp.properties`.
   a. Update the Namespace management parameters in the `wkplc_comp.properties` file for Advanced Security Configuration by using External Security Managers

1) For **wp.ac.impl.EACserverName**, type the Namespace context information to further distinguish externalized portal role names from other role names in the namespace.

   Note: If set, **wp.ac.impl.EACcellName** and **wp.ac.impl.EACappname** must also be set. All three parameters must be set or none of them.

2) For **wp.ac.impl.EACcellName**, type the Namespace context information to further distinguish externalized portal role names from other role names in the namespace.

   Note: If set, **wp.ac.impl.EACserverName** and **wp.ac.impl.EACappname** must also be set.

3) For **wp.ac.impl.EACappname**, type the Namespace context information to further distinguish externalized portal role names from other role names in the namespace.

   Note: If set, **wp.ac.impl.EACcellName** and **wp.ac.impl.EACserverName** must also be set.

4) For **wp.ac.impl.reorderRoles**, type `false` to keep the role order or `true` to reorder the roles by resource type first.

b. PDJrteCfg command and file system parameters

1) For **wp.ac.impl.TamHost** under the SvrSslCfg command parameter heading in the `wkplc_comp.properties` file, type the Security Access Manager Policy Server that is used when you run **PDJrteCfg**.

c. WebSphere Application Server WebSEAL TAI parameters

1) Enter the following parameter in the `wkplc_comp.properties` file; go to the WebSEAL junction parameters heading:

   Cluster note: Complete this step on all nodes in the cluster. The following parameters must match on all nodes in the clustered environment. The one exception is the **wp.ac.impl.PDServerName** parameter.

   - For **wp.ac.impl.TAICreds**, type the headers that are inserted by WebSEAL that the TAI uses to identify the request as originating from WebSEAL.

2) Enter the following parameters in the `wkplc_comp.properties` file; go to the WebSEAL TAI parameters heading:

   Cluster note: Complete this step on all nodes in the cluster. The following parameters must match on all nodes in the clustered environment. The one exception is the **wp.ac.impl.PDServerName** parameter.

   - Optional: For **wp.ac.impl.hostnames**, type the host name that sets the WebSEAL TAI's host name parameter. This value must match the **-h** and **-p** parameters from the junction creation command.
   - Optional: For **wp.ac.impl.ports**, type the port that is used to set the WebSEAL TAI's ports parameter. This value must match the **-p** parameter from the junction creation command.
   - For **wp.ac.impl.loginId**, type the reverse proxy identity that is used when you create a TCP junction. This value must match the trusted user account.

d. Update the following parameters in the `wkplc_comp.properties` file; go to the Portal authorization parameters heading:

1) For **wp.ac.impl.PDRoot**, type the root object space name in the Security Access Manager namespace for the resource entries for this portal. All Portal roles are installed with this entry. For multiple profiles and portal instances that all share a common Security Access Manager instance, choose a unique name for each root object space entry. This unique name helps to easily distinguish the resources for different instances. Or use a common *PDRoot* value for all Portal instances so that all Portal roles from any instance have a common parent. You can then use the **EACappname** parameter to distinguish between instances. If it better suits your administration models, you can also mix these two approaches, by using a common *PDRoot* value for some instances, and unique *PDRoot* values for others.
2) For **wp.ac.impl.PDAction**, type the Custom Action created by the Security Access Manager external authorization plug-in. The combination of the action group and the action determines the Security Access Manager permission string. The permission string is used to assign membership to externalized portal roles. You might want to check with your Security Access Manager administrator to determine what they want the *PDActionGroup* and *PDAction* values to be.
3) For **wp.ac.impl.PDActionGroup**, type the Custom Action group that is created by the Security Access Manager external authorization plug-in. The combination of the action group and the action determines the Security Access Manager permission string. The permission string is used to assign membership to externalized portal roles.
4) For **wp.ac.impl.PDCreateAcl**, set the value to `true` to automatically create and attach a Security Access Manager ACL when WebSphere Portal Express externalizes the roles for a resource. Set the value to `false` to not create and attach a Security Access Manager ACL when WebSphere Portal Express externalizes the roles for a resource. In this case, the Security Access Manager Administrator must manually create and attach ACLs to the object space entries for the externalized portal resources and roles. Any ACLs created manually in this way, must use the *PDAction* and *PDActionGroup* values in order for the permissions to be found.

e.

Enter the following parameters in the `wkplc_comp.properties` file; go to the Portal vault parameters heading:

Cluster note: Complete this step on all nodes in the cluster. The following parameters must match on all nodes in the clustered environment. The one exception is the **wp.ac.impl.PDServerName** parameter.

1) For **wp.ac.impl.vaultType**, type the new vault type identifier that represents the Tivoli GSO lockbox vault.
2) For **wp.ac.impl.vaultProperties**, type the file that is used to configure the vault with Security Access Manager specific user and SSL connection information.
3) For **wp.ac.impl.manageResources**, type `true` if the credential vault or any custom portlets are allowed to create new resource objects in Security Access Manager. Or type `false` to allow only the Security Access Manager administrator to define the accessible resources to associate users with from the command line or graphical user interface.
4) For **wp.ac.impl.readOnly**, type `true` to allow credential vault or any custom portlets to modify the secrets that are stored in Security Access

Manager. Or type `false` to allow only the Security Access Manager administrator to modify the secrets from the command line or graphical user interface.

8. Save your changes to the properties file.
9. The new TAI implementation version is only available as a download and must be added to the system. See the related links section to download the Extended Security Access Manager Trust Association Interceptor Plus (ETAI) and add the binary files to your environment. WebSphere Application Server deprecated the TAI implementation that is available with WebSphere Portal Express.
10. Optional: If you must use the deprecated TAI implementation, complete the following steps:
    a. Open the `wkplc_comp.properties` file.
    b. Add the **`TAMTAIName`** parameter to the `WAS WebSEAL TAI` section.
    c. Enter `com.ibm.ws.security.web.TAMTrustAssociationInterceptorPlus` as the value.
11. The WebSphere Portal Express ConfigEngine tasks that are used to configure TAIs in previous releases, are no longer supported by the new ETAI, and cannot be used to configure the new ETAI. Follow the directions in the documentation that accompanies the download. Install and configure the Extended TAI implementation into WebSphere Application Server, including specifying the necessary custom properties that control the operation of the ETAI.
12. Open a command prompt and change to the *wp_profile_root*`/ConfigEngine` directory.
13. Run the following task to enable Security Access Manager authentication, authorization, and the credential vault:
    - IBM i: `ConfigEngine.sh enable-tam-all -DWasPassword=`*`password`*
    - Linux: `./ConfigEngine.sh enable-tam-all -DWasPassword=`*`password`*
    - Windows: `ConfigEngine.bat enable-tam-all -DWasPassword=`*`password`*

    Clustered environments:

    Complete this step on all nodes.

    `WasPassword` is the Deployment Manager administrative password.

    If the task does not run successfully: Ensure the values that you specified in `wkplc_comp.properties` are valid.
14. Complete the following steps to set the value for the **`systemcred.dn`** property:

    Note: The **`systemcred.dn`** property defines the distinguished name of the vault administrative user. All system credentials are stored under the user account. For Security Access Manager, this user must be an existing Security Access Manager user. The Security Access Manager adapter checks if the user exists in Security Access Manager before the slots are accessed.
    a. Log on to the WebSphere Integrated Solutions Console.
    b. Go to **Resources** > **Resource Environment** > **Resource Environment Providers**.
    c. Click **WP CredentialVaultService**.
    d. Under **Additional Properties**, click **Custom properties**.
    e. Edit the **`systemcred.dn`** property. Set the value to an existing Security Access Manager user.

15. If you are using Security Access Manager integrated with WebSphere Portal Express in a stand-alone environment that does not include a web server between WebSEAL and Portal, complete the following steps:
    a. Log on to the WebSphere Integrated Solutions Console.
    b. Go to **Servers** > **Server Types** > **Web application servers** > **WebSphere_Portal** > **Web container settings** > **Web Container** and then click **Additional Properties** > **Custom properties**.
    c. Click **New** and then add the **com.ibm.ws.webcontainer.extracthostheaderport** custom property with a value of `true`.
    d. Click **OK**.
    e. Click **New** and add the **trusthostheaderport** custom property with a value of `true`.
    f. Click **OK**.
    g. Click **Save** to save your changes.
    h. Log out of the WebSphere Integrated Solutions Console.
16. Stop and restart the appropriate servers to propagate the changes. For specific instructions, see "Starting and stopping servers, deployment managers, and node agents" on page 1216.
17. Go to the WebSEAL node and edit the `webseald-`*instance*`.conf` file for the appropriate WebSEAL instance. An example is `webseald-default.conf`. This file sets the `basicauth-dummy-passwd` value to the password for the ID that WebSEAL uses to identify itself to WebSphere Application Server. This password is the trusted user ID and password that were created in an earlier step. Stop and start the WebSEAL server before you continue.
18. If your WebSEAL instance is on the Windows operating system, limit the length of the generated URLs. Edit the `webseald-`*instance*`.conf` file and change the **process-root-requests** property value to `filter` to avoid problems with WebSEAL processing.
19. Some functions of WebSphere Portal Express require the use of the PUT, and DELETE HTTP method. By default, WebSEAL does not allow these requests. You must either allow this method at the applicable WebSEAL ACL and web server, or change the HTTP methods in the x-method-override configuration in the WebSEAL config file `webseald-instance.conf`.

Related tasks:

"Migrating Security Access Manager" on page 899
The IBM WebSphere Portal Express migration process migrates the security configurations. However, there is no provision for the automatic migration of any junction definitions that exist for the previous version of WebSphere Portal Express in WebSEAL. You must replace the old junction definitions with the new virtual host junction definitions.

"Creating the `PdPerm.properties` file" on page 1629
The `PdPerm.properties` file configures the Access Manager Java Run Time Environment (AMJRTE). You must create the `PdPerm.properties` file before you configure IBM Security Access Manager for authentication, authorization, Credential Vault, or user provisioning. Run the **run-svrssl-config** task to create the files. This task also creates the keystore file that is used to encrypt communication with Security Access Manager.

"Setting up SSL" on page 1596
Get an overview of the tasks that are required to configure SSL for IBM WebSphere Portal Express. Some of these tasks are completed on the IBM WebSphere Application Server and the web server. The steps that refer to the WebSphere Application Server and the web server are summarized here; refer to the WebSphere Application Server and the web server documentation for detailed information. Steps that are unique to WebSphere Portal Express are described in detail here.

Related information:

Extended Tivoli Access Manager Trust Association Interceptor Plus (ETAI)

WebSEAL Administration Guide

ETAI Download

Removing Security Access Manager:

After you install and use IBM Security Access Manager, you might find that you no longer require its use. You can then remove it from the IBM WebSphere Portal Express environment and restore authentication capabilities to IBM WebSphere Application Server and authorization capabilities to WebSphere Portal Express.

About this task

Complete the following steps to remove Security Access Manager from the WebSphere Portal Express environment:

Procedure

1. Complete the following steps, from the WebSphere Integrated Solutions Console, if you configured Security Access Manager for authentication:
   a. Select **Security** > **Global security** > **Web and SIP security** > **Trust association** > **Interceptors**.
   b. Delete **com.ibm.sec.authn.tai.TAMETai** or if you are still using the deprecated Trust Association Interceptors (TAIs) implementation, delete **com.ibm.ws.security.web.TAMTrustAssociationInterceptorPlus**.
   c. Click **OK** then **Save**.
2. Optional: Complete the following steps, from the WebSphere Integrated Solutions Console, if you configured Security Access Manager for authorization:
   a. Change the **enableExternalization** property to `false` in WP AccessControlService in the Integrated Solutions Console. This action prevents the **Externalize/Internalize** icon from appearing in the Administrative Access portlet after Security Access Manager is removed.
   b. Use either the Resource Permissions portlet or the XML configuration interface to internalize any resources that Security Access Manager manages.
   c. Edit the `services.properties` file that is found in the *wp_profile_root*/`PortalServer/config/config` directory; find the value `com.ibm.wps.services.ac.ExternalAccessControlService`, and change it to `com.ibm.wps.ac.impl.ExternalAccessControlDefaultImpl`.

      Note: Complete step 2.c. on all nodes.
3. Optional: Complete the following steps to remove the credential vault adapter and its associated segments if you configured it for Security Access Manager:

a. Use the **Credential Vault portlet** to remove any segments that are added since installation.

   Note: Do not remove **DefaultAdminSegment**.

b. Remove the **Vault.AccessManager** Credential Vault adapter implementation properties; including class, config, manager, and read-only; from the portal Credential Vault Service configuration.

   Note: The **systemcred.dn** property cannot be removed.

c. Remove the accessmanagervault.properties file from the *wp_profile_root*/PortalServer/config/config directory.

   Note: Complete step 3.c. on all nodes.

4. Optional: If you enabled user provisioning, go to "Disabling user provisioning" on page 1657.
5. Optional: Restore the backup copy of the theme so that the login and logout pages restore to the default before Security Access Manager was enabled.
6. Optional: Remove all junction points, access control lists (ACLs), protected objectspace entries (POS entries), custom actions, and custom action groups.
7. Optional: Run the following task to remove the connection to Security Access Manager:
   - Linux : ./ConfigEngine.sh run-svrssl-unconfig -DWasPassword=*password* -Dwp.ac.impl.PDAdminPwd=*password* from the *wp_profile_root*/ConfigEngine directory
   - IBM i: ConfigEngine.sh run-svrssl-unconfig -DWasPassword=*password* -Dwp.ac.impl.PDAdminPwd=*password* from the *wp_profile_root*/ConfigEngine directory
   - Windows: ConfigEngine.bat run-svrssl-unconfig -DWasPassword=*password* -Dwp.ac.impl.PDAdminPwd=*password* from the *wp_profile_root*\ConfigEngine directory

   Clustered environments:

   Complete this step on all nodes.

   WasPassword is the Deployment Manager administrative password.

   Tip: If the connection still shows up after you run this task, go to Invalid or Stale Server Definitions for more information.
8. If necessary, uninstall any Security Access Manager components.
9. Stop and restart the appropriate servers to propagate the changes. For specific instructions, see "Starting and stopping servers, deployment managers, and node agents" on page 1216.

Enabling user provisioning

When users are created in WebSphere Portal Express, they are not automatically imported into Security Access Manager. Enabling automatic user provisioning to Security Access Manager changes this behavior. After this feature is enabled, users are automatically imported into Security Access Manager whenever they are created in WebSphere Portal Express. When user provisioning is enabled, anyone with access to the public URL can become an active user in Security Access Manager if the self-registration feature remains enabled.

About this task

Note: There are two ways to create users in WebSphere Portal Express:

- **Self-registration:** This feature is enabled by default.
- **Manage Users and Groups portlet:** Administrators can use this portlet to create WebSphere Portal Express users.

Complete the following steps to enable user provisioning within Security Access Manager:

Note: In a clustered environment, run the following tasks on each node in the cluster.

Procedure

1.

Clustered environments: Complete this step on all nodes.
Run the following task in the *wp_profile_root*/ConfigEngine directory to validate that the PdPerm.properties file is correct and that communication between WebSphere Portal Express and the Security Access Manager server works:

Tip: Run the **validate-pdadmin-connection** task on the WebSphere Portal Express node or on each node in a clustered environment. In a clustered environment, **WasPassword** is the Deployment Manager administrator password. The **wp.ac.impl.PDAdminPwd** is the Security Access Manager administrative user password.

Table 238. Task to validate that the PdPerm.properties file exists by operating system

Operating system	Task
IBM i	ConfigEngine.sh validate-pdadmin-connection -DWasPassword=*password* -Dwp.ac.impl.PDdAdminPwd=*password*
Linux	./ConfigEngine.sh validate-pdadmin-connection -DWasPassword=*password* -Dwp.ac.impl.PDAdminPwd=*password*
Windows	ConfigEngine.bat validate-pdadmin-connection -DWasPassword=*password* -Dwp.ac.impl.PDAdminPwd=*password*

If the task does not run successfully: Run the **run-svrssl-config** task to create the properties file. For information, refer to *Creating the PdPerm.properties file*. Then, run the **validate-pdadmin-connection** task again. If the task is not successful after a second attempt, do not proceed with any subsequent steps. The fact that the task does not run successfully indicates that your portal cannot connect to the Security Access Manager server. Troubleshoot the connectivity issue between your portal instance and the Security Access Manager server.

2. Start all servers before you run the **enable-tam-userprov** task.
3. Run the following task to enable user provisioning:

Table 239. Task to enable user provisioning by operating system

Operating system	Task
Linux	`./ConfigEngine.sh enable-tam-userprov -DPortalAdminId=`*password* `-DPortalAdminPwd=`*password* from the *wp_profile_root*`/ConfigEngine` directory
IBM i	`ConfigEngine.sh enable-tam-userprov -DPortalAdminId=`*password* `-DPortalAdminPwd=`*password* from the *wp_profile_root*`/ConfigEngine` directory
Windows	`ConfigEngine.bat enable-tam-userprov -DPortalAdminId=`*password* `-DPortalAdminPwd=`*password* from the *wp_profile_root*`\ConfigEngine` directory

4. Stop and restart the appropriate servers to propagate the changes. For specific instructions, see "Starting and stopping servers, deployment managers, and node agents" on page 1216.

Related tasks:

"Creating the `PdPerm.properties` file" on page 1629
The `PdPerm.properties` file configures the Access Manager Java Run Time Environment (AMJRTE). You must create the `PdPerm.properties` file before you configure IBM Security Access Manager for authentication, authorization, Credential Vault, or user provisioning. Run the **`run-svrssl-config`** task to create the files. This task also creates the keystore file that is used to encrypt communication with Security Access Manager.

Verifying external authorization to Security Access Manager

After configuring IBM WebSphere Portal Express to use Security Access Manager for externalized authorization, you should verify that it works properly before continuing with any additional configuration tasks.

About this task

Perform the following steps to verify that Security Access Manager is working properly:

Procedure

1. Verify that your topology matches the topology described in the protected object space. For example, ensure the value of the **`wp.ac.impl.PDroot`** parameter exists in the Security Access Manager protected object space.
2. Perform the following steps to verify that at least one user, typically the administrator, has the `Administrator@VIRTUAL/EXTERNAL ACCESS CONTROL_1` role:
   a. Enter the `pdadmin> acl show WPS_Administrator-VIRTUAL_wps-EXTERNAL_ACCESS_CONTROL_1` command on the pdadmin command line to verify that the administrator and administrator group have the `Administrator@VIRTUAL/EXTERNAL ACCESS CONTROL_1` role.
   b. Optional: Enter the following commands to add the administrator to the `Administrator@VIRTUAL/EXTERNAL ACCESS CONTROL_1` role if no entry is found:

```
pdadmin> acl modify WPS_Administrator-VIRTUAL_wps-
EXTERNAL_ACCESS_CONTROL_1 set user wpsadmin T[WPS]m
```

```
pdadmin> acl modify WPS_Administrator-VIRTUAL_wps-
EXTERNAL_ACCESS_CONTROL_1 set group wpsadmins T[WPS]m
```

where *wpsadmin* is the administrator user ID and *wpsadmins* is the administrator group.

3. Perform the following steps from the Resource Permissions portlet:
   a. Select a resource type.
   b. Click the **Assign Access** icon for the specific resource.
   c. Click the **Edit Role** icon for a role that you want to externalize.
   d. Click **Add** to explicitly assign at least one user or group to your chosen role for the resource.
   e. Click **Search for Users or User Groups** or click the pull down for the **Search by** option where the default is set to **All available** to select specific users or user groups. Then click **OK**. An informational message box should display the message that members were successfully added to the role.
   f. Optional: Explicitly assign additional roles. If you do not assign at least one user or group to each role type for the resource, you must use the external security manager interface to create this role type later. For example, if you do not assign any users or groups to the Editor role type for the resource, then you must use the external security manager interface to create the Editor role type later.
   g. Click the **Externalize** icon for the resource. These steps move every role that is defined for each resource you assigned to the Security Access Manager protected object space. One ACL is created for each externalized role.
4. Add users to the ACLs that are attached to the role types on that resource by using either the Security Access Manager GUI or the pdadmin command line.

   Remember: If you log on as an administrator to externalize resources to Security Access Manager,

   You must be a member of the `wpsadmins` group.

   The `wpsadmins` group must appear in the `VIRTUAL/EXTERNAL_ACCESS_CONTROL_1` ACL.

Removing the Credential Vault adapter

If you no longer require the use of the credential vault adapter that you created, you can remove it from your configuration.

About this task

Perform the following steps to remove the credential vault adapter:

Procedure

1. Use the **Credential Vault portlet** to remove any segments created in the Security Access Manager Vault. See the Credential Vault portlet help for more information.

   Note: Do not remove **`DefaultAdminSegment`**.
2. Remove the **`Vault.AccessManager`** Credential Vault Adapter implementation properties from the Credential Vault Segment configuration; including class, config, manager, and read only.

   Note: Do not remove the **`systemcred.dn`** parameter.

3. Remove the `accessmanagervault.properties` file from the *wp_profile_root*/`PortalServer/config/config` directory.

   Clustered environments: Perform this step on all portal nodes.

Disabling user provisioning

After you enable and use the user provisioning feature within IBM Security Access Manager, you can disable the feature.

About this task

Perform the following steps to disable user provisioning within Security Access Manager:

Important: Perform these steps on one portal node in a clustered environment.

Procedure

1. Run the following task to disable user provisioning:
   - Linux : `./ConfigEngine.sh disable-tam-userprov -DWasPassword=`*`password`* `-Dwp.ac.impl.PDAdminPwd=`*`password`* from the *wp_profile_root*`/ConfigEngine` directory.
   - IBM i: `ConfigEngine.sh disable-tam-userprov -DWasPassword=`*`password`* `-Dwp.ac.impl.PDAdminPwd=`*`password`* from the *wp_profile_root*`/ConfigEngine` directory.
   - Windows: `ConfigEngine.bat disable-tam-userprov -DWasPassword=`*`password`* `-Dwp.ac.impl.PDAdminPwd=`*`password`* from the *wp_profile_root*`\ConfigEngine` directory.

   Note: In a clustered environment **`WasPassword`** is the Deployment Manager administrative password.
2. Stop and restart the appropriate servers to propagate the changes. For specific instructions, see "Starting and stopping servers, deployment managers, and node agents" on page 1216.

Configuring eTrust SiteMinder

IBM WebSphere Portal Express supports the use of Computer Associates eTrust SiteMinder for authentication and authorization.

About this task

Before you configure eTrust SiteMinder for authentication or authorization, you must complete the following tasks:

Procedure

1. Install and configure WebSphere Portal Express, including databases and LDAP user registry.
2. Install Computer Associate's Policy Server.
3. Install the eTrust SiteMinder Software Development Kit on the same server as WebSphere Portal Express if you plan to use eTrust SiteMinder for both authentication and authorization. Refer to the eTrust SiteMinder documentation for more information.

4. Install the eTrust SiteMinder Application Server Agent. Configure the eTrust SiteMinder Trust Association Interceptor (TAI). Follow the instructions in the eTrust SiteMinder documentation

   Note: Copy the `smagent.properties` file from the eTrust SiteMinder application server agent installation directory to the *wp_profile_root*`/ConfigEngine/properties` directory. By default, the Application Server Agent installation enables agents other than the one used for authentication. These agents are not tested with WebSphere Portal Express and should be disabled. Modify the following files in the eTrust SiteMinder installation directory to set **`EnableWebAgent=`**`no`:
   By default, the Application Server Agent installation enables agents other than the one used for authentication. These agents are not tested with WebSphere Portal Express and must be disabled. Modify the following files in the eTrust SiteMinder installation directory to set **`EnableWebAgent=`**`no`:

   `AsaAgent-az.conf`

   `AsaAgent-auth.conf`
5. If you plan to use eTrust SiteMinder for both authentication and authorization, ensure that the following two files are in the WebSphere Application Server `lib/ext` directory.
   - `smjavasdk2.jar`
   - `cryptoj.jar`

   If the directory is missing the JAR files, copy them from the eTrust SiteMinder SDK `CA/sdk/java` directory.
6. Configure the security provider. Go to Configure the JVM to Use the JSafeJCE Security Provider for instructions.
7. Create and specify the following eTrust SiteMinder Domain objects if you plan to use eTrust SiteMinder for both authentication and authorization. Refer to the eTrust SiteMinder Policy Design documentation for information about how to create these objects:
   - **User Directory**: The LDAP server and suffix
   - **Authentication Scheme**: Associates with the eTrust SiteMinder realms that WebSphere Portal Express creates.

     Note: An eTrust SiteMinder realm is different from an LDAP realm or a basic authentication realm. Within the eTrust SiteMinder administrative console, a realm is an administrative object that represents a protected URL root. An example is `/wps/myportal`. eTrust SiteMinder realms in combination with eTrust SiteMinder policies determine which users and groups are allowed to go to the protected URL root and its child URL.
   - **Agent**: An eTrust SiteMinder WebAgent that is configured to support 4.x agents or a custom eTrust SiteMinder agent. The agent must have a static shared secret to allow communication with the eTrust SiteMinder Policy Server.

What to do next

Choose the appropriate task to configure eTrust SiteMinder:

"Configuring eTrust SiteMinder for authentication and authorization" on page 1659
You can configure Computer Associates eTrust SiteMinder to perform both authentication and authorization for IBM WebSphere Portal Express. Using eTrust SiteMinder to perform only authorization is not supported at this time.

"Configuring eTrust SiteMinder to perform authentication" on page 1661
IBM WebSphere Portal Express includes a configuration task called `enable-sm-tai`. This task interacts with IBM WebSphere Application Server security configuration to enable the eTrust SiteMinder TAI and to create it as one of the interceptors. You can configure eTrust SiteMinder to provide authentication independently from configuring it to provide authorization. Using it to perform authorization only is not supported at this time.

"Configuring eTrust SiteMinder to perform authorization" on page 1662
You can configure Computer Associates eTrust SiteMinder to perform authorization independently from configuring it to perform authentication. However, if you use eTrust SiteMinder to perform authorization for IBM WebSphere Portal Express, you should also use it to perform authentication. Using eTrust SiteMinder to perform only authorization is not supported at this time.

"Removing eTrust SiteMinder" on page 1665
After you have installed and used Computer Associates eTrust SiteMinder, you may find that you no longer require its use. You can then remove it from the IBM WebSphere Portal Express environment and restore authentication capabilities to IBM WebSphere Application Server and authorization capabilities to WebSphere Portal Express.

Configuring eTrust SiteMinder for authentication and authorization

You can configure Computer Associates eTrust SiteMinder to perform both authentication and authorization for IBM WebSphere Portal Express. Using eTrust SiteMinder to perform only authorization is not supported at this time.

About this task

Install Computer Associates eTrust SiteMinder Trust Association Interceptor (TAI) distribution on the same machine as WebSphere Portal Express. If you are completing this task in a clustered environment, you must install the eTrust SiteMinder TAI distribution on each node in the cluster.

Complete the following steps to configure eTrust SiteMinder for authentication and authorization:

Procedure

1. Copy the `smagent.properties` file from the eTrust SiteMinder application server agent installation directory to the *wp_profile_root*`/properties` directory.

   Clustered environments: Complete this step on all nodes.
2. By default, the Application Server Agent installation enables agents other than the one used for authentication. These agents are not tested with WebSphere Portal Express and must be disabled. Modify the following files in the eTrust SiteMinder installation directory to set **`EnableWebAgent=`**`no`:

   ```
   AsaAgent-az.conf
   AsaAgent-auth.conf
   ```

   Clustered environments: Complete this step on all nodes.
3. Update the Namespace management parameters in the `wkplc_comp.properties` file

a. For **wp.ac.impl.EACserverName**, type the Namespace context information to further distinguish externalized portal role names from other role names in the namespace.

   Note: If set, **wp.ac.impl.EACcellName** and **wp.ac.impl.EACappname** must also be set. All three parameters must be set or none of them.

b. For **wp.ac.impl.EACcellName**, type the Namespace context information to further distinguish externalized portal role names from other role names in the namespace.

   Note: If set, **wp.ac.impl.EACserverName** and **wp.ac.impl.EACappname** must also be set.

c. For **wp.ac.impl.EACappname**, type the Namespace context information to further distinguish externalized portal role names from other role names in the namespace.

   Note: If set, **wp.ac.impl.EACcellName** and **wp.ac.impl.EACserverName** must also be set.

d. For **wp.ac.impl.reorderRoles**, type `false` to keep the role order or `true` to reorder the roles by resource type first.

4. Enter the following parameters in the `wkplc_comp.properties` file; go to the SiteMinder heading:

   Clustered environments: Complete this step on all nodes.

   Cluster note: Complete this step on all nodes in the cluster. The following parameters must match on all nodes in the clustered environment. The one exception is the **wp.ac.impl.PDServerName** parameter.

   a. For **wp.ac.imp.SMDomain**, type the eTrust SiteMinder Domain containing all externalized resources.
   b. For **wp.ac.impl.SMScheme**, type the eTrust SiteMinder Authentication scheme object name to use when you create realms.
   c. For **wp.ac.impl.SMAgent**, type the agent name that is created on eTrust SiteMinder for a specific external security manager instance.
   d. For **wp.ac.impl.SMAgentPwd**, type the password for **wp.ac.impl.SMAgent**.
   e. For **wp.ac.impl.SMadminId**, type the administrative user ID that eTrust SiteMinder uses to access the eTrust SiteMinder policy server.
   f. For **wp.ac.impl.SMAdminPwd**, type the password for **wp.ac.impl.SMadminId**.
   g. For **wp.ac.impl.SMUserDir**, type the eTrust SiteMinder User Directory object that references the LDAP user registry.
   h. For **wp.ac.impl.SMFailover**, type `true` if more than one server is listed in **wp.ac.impl.SMServers** or type `false` if no additional servers are available for failover.
   i. For **wp.ac.impl.SMServers**, type a comma-delimited list of servers for the eTrust SiteMinder agent.

5. Save your changes to the properties file.
6. Run the following task to configure eTrust SiteMinder for authentication and authorization:
   - Linux : `./ConfigEngine.sh enable-sm-all` from the *wp_profile_root*`/ConfigEngine` directory
   - IBM i: `ConfigEngine.sh enable-sm-all` from the *wp_profile_root*`/ConfigEngine` directory

- Windows: `ConfigEngine.bat enable-sm-all` from the *wp_profile_root*`\ConfigEngine` directory

Clustered environments: Complete this step on all nodes.

7. Stop and restart the appropriate servers to propagate the changes. For specific instructions, see "Starting and stopping servers, deployment managers, and node agents" on page 1216.

What to do next

Depending on your configuration, the XML configuration interface might not be able to access WebSphere Portal Express through eTrust SiteMinder. To allow the XML configuration interface access, use eTrust SiteMinder to define the configuration URL (`/wps/config`) as unprotected. Refer to the eTrust SiteMinder documentation for specific instructions.

Configuring eTrust SiteMinder to perform authentication

IBM WebSphere Portal Express includes a configuration task called `enable-sm-tai`. This task interacts with IBM WebSphere Application Server security configuration to enable the eTrust SiteMinder TAI and to create it as one of the interceptors. You can configure eTrust SiteMinder to provide authentication independently from configuring it to provide authorization. Using it to perform authorization only is not supported at this time.

Before you begin

Install Computer Associates eTrust SiteMinder Trust Association Interceptor (TAI) distribution on the same machine as WebSphere Portal Express. If you are performing this task in a clustered environment, you must install the eTrust SiteMinder TAI distribution on each node in the cluster.

Important: If you have completed the TAI installation and configuration instructions included with the Computer Associates eTrust SiteMinder distribution, including registering the TAI with WebSphere Application Server, execution of this configuration task is not required.

About this task

Perform the following steps to enable the eTrust SiteMinder TAI and create a new interceptor:

Procedure

1. Copy the `smagent.properties` file from the eTrust SiteMinder application server agent installation directory to the *wp_profile_root*`/properties` directory:

   Clustered environments: Complete this step on all nodes.

2. By default, the Application Server Agent installation enables agents other than the one used for authentication. These agents are not tested with WebSphere Portal Express and must be disabled. Modify the following files in the eTrust SiteMinder installation directory to set **`EnableWebAgent=`**`no`:

   ```
   AsaAgent-az.conf
   AsaAgent-auth.conf
   ```

   Clustered environments: Complete this step on all nodes.

3. Run the following task to enable eTrust SiteMinder TAI:

- Windows: `ConfigEngine.bat enable-sm-tai -DWasPassword=password` from the *wp_profile_root*`\ConfigEngine` directory
- Linux: `./ConfigEngine.sh enable-sm-tai -DWasPassword=password` from the *wp_profile_root*`/ConfigEngine` directory

4. Stop and restart the appropriate servers to propagate the changes. For specific instructions, see "Starting and stopping servers, deployment managers, and node agents" on page 1216.
5. Go to the "Verifying Trust Association Interceptors for authentication" on page 1666 file to verify that the TAI is working properly.

What to do next

Depending on your configuration, the XML configuration interface might not be able to access WebSphere Portal Express through eTrust SiteMinder. To allow the XML configuration interface access, use eTrust SiteMinder to define the configuration URL (`/wps/config`) as unprotected. Refer to the eTrust SiteMinder documentation for specific instructions.

Configuring eTrust SiteMinder to perform authorization

You can configure Computer Associates eTrust SiteMinder to perform authorization independently from configuring it to perform authentication. However, if you use eTrust SiteMinder to perform authorization for IBM WebSphere Portal Express, you should also use it to perform authentication. Using eTrust SiteMinder to perform only authorization is not supported at this time.

About this task

Complete the following steps to configure eTrust SiteMinder for authorization:

Procedure

1. Optional: In eTrust SiteMinder version 5.5 and higher, the configuration for eTrust SiteMinder Web Agents, including shared secrets, is centrally administered and can be dynamic. You may create a new custom agent to ensure a static shared secret. Follow these steps to create a custom agent in eTrust SiteMinder:
   a. Open the eTrust SiteMinder Administration console.
   b. Select **Agent Types** from the **View** > **Agent Types** menu.
   c. Right-click **Agent Types**, and select **Create Agent Type** from the pop-up menu.
   d. Enter a **Name** and an **Action** for the new agent type. Other fields are optional.
   e. Click **OK**.
   f. Select **Agents** from the **View** > **Agents** menu.
   g. Right-click **Agent**, and select **Create Agent** to create an agent object of the new agent type.
2. Optional: Ensure that users are no longer created through WebSphere Portal Express.

   If you use eTrust SiteMinder, you probably have a user provisioning process for creating and updating users and groups and administering group membership. You will probably want to continue using that user provisioning process instead of managing your directory through WebSphere Portal Express. WebSphere Portal Express creates entries in the directory in two ways:

- Administrators can create entries with the Manage Users and Groups portlet
- Users can create entries with the self-registration screen

3. Use a text editor to open the `wkplc_comp.properties` file in the following directory:
   - Linux: *wp_profile_root*`/ConfigEngine/properties`
   - IBM i: *wp_profile_root*`/ConfigEngine/properties`
   - Windows: *wp_profile_root*`\ConfigEngine\properties`
4. Update the Namespace management parameters in the `wkplc_comp.properties` file
   a. For **wp.ac.impl.EACserverName**, type the Namespace context information to further distinguish externalized portal role names from other role names in the namespace.

      Note: If set, **wp.ac.impl.EACcellName** and **wp.ac.impl.EACappname** must also be set. All three parameters must be set or none of them.
   b. For **wp.ac.impl.EACcellName**, type the Namespace context information to further distinguish externalized portal role names from other role names in the namespace.

      Note: If set, **wp.ac.impl.EACserverName** and **wp.ac.impl.EACappname** must also be set.
   c. For **wp.ac.impl.EACappname**, type the Namespace context information to further distinguish externalized portal role names from other role names in the namespace.

      Note: If set, **wp.ac.impl.EACcellName** and **wp.ac.impl.EACserverName** must also be set.
   d. For **wp.ac.impl.reorderRoles**, type `false` to keep the role order or `true` to reorder the roles by resource type first.
5. Enter the following parameters in the `wkplc_comp.properties` file; go to the SiteMinder heading:

   Clustered environments: Complete this step on all nodes.

   Cluster note: Complete this step on all nodes in the cluster. The following parameters must match on all nodes in the clustered environment. The one exception is the **wp.ac.impl.PDServerName** parameter.
   a. For **wp.ac.imp.SMDomain**, type the eTrust SiteMinder Domain containing all externalized resources.
   b. For **wp.ac.impl.SMScheme**, type the eTrust SiteMinder Authentication scheme object name to use when you create realms.
   c. For **wp.ac.impl.SMAgent**, type the agent name that is created on eTrust SiteMinder for a specific external security manager instance.
   d. For **wp.ac.impl.SMAgentPwd**, type the password for **wp.ac.impl.SMAgent**.
   e. For **wp.ac.impl.SMadminId**, type the administrative user ID that eTrust SiteMinder uses to access the eTrust SiteMinder policy server.
   f. For **wp.ac.impl.SMAdminPwd**, type the password for **wp.ac.impl.SMadminId**.
   g. For **wp.ac.impl.SMUserDir**, type the eTrust SiteMinder User Directory object that references the LDAP user registry.

h. For **wp.ac.impl.SMFailover**, type `true` if more than one server is listed in **wp.ac.impl.SMServers** or type `false` if no additional servers are available for failover.

i. For **wp.ac.impl.SMServers**, type a comma-delimited list of servers for the eTrust SiteMinder agent.

6. If any of the policy servers listed in the **wp.ac.impl.SMServers** parameter are configured to use ports other than the defaults, you can customize the following values for each server listed:

   ipaddress.accountingPort=44441

   ipaddress.authenticationPort=44442

   ipaddress.authorizationPort=44443

   ipaddress.connectionMax=30

   ipaddress.connectionMin=10

   ipaddress.connectionStep=5

   ipaddress.timeout=60

   Where *ipaddress* is the specific server IP listed in the **wp.ac.impl.SMServers** parameter.

   Clustered environments: Complete this step on all nodes.

7. Save your changes to the properties file.
8. Run the following task to configure eTrust SiteMinder for authorization:
   - Windows: `ConfigEngine.bat enable-sm-authorization -DWasPassword=`*password* `-Dwp.ac.impl.SmAgentPw=`*password* `-Dwp.ac.impl.SmAdminPw=`*password* from the *wp_profile_root*`\ConfigEngine` directory
   - Linux: `./ConfigEngine.sh enable-sm-authorization -DWasPassword=`*password* `-Dwp.ac.impl.SmAgentPw=`*password* `-Dwp.ac.impl.SmAdminPw=`*password* from the *wp_profile_root*`/ConfigEngine` directory

   Clustered environments: Complete this step on all nodes.

9. Stop and restart the appropriate servers to propagate the changes. For specific instructions, see "Starting and stopping servers, deployment managers, and node agents" on page 1216.
10. If users other than the administrator are allowed to externalize resources, add those users to the eTrust SiteMinder realm representing the Administrator of EXTERNAL_ACCESS_CONTROL.
11. Complete the following steps from the Resource Permissions portlet:
    a. Select a resource type.
    b. Click the **Assign Access** icon for the specific resource.
    c. Click the **Edit Role** icon for a role that you want to externalize.
    d. Click **Add** to explicitly assign at least one user or group to your chosen role for the resource.
    e. Select the specific users or user groups by clicking on **Search for Users or User Groups** or clicking on the pull down for the **Search by** option where the default is set to All available. Click **OK**.
    f. An informational message box should display the message that members were successfully added to the role.
    g. Optional: Explicitly assign additional roles. If you do not assign at least one user or group to each role type for the resource, you must use the

external security manager interface to create this role type later. For example, if you do not assign any users or groups to the Editor role type for the resource, then you must use the external security manager interface to create the Editor role type later.

h. Click the **Externalize** icon for the resource. These steps move every role that is defined for each resource you assigned to the eTrust SiteMinder Policy Domain. One policy is defined for each externalized role.

12. Add users and groups to the eTrust SiteMinder policies corresponding to the appropriate roles.

Removing eTrust SiteMinder

After you have installed and used Computer Associates eTrust SiteMinder, you may find that you no longer require its use. You can then remove it from the IBM WebSphere Portal Express environment and restore authentication capabilities to IBM WebSphere Application Server and authorization capabilities to WebSphere Portal Express.

About this task

Perform the following steps to remove eTrust SiteMinder from the WebSphere Portal Express environment:

Procedure

1. Perform the following steps if you used eTrust SiteMinder for authorization:
   a. Use either the Resource Permissions portlet or, if you are set up to execute it, the XML Configuration Interface (xmlaccess) to internalize any resources managed by eTrust SiteMinder.
   b. Edit the *wp_profile_root*`/PortalServer/config/config/services.properties` file and change the value of **`com.ibm.wps.services.ac.ExternalAccessControlService`** to `com.ibm.wps.ac.impl.ExternalAccessControlDefaultImpl`.

      Note: In a cluster environment, you must edit the `services.properties` file on all nodes.

   c. Change the **`enableExternalization`** property to `false` in the External Access Control Service. This will prevent the Externalize/Internalize icon from appearing in the Administration Access portlet after removing eTrust SiteMinder.
2. If you previously disabled the ability to create users through WebSphere Portal Express, restore it. Re-enable WebSphere Portal Express auto-registration. Restore the backup copy of the theme as appropriate to remove any features specific to eTrust SiteMinder.
3. Optional: Complete the following steps to remove the eTrust SiteMinder TAI module from the WebSphere Application Server console:
   a. In the WebSphere Application Server Administration Console, click **Securiry** > **Global security** > **Web and SIP security** > **Trust association** > **Interceptors.**.
   b. Select the eTrust SiteMinder TAI module and then click **Delete**.
   c. Click **OK** and then click **Save**.
4. Stop and restart the appropriate servers to propagate the changes. For specific instructions, see "Starting and stopping servers, deployment managers, and node agents" on page 1216.
5. If necessary, uninstall any Computer Associates components.

Verifying Trust Association Interceptors for authentication

After configuring IBM WebSphere Portal Express to use an external security manager for authentication, you should verify that the Trust Association Interceptors (TAI) are working properly before continuing with any additional configuration tasks.

About this task

Complete the following steps to verify that the Trust Association Interceptors are working properly for authentication:

Procedure

1. Enter the appropriate URL in the Web browser address bar:
   - IBM Security Access Manager: `https://`*`WebSEAL_hostname:WebSEAL_port/junction`*`/wps/myportal`
   - Computer Associates eTrust SiteMinder: `http://`*`SM_agent_hostname:SM_agent_port`*`/wps/myportal`
2. Authenticate through the external security managers. After you log in, you should be directed to the secure and personalized `myportal` page. If you are directed to the login page or the public page, there is a problem with the TAI configuration. If you are using the Security Access Manager TAI (**com.ibm.sec.authn.tai.TAMETai** or **com.ibm.ws.security.web.TAMTrustAssociationInterceptorPlus**) that WebSphere Portal Express set up, ensure that the Security Access Manager authorization server is up and running.

Changing the login and logout pages

By default, when unauthenticated users attempt to access the `myportal` page, they get redirected to the login page to provide a user name and password. When using a WebSEAL or Computer Associates eTrust SiteMinder TAI for authentication, you no longer need to use the IBM WebSphere Portal Express login page. Instead, the login icon should point to the protected portal page.

About this task

Complete the following steps to change the login and logout pages:

Procedure

1. Locate the theme files that contain the login and logout links. The files that contain the login and logout links can be different, depending on the theme. In more recent themes, these links might be located in `Default.jsp`. In older themes, the links might be located in `banner.jspf`.

   Finding theme resources: See the *Location of theme resources* link in the Related section.
2. Create a backup copy of the theme file before proceeding.
3. Open the theme file and locate the section for the login button.
4. Replace the login button anchor tag that is not commented out with the following JSP fragment:

```
<%-- comment this to enable screen login --%>
            <%-- loginOnClick is provided so the client-side aggregation
theme can add this link without creating a different copy of this file.
--%>
            <portal-logic:if loggedIn="no">
```

```
            <c:if test="${empty loginOnClick}">
            <li class="wptheme-toolbar-last"><a
href='<portal-navigation:url
                 home="protected" screen="Home"/>'
<%=bidiDirAttr%>><portal-fmt:text key="link.login"
bundle="nls.engine"/></a></li>
            </c:if>
        </portal-logic:if>
        --%>
```

Note: The previous example uses the 'portal-fmt:' prefix to designate JSP tags from the tag library in `portal.tld`. Your custom JSPs might use a different tag prefix.

5. Touch the `Default.jsp` file after editing any JSP files and before any restart. This updates the timestamp on the file to the current time and will signal a recompile of `Default.jsp` to incorporate the edit changes from other JSP files. Type: `touch Default.jsp`. An alternative is to edit (open and save) `Default.jsp`, which has the same effect as the touch command.
6. Optional: Redirect the browser to navigate to the logoff page of the external security manager (ESM) after the WebSphere Portal Express logoff command executes. Learn how to invalidate the single sign on session of the ESM by reviewing the documentation provided by the ESM relating to logoff pages.

   Security Access Manager WebSEAL provides `http://webseal/pkmslogout` as a special URL to terminate the WebSEAL single sign on session

   In eTrust SiteMinder, the Web Agent configuration object contains a property named **`LogoffUri`** where you can supply a URL to terminate the eTrust SiteMinder login session

   Complete the following steps to enable WebSphere Portal Express to execute the external security manager logoff URL after completing its logoff command:

   a. Specify the following values in the *wp_profile_root*`/PortalServer/config/ConfigService.properties` file:

      `redirect.logout=``true`

      `redirect.logout.ssl=``false` or `true`, depending on your environment

      `redirect.logout.url=`*`protocol://host_name/logout_page`*

      where *protocol* is the protocol of the ESM machine: `http` or `https`, *host_name* is the fully qualified host name of the ESM machine, and *logout_page* is the ESM page that users will be directed to when they log out. Refer to the ESM Administrator's Guide for information about using logout forms.

   b. Run the following task to update the property:

Table 240. `update-properties` task and parameters by operating system

Operating system	Task
Windows:	`ConfigEngine.bat update-properties -DWasPassword=`*`password`* from the *wp_profile_root*`\ConfigEngine` directory
Linux:	`./ConfigEngine.sh update-properties -DWasPassword=`*`password`* from the *wp_profile_root*`/ConfigEngine` directory

   c. Restart the WebSphere_Portal server on the standalone server or on each cluster member.

Related concepts:

"Understanding the Portal Version 8.5 modularized theme" on page 2521
Modern websites and browsers enable incredible new capabilities that can greatly enhance your user's web experiences. However, these capabilities are not without cost in terms of large page sizes and more processing in the browser when each page is rendered. These capabilities are worth it when you need them, but removing them for an entire site or including them only on pages that take advantage of these capabilities provides for more flexibility.

Related tasks:

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

Managing access control with external security managers

IBM WebSphere Portal Express externalizes roles and uses access control to control role membership. From the perspective of the external security manager, these externalized roles contain only one permission: membership in the role. WebSphere Portal Express always determines the permissions associated with each role.

About this task

Note: It is not possible to combine the usage of externalizes roles and externalized role mappings with portal managed pages feature. Portal pages cannot be externalized when being edited within a project and externalized resources cannot be added to projects.

For example, if you externalize the Editor@Market News Page role, you must use the external security manager to edit the access control for that role. WebSphere Portal Express still determines the permissions that are associated with the Editor role type. Roles are always associated with a specific resource, so the role Editor@Market News Page contains specific permissions on the Market News Page only. Use the Resource Permissions portlet or the XML configuration interface to move resources back and forth from internal to external access control.

By default, externalized roles appear in the external security manager as *Role Type@Resource Type/Name/Object ID*. For example, Administrator@PORTLET_APPLICATION/Welcome/1_1_1G.

You can change this format to *Resource Type/Name/Object ID@Role type*. This format change groups the roles by resource name instead of by role type. For example, PORTLET_APPLICATION/Welcome/1_0_1G@Administrator. This format change is visible only when the roles are externalized. This change does not affect the way roles are displayed in WebSphere Portal Express.

The `Administrator@VIRTUAL/wps.EXTERNAL ACCESS CONTROL/1` role is never affected by this format change. This role always appears with the role type Administrator.

Complete the following steps to manage access control with external security managers:

Procedure

1. Use the Resource Permissions portlet to internalize any external roles.
2. Log on to the WebSphere Integrated Solutions Console.

3. Modify the **WP AccessControlDataManagementService** Resource Environment Provider; change the **accessControlDataManagement.reorderRoleNames** parameter to `true`.

   Note: Add the **accessControlDataManagement.reorderRoleNames** parameter if it does not exist.
4. Save your changes and restart the WebSphere_Portal server.
5. Use the **Resource Permissions** portlet to externalize the resources you internalized in the first step.

Example

Example of roles list with **reorderRoleNames**=`false`:

```
Administrator@WEB_MODULE/Tracing.war/1_0_3K
Administrator@PORTLET_APPLICATION/Welcome/1_0_1G
User@WEB_MODULE/Tracing.war/1_0_3K
Privileged User@WEB_MODULE/Tracing.war/1_0_3K
Privileged User@PORTLET_APPLICATION/Welcome/1_0_1G
```

Example of roles list with **reorderRoleNames**=`true`:

```
PORTLET_APPLICATION/Welcome/1_0_1G@Administrator
PORTLET_APPLICATION/Welcome/1_0_1G@Privileged User
WEB_MODULE/Tracing.war/1_0_3K@Administrator
WEB_MODULE/Tracing.war/1_0_3K@Privileged User
WEB_MODULE/Tracing.war/1_0_3K@User
```

Related tasks:

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

Deleting passwords from properties files

The configuration tasks might require you to write security-sensitive information, such as passwords, into multiple properties files. When you no longer need this security-sensitive information for your configuration, you should remove them and move the files to a safe place or set the file permissions so that only authorized users can read them.

About this task

Complete the following steps to delete passwords and other security-sensitive information from the properties files:

Note: After completing the tasks to clean up the work directory and delete the passwords, you might find that in order to successfully perform additional configuration tasks, you need these passwords. Some tasks require you to add the passwords back to the `wkplc.properties`, `wkplc_comp.properties`, `wkplc_dbdomain.properties`, and `wkplc_sourceDb.properties` files while other tasks allow you to specify the passwords on the command line using the –D flag. Refer to the configuration task documentation to determine which method is required.

Procedure

1. Complete the following steps to access the WebSphere Portal Express configuration directory:

Table 241. Steps to access the configuration directory by operating system

Operating system	Steps
Windows	Complete the following steps to access the configuration directory: 1. Open a command prompt. 2. Change to the *wp_profile_root*\ `ConfigEngine` directory.
Linux	Complete the following steps to access the configuration directory: 1. Open a terminal session. 2. Change to the *wp_profile_root*/ `ConfigEngine` directory.
IBM i	Complete the following steps to access the configuration directory: 1. Type `STRQSH` on an OS/400 command line to start the Qshell Interpreter. 2. Change to the *wp_profile_root*/ `ConfigEngine` directory.

2. Remove the work directory that was created during the installation:
   - Windows: `ConfigEngine.bat cleanup-work-dir -DWasPassword=`*`password`*
   - Linux: `./ConfigEngine.sh cleanup-work-dir -DWasPassword=`*`password`*
   - IBM i: `ConfigEngine.sh cleanup-work-dir -DWasPassword=`*`password`*

   Note: Before running additional tasks, check the output for any error messages and, if instructed, correct any items before rerunning the task.
3. Enter the following commands from the configuration directory to remove all passwords from the `wkplc.properties`, `wkplc_comp.properties`, `wkplc_sourceDb.properties`, and `wkplc_dbdomain.properties` files:
   - Windows: `ConfigEngine.bat delete-passwords -DWasPassword=`*`password`*
   - Linux: `./ConfigEngine.sh delete-passwords -DWasPassword=`*`password`*
   - IBM i: `ConfigEngine.sh delete-passwords -DWasPassword=`*`password`*

   Note: Before running additional tasks, check the output for any error messages and, if instructed, correct any items before rerunning the task.

Updating user ID and passwords

IBM WebSphere Portal Express and IBM WebSphere Application Server use some accounts from the registry (for example, the LDAP server) including administrative and bind IDs for authenticated access to databases and LDAP severs respectively, as well as the WebSphere Portal Express and WebSphere Application Server administrative IDs. Often this means that the account passwords are stored in the WebSphere Portal Express and WebSphere Application Server bootstraps configuration files, which allows the authentication process to work.

About this task

Note: Before updating any user ID or password, review "User IDs and passwords" located under Planning for WebSphere Portal Express.

If the password for any ID is changed (either through WebSphere Portal Express or through any other means, including directly through the LDAP administration interfaces), then the password value stored in the appropriate configuration file must be changed at the same time. The following instructions describe how to make the appropriate changes based on which account passwords might have changed.

Remember: If you reuse the same account ID/password for multiple purposes, such as using `wpsbind` as the administrative ID and the LDAP access ID, then you might have to do more than one of the following steps to accommodate the password change. Some changes, particularly changes made through the WebSphere Integrated Solutions Console, require that the WebSphere Integrated Solutions Console is open and the current ID/password logged in before actually making the password change in the registry. Carefully plan which steps are required and in what order to avoid not being able to bring up server processes or log in.

Use the following topics to change passwords to better secure your environment.

"Changing the WebSphere Portal Express administrator password" on page 1672
IBM WebSphere Portal Express treats `wpsadmin` (the administrator) as any other user, just with more permissions granted. With a normal configuration, it is possible to change the `wpsadmin` or equivalent password through the user interface, just like any other user can manage their own password through the user interface. However, if the `wpsadmin` account is also used for more than just the administrator, then additional changes, outlined in other steps in this section, must be made to accommodate the change.

"Changing the WebSphere Application Server administrator password in the file registry" on page 1672
If you are using the file registry in the federation repository to store passwords, you need to change the passwords in the file registry.

"Changing the WebSphere Application Server administrator password in the LDAP server using the LDAP administration interface" on page 1673
If you are using the IBM Directory Server or IBM SecureWay Security Server for z/OS and OS/390 LDAP server, you can change the IBM WebSphere Application Server administrator password in the LDAP server using the LDAP administration interface. If you are using any other LDAP server, refer to the product documentation for information about changing passwords.

"Replacing the WebSphere Application Server administrator user ID" on page 1674
If you change your security configuration, you might need to replace your old IBM WebSphere Application Server administrator user ID with a new WebSphere Application Server administrator user ID.

"Replacing the WebSphere Portal Express administrator user ID" on page 1675
If you change your security configuration, you might need to replace your old IBM WebSphere Portal Express administrator user ID with a new WebSphere Portal Express administrator user ID.

"Changing the LDAP bind password" on page 1676
If you use an LDAP user registry, you must adapt the LDAP bind user ID.

"Changing database passwords that are used by WebSphere Portal Express" on page 1677
If database passwords are modified or expired, you must specify the new

passwords on the IBM WebSphere Application Server and on the IBM DB2 Universal Database Enterprise Server Edition server so that IBM WebSphere Portal Express can access them.

Changing the WebSphere Portal Express administrator password

IBM WebSphere Portal Express treats `wpsadmin` (the administrator) as any other user, just with more permissions granted. With a normal configuration, it is possible to change the `wpsadmin` or equivalent password through the user interface, just like any other user can manage their own password through the user interface. However, if the `wpsadmin` account is also used for more than just the administrator, then additional changes, outlined in other steps in this section, must be made to accommodate the change.

About this task

Perform the following steps to change the administrator password:

Note: You can also change the Administrator password, like any other user password, using an LDAP editor.

Procedure

1. Log in to WebSphere Portal Express as an administrator.
2. Click your user ID.
3. Complete the appropriate fields to change your password.
4. Click **OK**.
5. Complete the following steps to change the information stored in the **SearchAdminUser** alias:
   a. Log in to the WebSphere Integrated Solutions Console.
   b. Click **Security** > **Global security**.
   c. Under Authentication, click **Java Authentication and Authorization Service** > **J2C authentication data**.
   d. Edit the **SearchAdminUser** alias.
   e. Update the user ID and/or password to match your WebSphere Portal Express administrator information.

What to do next

Additionally, you should also change the password in the `wkplc.properties` file, located in the *wp_profile_root*`/ConfigEngine/properties` directory.

Changing the WebSphere Application Server administrator password in the file registry

If you are using the file registry in the federation repository to store passwords, you need to change the passwords in the file registry.

About this task

Complete the following steps to change the WebSphere Application Server administrator password stored in the file registry:

Procedure

1. Using a command prompt, change to the *wp_profile_root*/bin directory.
2. Issue the wsadmin -conntype NONE command and press **Enter**.
3. Issue the $AdminTask changeFileRegistryAccountPassword {-userId <wpsadmin_ID> -password <wpsadmin_new_password>} command and press **Enter**.
4. Issue the $AdminConfig save command and press **Enter**.
5. Complete the following steps to update the **RunAsRole**, which changes the stored password:

   You can change the password for the WebSphere Application Server administrator user ID using the WebSphere Portal Express Edit My Profile portlet, the native utilities for the user repository, such as the LDAP administration interface or the WebSphere Application Server Administrative utilities. Regardless of which option you choose, once you have updated the password, you must also update the **RunAsRole** for the PZNScheduler application.

   a. Log on to the WebSphere Integrated Solutions Console with your new password.
   b. Go to **Applications** > **Application Types** > **WebSphere enterprise applications**.
   c. Locate and click the **pznscheduler** application.
   d. Click **User RunAs Roles** under Detail Properties.
   e. Select **RuleEventRunAsRole** and then click **Remove**.
   f. Enter the fully distinguished name (DN) of the WebSphere Application Server Administrator in the **username** field and the new password in the **password** field.
   g. Select **RuleEventRunAsRole** and then click **Apply** to apply your changes.
   h. Click **OK**, save your changes, and then restart the server.

Changing the WebSphere Application Server administrator password in the LDAP server using the LDAP administration interface

If you are using the IBM Directory Server or IBM SecureWay Security Server for z/OS and OS/390 LDAP server, you can change the IBM WebSphere Application Server administrator password in the LDAP server using the LDAP administration interface. If you are using any other LDAP server, refer to the product documentation for information about changing passwords.

About this task

Perform the following steps to change the WebSphere Application Server administrator password in the LDAP server using the LDAP administration interface:

Attention: The following directions assume an LDAP tree layout where the users are all in the cn=users,o=wps subtree in the directory server. You should adjust these directions based on your own LDAP server layout.

Tip: When you change the WebSphere Application Server administrator password, you should also change it in LDAP server.

Procedure

1. Log in to the LDAP server Web Administration Tool.
2. Click **Directory management** > **Manage entries**.
3. Select the **o=wps RDN** and click **Expand**.
4. Select **cn=users** and click **Expand**.
5. Select the WebSphere Application Server administrator user and click **Edit Attributes**.

   Note: If this is your first time navigating to this screen, you may need to click **Next** before you can click the **Optional attributes** link.
6. Click **Optional attributes**.
7. Enter the new password in the **userPassword** field.
8. Click **OK**.
9. Exit the Web Administration Tool.
10. Complete the following steps to update the **RunAsRole**, which changes the stored password:

    You can change the password for the WebSphere Application Server administrator user ID using the WebSphere Portal Express Edit My Profile portlet, the native utilities for the user repository, such as the LDAP administration interface or the WebSphere Application Server Administrative utilities. Regardless of which option you choose, once you have updated the password, you must also update the **RunAsRole** for the PZNScheduler application.
    a. Log on to the WebSphere Integrated Solutions Console with your new password.
    b. Go to **Applications** > **Application Types** > **WebSphere enterprise applications**.
    c. Locate and click the **pznscheduler** application.
    d. Click **User RunAs Roles** under Detail Properties.
    e. Select **RuleEventRunAsRole** and then click **Remove**.
    f. Enter the fully distinguished name (DN) of the WebSphere Application Server Administrator in the **username** field and the new password in the **password** field.
    g. Select **RuleEventRunAsRole** and then click **Apply** to apply your changes.
    h. Click **OK**, save your changes, and then restart the server.

Replacing the WebSphere Application Server administrator user ID

If you change your security configuration, you might need to replace your old IBM WebSphere Application Server administrator user ID with a new WebSphere Application Server administrator user ID.

About this task

Complete the following steps to replace the WebSphere Application Server administrator user ID:

Procedure

1. Create a user in the **Manage Users and Groups** portlet to replace the current WebSphere Application Server administrative user.

2. Run the following task to replace the old WebSphere Application Server administrative user with the new user:
   - Linux : `./ConfigEngine.sh wp-change-was-admin-user -DWasPassword=`*`password`* `-DnewAdminId=`*`newadminid`* `-DnewAdminPw=`*`newpassword`* from the *wp_profile_root*`/ConfigEngine` directory.
   - IBM i: `ConfigEngine.sh wp-change-was-admin-user -DWasPassword=`*`password`* `-DnewAdminId=`*`newadminid`* `-DnewAdminPw=`*`newpassword`* from the *wp_profile_root*`\ConfigEngine` directory.
   - Windows: `ConfigEngine.bat wp-change-was-admin-user -DWasPassword=`*`password`* `-DnewAdminId=`*`newadminid`* `-DnewAdminPw=`*`newpassword`* from the *wp_profile_root*`\ConfigEngine` directory.

   Note: Refer to the description of the `newAdminId` property in `wkplc.properties` under `wp_profile_root\ConfigEngine\properties\` for guidance on populating the value in the command line.

   Additional parameter for stopped servers: This task verifies the user against a running server instance. If the server is stopped, add the **`-Dskip.ldap.validation`**`=true` parameter to the task to skip the validation.
3. Verify that the task completed successfully. Stop and restart all required servers.

What to do next

If you use an external security manager such as Security Access Manager, you must manually remove the old administrator user ID from the external security manager.

Replacing the WebSphere Portal Express administrator user ID

If you change your security configuration, you might need to replace your old IBM WebSphere Portal Express administrator user ID with a new WebSphere Portal Express administrator user ID.

About this task

Complete the following steps to replace the WebSphere Portal Express administrator user ID:

Important cluster note: If you are using IBM Web Content Manager within your clustered environment, you must complete these steps on every node in the cluster. If Web Content Manager is not configured, complete these steps only on the primary node.

Procedure

1. Create a user in the **Manage Users and Groups** portlet to replace the current WebSphere Portal Express administrative user.
2. Run the following task to replace the old WebSphere Portal Express administrative user with the new user:
   - **Linux :** `./ConfigEngine.sh wp-change-portal-admin-user -DWasPassword=`*`password`* `-DnewAdminId=`*`newadminid`* `-DnewAdminPw=`*`newpassword`* `-DnewAdminGroupId=`*`newadmingroupid`* from the *wp_profile_root*`/ConfigEngine` directory. The `-DnewAdminGroupId` parameter is required only if you plan to replace the old administrative group ID.

- **IBM i:** `ConfigEngine.sh wp-change-portal-admin-user -DWasPassword=`*`password`* `-DnewAdminId=`*`newadminid`* `-DnewAdminPw=`*`newpassword`* `-DnewAdminGroupId=`*`newadmingroupid`* from the *wp_profile_root*`/ConfigEngine` directory. The `-DnewAdminGroupId` parameter is required only if you plan to replace the old administrative group ID.
- **Windows:** `ConfigEngine.bat wp-change-portal-admin-user -DWasPassword=`*`password`* `-DnewAdminId=`*`newadminid`* `-DnewAdminPw=`*`newpassword`* `-DnewAdminGroupId=`*`newadmingroupid`* from the *wp_profile_root*`\ConfigEngine` directory. The `-DnewAdminGroupId` parameter is required only if you plan to replace the old administrative group ID.

Additional parameter for stopped servers: This task verifies the user against a running server instance. If the server is stopped, add the **`-Dskip.ldap.validation`**`=true` parameter to the task to skip the validation.

3. Verify that the task completed successfully. Stop and restart all required servers.
4. Complete the following steps to change the information stored in the **SearchAdminUser** alias:
   a. Log in to the WebSphere Integrated Solutions Console.
   b. Click **Security** > **Global security**.
   c. Under Authentication, click **Java Authentication and Authorization Service** > **J2C authentication data**.
   d. Edit the **SearchAdminUser** alias.
   e. Update the user ID and/or password to match your WebSphere Portal Express administrator information.
5. Clustered environments: Synchronize the nodes.
   a. Log on to the Deployment Manager.
   b. Go to **System Administration** > **Nodes**.
   c. Select the nodes to synchronize from the list.
   d. Click **Full Resynchronize**.

What to do next

Notes:

- If you use an external security manager such as Security Access Manager, you must manually remove the old administrator user ID from the external security manager.
- If you set the default portal administrator user ID to be used as the crawler user ID for Portal Search, you need to adapt that crawler user ID accordingly. For more information, see the topic about *Managing the content sources of a search collection*.

Related tasks:

"Managing the content sources of a search collection" on page 701
Search collections consist of one or more content sources. You can administer the content sources.

Changing the LDAP bind password

If you use an LDAP user registry, you must adapt the LDAP bind user ID.

About this task

Go to "Updating the federated LDAP user registry" on page 594 to view information about how to change the LDAP bind password:

Changing database passwords that are used by WebSphere Portal Express

If database passwords are modified or expired, you must specify the new passwords on the IBM WebSphere Application Server and on the IBM DB2 Universal Database Enterprise Server Edition server so that IBM WebSphere Portal Express can access them.

Before you begin

- Ensure that the administrative server for WebSphere Application Server is running.
- If this server is a clustered environment, you must have the Deployment Manager and the nodes up and the WebSphere Portal Express server stopped.
- If the Virtual Member Manager (VMM) is using the database, see *Changing the password for a repository under a federated repositories configuration* in the related links for instructions on updating the VMM database password.

Procedure

1. Check whether the database user was disabled because of invalid login attempts. Re-enable the database user if necessary.
2. Update the data sources for WebSphere Portal Express:
   a. Log in to the WebSphere Integrated Solutions Console.
   b. Click **Security** > **Global security** > **Java Authentication and Authorization Service** > **J2C Authentication Data**.
   c. Select the alias that you want to change; for example:
      wpsDBAuth
      jcrDBAuth
      fbkDBAuth
      lmDBAuth
      wcmDBAuth
   d. Update the password accordingly.
   e. Click **Apply**, and then click **Save** to save the configuration. You are informed that `security.xml` was changed.
3. If you are an administrator and must change the DB2 password, you must change the database administrator user password on the system.
   a. Stop the DB2 server.
   b. Use the `passwd` command to change the password.
   c. Restart the DB2 server. You can verify the new password by running the `db2 CONNECT TO WPSDB user db2admin using password` task.
4. Restart the WebSphere Integrated Solutions Console.
5. On the DB2 server, complete the following steps:
   a. Click **Administrative Tools** > **Services**.
   b. Stop all running DB2 services.
   c. For each service that your DB2 instance uses, display the menu and select **Properties**.

d. Select the **Log On** tab and change the DB2 administrator's user name and password.

If you do not change the DB2 administrator's user name and password in the properties of each DB2 service, the DB2 database application does not start.

Results

Repeat for each JDBC Provider, data source, and alias that is affected.

Results:

WebSphere Application Server and WebSphere Portal Express were updated to use your new database passwords. Verify that the WebSphere Portal Express application server is running by opening the following URL in a browser: `http://hostname.example.com:10039/wps/portal`, where *hostname.example.com* is the fully qualified host name of the server where Portal is running and *10039* is the default transport port that is created by WebSphere Application Server. The port number might be different for your environment..

Related information:

Changing the password for a repository under a federated repositories configuration

Chapter 13. Monitoring

WebSphere Portal includes tools and features to help you monitor the portal site.

"Portlet load monitoring for WebSphere Portal Express"
IBM WebSphere Portal Express now provides Portlet load monitoring. This can make your portal safer and more responsive.

"Analyzing portal usage data" on page 1687
You can collect data about the usage of your portal and analyze them.

"Auditing" on page 1731
IBM WebSphere Portal Express includes an auditing function that allows users to log certain events and their originators in to a separate log file. This file can then be used to track administrative activities.

Portlet load monitoring for WebSphere Portal Express

IBM WebSphere Portal Express now provides Portlet load monitoring. This can make your portal safer and more responsive.

Portlet load monitoring allows WebSphere Portal Express administrators to protect their portal by limiting the number of concurrent requests and the average response time allowed for JSR 168 or JSR 286 portlets. If a portlet exceeds either the defined maximum number of concurrent requests, or the average response time, then Portlet Load Monitoring no longer allows further requests by the portlet. Instead, the portal renders the portlet as unavailable, and the portlet code is no longer called for further requests. This way, the portal installation is protected from non-responsive portlets consuming an increasing number of threads.

These topics describe how you configure, administer, and use Portlet load monitoring and provides other useful information about it.

Note: Portlet load monitoring monitors JSR 168 and JSR 286 portlets only.

"Configuring and administering Portlet load monitoring" on page 1680
Here is an overview of the configuration and administration tasks and parameters for Portlet load monitoring.

"Portlet load monitoring properties" on page 1680
By setting the portal and portlet configuration parameters, you can monitor the session load and configure the parameters to increase performance.

"Administering Portlet load monitoring" on page 1682
You can administer Portlet load monitoring and the portlets that it monitors. You can determine whether and in which cases Portlet load monitoring blocks or reenables a portlet by setting its portlet preferences. You can also check which portlets Portlet load monitoring monitors, and you can manually block or activate these portlets. For example, you can reenable a portlet after Portlet load monitoring has blocked it.

"Logging and auditing events" on page 1684
Portlet load monitoring allows you to log events. For example, this can help you audit events. When Portlet load monitoring blocks or enables a portlet, it creates a log file entry in the IBM WebSphere Portal Express log file `SystemOut.log`. This log file entry contains the portlet object ID, the portlet name, the WAR file name of the portlet and the EAR file display name. The log

file entries consist of translated messages. If you use tools that monitor log files for events, you can check for log file entries that are related to Portlet load monitoring as described here.

"API for accessing Portlet load monitoring data" on page 1686
Portlet load monitoring provides an API for accessing the monitoring data. You can use this API to write custom code to access that data.

Configuring and administering Portlet load monitoring

Here is an overview of the configuration and administration tasks and parameters for Portlet load monitoring.

Portal wide configuration

You can do the following configuration tasks for Portlet load monitoring:

- Enabling or disabling Portlet load monitoring in your portal.
- Setting the sample number of requests by which the average response time for portlets is calculated. For example, if you set the sample number to 50, the average response time is calculated from the latest 50 requests that this portlet served. This setting is related to the average response time setting listed under administrative tasks.

Perform both of these tasks in the WebSphere Integrated Solutions Console. These settings apply to all portlets that Portlet load monitoring monitors.

Portlet preferences configuration and administration

You can set the following parameters for individual portlets in the portlet preferences. You can either configure these parameters in the `portlet.xml` file before you deploy the portlet or administer them later by using the **Manage Portlets** administration portlet:

- Maximum number of concurrent requests that are allowed for a portlet
- Reactivation limit of concurrent requests that are allowed for reactivating the portlet
- Allowed average response time. You can configure the sample number of requests by which this average response time for portlets is calculated as mentioned previously under configuration tasks.

Portlet load monitoring administration

You can also do the following administrative tasks on individual portlets by using the **Manage Portlets** portlet only:

- Check whether a portlet is being monitored by Portlet load monitoring
- Manually reenable a portlet that Portlet load monitoring has blocked
- Manually block a portlet that is being monitored by Portlet load monitoring from responding to requests.

Portlet load monitoring properties

By setting the portal and portlet configuration parameters, you can monitor the session load and configure the parameters to increase performance.

Portal configuration

After you set the service configuration properties, add the following custom properties for resource environment provider WP ConfigService. These parameters affect all parameters in the portal:

com.ibm.wps.plm.enabled = false

Use this parameter to enable and disable Portlet load monitoring in the

portal. By default Portlet load monitoring is disabled. To enable it, set the entry **com.ibm.wps.plm.enabled** to **true** in the WebSphere Integrated Solutions Console. If you want to disable Portlet load monitoring, set **com.ibm.wps.plm.enabled** to **false**.

com.ibm.wps.plm.statistics.requestnumber = 50
Use this parameter to define the number of samples for the calculation of the average response time for a portlet. The default value is 50. The average response time for the portlet is calculated from the 50 latest requests that this portlet served. You can change the number of samples used for the average response time calculation. Add an entry with the name **com.ibm.wps.plm.statistics.requestnumber** to the WebSphere Integrated Solutions Console. Set its value to the number of requests that you want to be used for average response time calculation. For example: If you want to consider the latest 75 requests for the average response time calculation, set the property **com.ibm.wps.plm.statistics.requestnumber** to a value of **75**.

Portlet configuration

Portlet load monitoring can monitor every JSR 168 or JSR 286 portlet installed in your WebSphere Portal Express. By default, Portlet load monitoring does not monitor portlets. If you want Portlet load monitoring to monitor a portlet, you need to set specific portlet preferences for a portlet definition.

You can configure these portlet preferences for a portlet either by setting them in the portlet.xml file before you deploy the portlet. Or you can administer them after the portlet was deployed by using the **Manage Portlets** administration portlet.

com.ibm.wps.pe.plm.maxrequest
Use this parameter to define the maximum number of concurrent requests that is allowed for a portlet. If the number of requests that the portlet serves at any time exceeds the maximum number of concurrent requests that you specify here, then Portlet load monitoring blocks further requests to this portlet. Instead of responding to the requests the portlet renders with a message that states the portlet is not available. To re-enable this portlet for rendering, a portal administrator can enable the portlet by using the Manage Portlets administration portlet.

Example: If you want to allow no more than a maximum of 10 concurrent requests for a portlet, set the portlet preference **com.ibm.wps.pe.plm.maxrequest** to a value of **10**.

com.ibm.wps.pe.plm.minrequest
Use this parameter to define the reactivation limit for a portlet. Use this parameter for a recovery process. If Portlet load monitoring blocked the portlet because it exceeded the maximum number of allowed concurrent requests, then no more requests to this portlet are allowed. If the portlet then completes its active requests after some time, the number of concurrent requests that are currently served by the portlet decreases. If the number of concurrent requests in the portlet falls back down to the reactivation limit that you specified for the portlet preference parameter **com.ibm.wps.pe.plm.minrequest**, Portlet load monitoring enables the portlet for responding to requests and rendering again. This setting is subject to two restrictions:

1. This setting is only evaluated when you set the maximum number of concurrent requests by using the parameter **com.ibm.wps.pe.plm.maxrequest**.
2. This value must be less than the maximum number of concurrent requests that you entered for the parameter **com.ibm.wps.pe.plm.maxrequest**.

Example 1: If you want the portlet to respond to requests again when it has no more than three active requests open, set the portlet preference parameter **com.ibm.wps.pe.plm.minrequest** to 3.

Example 2: If you set the maximum number of concurrent requests for a portlet to 10 by using the parameter **com.ibm.wps.pe.plm.maxrequest** and the reactivation limit of concurrent requests for the same portlet to three using the parameter **com.ibm.wps.pe.plm.minrequest**, Portlet load monitoring works as follows:

1. When the portlet exceeds 10 and reaches 11 concurrent requests, Portlet load monitoring blocks the portlet from further requests.
2. When the portlet completes 8 active requests and has three active requests to complete, Portlet load monitoring allows the portlet to respond to incoming requests again.

com.ibm.wps.pe.plm.average.time.processing

Use this parameter to define the allowed average response time for the portlet. Specify a value in milliseconds. If the portlet exceeds the average response time that you specified, then Portlet load monitoring blocks further requests to this portlet. To reenable this portlet for rendering, a portal administrator can enable the portlet by using the Manage Portlets administration portlet.

For example: If you want to specify 3 seconds as the average response time allowed for a portlet, set the portlet preference `com.ibm.wps.pe.plm.average.time.processing` to `3000` (milliseconds) for this portlet.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Administering Portlet load monitoring

You can administer Portlet load monitoring and the portlets that it monitors. You can determine whether and in which cases Portlet load monitoring blocks or reenables a portlet by setting its portlet preferences. You can also check which portlets Portlet load monitoring monitors, and you can manually block or activate these portlets. For example, you can reenable a portlet after Portlet load monitoring has blocked it.

"Administering portlet preferences for Portlet load monitoring" on page 1683
You can set portlet preferences to influence in which cases Portlet load monitoring blocks or reenables JSR portlets.

"Administering portlets for Portlet load monitoring" on page 1684
You can administer the portlets that Portlet load monitoring monitors. You can check which portlets Portlet load monitoring monitors, and you can manually block or activate these portlets. For example, you can reenable a portlet after Portlet load monitoring blocked it.

Administering portlet preferences for Portlet load monitoring

You can set portlet preferences to influence in which cases Portlet load monitoring blocks or reenables JSR portlets.

About this task

You can configure parameters for the following:

- Maximum number of concurrent requests
- Reactivation limit of concurrent requests for reactivating the portlet
- Allowed average response time. You can configure the sample number of requests by which this average response time for portlets is calculated as mentioned in configuration tasks.

For more information about these three parameters and their meaning, refer to the topic about Portlet configuration parameters for Portlet load monitoring. For information about setting the sample by which the average response time is calculated refer to the topic about Portal configuration parameters for Portlet load monitoring.

Procedure

1. Log in to the portal by using a portal administrator user ID.
2. Click the **Administration menu** icon. Then, click **Portlet Management** > **Portlets**. The Manage Portlets portlet is displayed. It lists the portlets in your portal.
3. Search for the portlet for which you want to set portlet preferences.
4. Select the **Configure Portlet** icon.
5. Add a new portlet preference as required and type **number** for the new portlet preference as required:
   - To set the maximum number of requests that you want to allow for this portlet, add `com.ibm.wps.pe.plm.`*`max`*`request = `*`xyz`* for the new portlet preference. For example, if you want to allow this portlet no more than 15 concurring requests, specify `com.ibm.wps.pe.plm.maxrequest = 15`.
   - To set the reactivation limit for this portlet, add `com.ibm.wps.pe.plm.`*`min`*`request = `*`xyz`* for the new portlet preference. For example, if you want this portlet to be reenabled when its concurrent requests fall back to 5 or less, specify `com.ibm.wps.pe.plm.minrequest = 5`.
   - To set the allowed average response time for this portlet, add `com.ibm.wps.pe.plm.average.time.processing = `*`xyz`*, where `xyz` is the average response time in milliseconds. For example, if you specify `com.ibm.wps.pe.plm.average.time.processing = 3000` and the average response time for this portlet exceeds 3 seconds, then Portlet load monitoring blocks further requests to this portlet.
6. Click the **Add** button to add the preference.
7. Save your changes by clicking the **OK** button.

What to do next

As an alternative, you can define a portlet preference before you deploy the portlet. In this case add the portlet preference to the `portlet.xml` deployment descriptor of your WAR file.

Administering portlets for Portlet load monitoring

You can administer the portlets that Portlet load monitoring monitors. You can check which portlets Portlet load monitoring monitors, and you can manually block or activate these portlets. For example, you can reenable a portlet after Portlet load monitoring blocked it.

Procedure

1. Log in to the portal by using a portal administrator user ID.
2. Click the **Administration menu** icon. Then, click **Portlet Management** > **Portlets**. The Manage Portlets portlet is displayed. It lists the portlets in your portal.
3. Search for the portlet that you want to administer.
4. Administer the portlet as required:
   - For each portlet that is monitored by Portlet load monitoring, the Manage Portlets portlet displays an icon for disabling or enabling that portlet.
   - If a portlet that is monitored by Portlet load monitoring is active, the list shows an icon for disabling that portlet. If you click this icon, the portlet is blocked by Portlet load monitoring and is no longer rendered in the portal.
   - If a portlet that is monitored by Portlet load monitoring is blocked and inactive, the list shows an icon for enabling that portlet. If you click this icon, the portlet is enabled for rendering again.

   Notes:

   a. If the portlet was blocked for exceeding the maximum number of concurrent requests and the number of requests did not decrease and you did not set the value for maximum number of requests to a higher value, clicking this icon has no effect. In this case, the portlet is blocked again with the next request that arrives for this portlet.

   b. Activating a blocked portlet clears the response times that are stored for this portlet.

Logging and auditing events

Portlet load monitoring allows you to log events. For example, this can help you audit events. When Portlet load monitoring blocks or enables a portlet, it creates a log file entry in the IBM WebSphere Portal Express log file `SystemOut.log`. This log file entry contains the portlet object ID, the portlet name, the WAR file name of the portlet and the EAR file display name. The log file entries consist of translated messages. If you use tools that monitor log files for events, you can check for log file entries that are related to Portlet load monitoring as described here.

About this task

Portlet load monitoring creates log file entries for the events that are described in the following list.

Portlet load monitoring blocks a portlet because the portlet exceeds the maximum number of requests.

If Portlet load monitoring blocks a portlet because it exceeded the maximum number of concurrent requests that are specified for it, Portlet load monitoring creates the following log file entry with the message code `EJPPG3001W`:

```
EJPPG3001W: Portlet load monitoring disabled the portlet with
object ID: Object_ID, portlet name: Portlet_Name,
WAR file name: WAR_File_Name, EAR file display name: EAR_File_Display_Name,
because the portlet exceeded its maximum number of requests.
```

Portlet load monitoring blocks a portlet because the portlet exceeds the average response time.

If Portlet load monitoring blocks a portlet because that portlet exceeded the average response time that is specified for this portlet, Portlet load monitoring creates the following log file entry with the message code EJPPG3002W:

```
EJPPG3002W: Portlet load monitoring disabled the portlet with
object ID: Object_ID, portlet name: Portlet_Name,
WAR file name: WAR_File_Name, EAR file display name: EAR_File_Display_Name,
because the portlet exceeded its average response time.
```

Portlet load monitoring activates a portlet because the portlet returns to the reactivation limit.

If Portlet load monitoring reenables a blocked portlet because the blocked portlet falls back down to the reactivation limit of concurrent requests that are defined for the portlet, Portlet load monitoring creates the following log file entry with the message code EJPPG3003I:

```
EJPPG3003I: Portlet load monitoring reenabled the portlet
with object ID: Object_ID, portlet name: Portlet_Name,
WAR file name: WAR_File_Name, EAR file display name: EAR_File_Display_Name
because number of portlet requests fell below the reactivation limit.
```

Administrator manually blocks requests to a portlet.

If a portal administrator manually blocks requests to a portlet, Portlet load monitoring creates the following log file entry with the message code EJPPD0101I:

```
EJPPD0101I: Portal administrator admin_user_ID manually blocked requests
to the portlet with object ID: Object_ID, portlet name: Portlet_Name,
WAR file name: WAR_File_Name, EAR file display name: EAR_File_Display_Name
```

Administrator manually unblocks requests to a portlet.

If an administrator manually unblocks requests to a blocked portlet, Portlet load monitoring creates the following log file entry with the message code EJPPD0100I:

```
EJPPD0100I: Portal administrator admin_user_ID manually unblocked requests
to the portlet with object ID: Object_ID, portlet name: Portlet_Name,
WAR file name: WAR_File_Name, EAR file display name: EAR_File_Display_Name.
```

In the log entry, the variables *Object_ID*, *Portlet_Name*, *War_File_Name*, *EAR_File_Dislay_Name*, *admin_user_ID* are substituted with the corresponding values for the affected portlet and the administrative user.

Example: If Portlet load monitoring blocks a portlet with the portlet name **StdWorldClock** because the portlet exceeded the maximum number of requests that are allowed, then the log file entry might look like this:

```
EJPPG3001W: Portlet load monitoring disabled the portlet with
   ObjectID: [ObjectIDImpl '3_MLSU3F540O0360ISG212TT2003',
             PORTLET_DEFINITION, VP: 0,
             [Domain: rel], DB: 0000-B6723F5E2100836180E45004D1BB0060],
   portlet name: StdWorldClock,
   WAR file name: StdWorldClock.war,
   EAR file display name: PA_StandardWorldClock
because portlet exceeded its maximum number of requests.
```

API for accessing Portlet load monitoring data

Portlet load monitoring provides an API for accessing the monitoring data. You can use this API to write custom code to access that data.

Portlet load metrics interface

The PortletLoadMetrics object is defined in the package com.ibm.portal.plm.PortletLoadMetrics. It contains metrics data for one specific PortletDefinition. The interface looks as follows:

```
package com.ibm.portal.plm;

/*
 *  This interface provides access to portlet metrics stored for Portlet Load Monitoring.
 */
public interface PortletLoadMetrics {

 /*
  * Returns true if the portlet is enabled for rendering (either by admin or by PLM).
  * If the portlet is disabled for rendering, this method returns false.
  *
  * @return boolean true if the portlet is enabled either by admin or by PLM.
  */
 public abstract boolean isPortletEnabled();

 /*
  * Checks if the Portlet is currently enabled for rendering due to PLM constraints.
  * If PLM disabled the portlet for rendering, this method returns false.
  *
  * @return boolean true if the Portlet is enabled for rendering,
                    false if the Portlet is disabled for rendering.
  */
 public abstract boolean isPortletEnabledByPLM();

 /*
  * Returns true if the portlet is manually enabled by the portal administrator.
  * If the portal administrator disabled the portlet for rendering, this method returns false.
  *
  * @return boolean true if the portlet is enabled for rendering by the portal administrator.
  */
 public abstract boolean isPortletEnabledByAdmin();

 /*
  * Returns the number of requests this portlet currently serves.
  *
  * @return int number of requests this portlet currently serves.
  */
 public abstract int getCurrentNumberOfRequests();

 /*
  * Calculates the average response time for requests of this portlet in milliseconds.
  *
  * @return int average response time of the portlet in milliseconds.
  */
 public abstract int getAverageResponseTime();
```

Portlet load metrics service

The isPortletLoadMetricsService is contained in the package com.ibm.portal.plm.service.PortletLoadMetricsService. You can use this service to access PortletLoadMetrics objects. It also allows you to query general Portlet load monitoring related information. In order to obtain the PortletLoadMetricsService, you have to perform a JNDI lookup in the following way:

```
     PortletLoadMetricsEnabled(ObjectID portletID) plm;
      javax.naming.Context ctx = new javax.naming.InitialContext();
      try {
           plm = (PortletLoadMetricsService
              ctx.lookup(PortletLoadMetricsService.JNDI_NAME);
      } catch(javax.naming.NameNotFoundException ex) {
           ... error handling ...
      }
```

Once you successfully obtained the PortletLoadMetricsService, you can use methods as defined in the PortletLoadMetrics interface:

```
public interface PortletLoadMetricsService {

 String JNDI_NAME = "portal:service/plm/PortletLoadMetrics";

 /**
  * Determines if PLM is enabled for a given portlet.
  *
  * @param portletID ObjectID of the portlet.
  * @return true If PLM is enable for a given portlet.
  *       false If PLM is not enabled for a given portlet or if portlet does not exist.
  */
 boolean isPortletLoadMetricsEnabled(ObjectID portletID);

 /**
  * Returns a PortletLoadMetrics object for the portlet.
  * Creates the PortletLoadMetrics if PLM is enabled for this portlet and
  * it the PortletLoadMetrics object does not yet exist.
  *
  * @param portletID ObjectID of the portlet.
  * @return Reference to a PortletLoadMetrics object.
  *       Null if PLM is not enabled for the specified portlet.
  */
 PortletLoadMetrics getPortletLoadMetrics(ObjectID portletID);

 /**
  * Returns a String object containing the portlet preference value set for maximum concurrent
  * requests allowed for this portlet.
  *
  * @param portletID ObjectID of the portlet.
  * @return String containing the portlet preference value set for maximum concurrent
  *          requests allowed for this portlet. Returns "null" if the portlet did not set the
  *          portlet preference for maximum concurrent requests.
  */
 String getMaximumRequestPreferenceValue(ObjectID portletID);

 /**
  * Returns a String object containing the portlet preference value set for the
  * reactivation limit for this portlet (PLM self-healing)
  *
  * @param portletID ObjectID of the portlet.
  * @return String containing the portlet preference value set for the reactivation
  *          limit for this portlet (PLM self-healing). Returns "null" if the portlet
  *          did not set the portlet preference for the reactivation limit.
  */
 String getMinimumRequestPreferenceValue(ObjectID portletID);

 /**
  * Returns a String object containing the portlet preference value set for allowed average
  * response time for this portlet
  *
  * @param portletID ObjectID of the portlet.
  * @return String containing the portlet preference value set for allowed average
  *          response time for this portlet. Returns "null" if the portlet
  *          did not set the portlet preference for allowed average response time.
  */
 String getAverageResponseTimePreferenceValue(ObjectID portletID);

 /**
  * Returns a Map containing all PortletLoadMetrics objects.
  *
  * @return Map<ObjectID, PortletLoadMetrics> containig all PortletLoadMetrics objects.
  *          Key of the Map is the ObjectID of the PortletDefinition.
  *          The value of the Map is the PortletLoadMetrics object of the PortletDefinition.
  */
 Map<ObjectID, PortletLoadMetrics> getAllPortletLoadMetricsObjects();

}
```

Analyzing portal usage data

You can collect data about the usage of your portal and analyze them.

About this task

You can collect these types of data:

- Server side data of your portal site. This consists mainly of technical data internal to the portal.
- Server side data of your IBM Web Content Manager items.
- Data about the behavior of your client users. You analyze this data by using Active Site Analytics (ASA).

  "Logging and analyzing server side site data"
  IBM WebSphere Portal Express implements a logging function for your usage data. The portal writes usage records to a dedicated log file if site analysis logging is enabled. Multiple types of site analyzer loggers allow portal administrators to collect statistical data in various areas. The portal server manages the collection of data on its own, but from a business point of view you can also log custom details of business events. You can configure the portal for site analysis logging for the web content viewer.

  "Analyzing user behavior by Active Site Analytics" on page 1696
  You can collect data about user behavior in your portal and send that data to a service for analysis. For this purpose the portal provides Active Site Analytics (ASA).

Related tasks:

"Setting up site analysis for the Web Content Viewer" on page 411
To track usage data for the Web Content Viewer, you can configure the portal for site analysis logging for the Web Content Viewer.

Logging and analyzing server side site data

IBM WebSphere Portal Express implements a logging function for your usage data. The portal writes usage records to a dedicated log file if site analysis logging is enabled. Multiple types of site analyzer loggers allow portal administrators to collect statistical data in various areas. The portal server manages the collection of data on its own, but from a business point of view you can also log custom details of business events. You can configure the portal for site analysis logging for the web content viewer.

The portal configuration service SiteAnalyzerLogService determines the type of site analysis data that the portal logs at run time. Depending on the service configuration, the portal logs the following events:

- Page management, such as creating, reading, updating, deleting pages.
- Requests of pages by users.
- Requests of portlets by users.
- Session activities, such as login, logout, time out, login failed.
- User management actions, such as creating, reading, updating, and deleting users and groups.

The resulting log entries comply with the NCSA Combined industry standard. By analyzing the log entries, you can monitor applications that are running on your portal site. The site analysis infrastructure that is provided by the portal accommodates most scenarios.

Note: To obtain a more sophisticated evaluation of the portal usage, or to generate reports for portlet actions, you must write a custom report to log custom business events that occur in portlets.

"Enabling site analysis logging"
Site analysis logging is not enabled by default. To enable site analysis logging, specify the names and locations of the log files as values for the following parameters in the WP SiteAnalyzerLogService.

"Analysis loggers reference" on page 1690
The following table lists and describes available loggers.

"Understanding the site analysis log" on page 1691
Learn more details about how to read the site analysis log.

"Logging custom details of business events for site analysis" on page 1692
From a business point of view, you might want to log custom details of business events.

Enabling site analysis logging

Site analysis logging is not enabled by default. To enable site analysis logging, specify the names and locations of the log files as values for the following parameters in the WP SiteAnalyzerLogService.

About this task

For details, see *Setting service configuration properties*.

Procedure

1. Specify the names for the log files and backup log files.

   SiteAnalyzerFileHandler.fileName
   Specify the location and file name of the log file. The default value is logs/*$APPSERVER_NAME*/sa.log.

   SiteAnalyzerFileHandler.backupFileName
   Specify the location and file name of the backup file for the log file. The default value is logs/*$APPSERVER_NAME*/sa_*$CREATE_TIME*.log. When the log file is backed up, the current data is stored in the backup file, sa_*$CREATE_TIME*.log, and a new log file, sa.log, is created.

   You can specify the following tokens as part of the directory location or file name:

   $APPSERVER_NAME
   Name of the application server. Use this token for vertical clusters to enforce that the different application servers write into different files if they share file system.

   $CREATE_TIME
   Date and time the file is created. Specify the format of this token in this parameter: **SiteAnalyzerFileHandler.dateFormat**.

   $CLOSE_TIME
   Date and time the file is closed. Specify the format of this token in this parameter: **SiteAnalyzerFileHandler.dateFormat**.

   For example, you can specify log file locations and names as follows: log/backup/$APPSERVER_NAME/sa_$CREATE_TIME_$CLOSE_TIME.log.

   Note: If WebSphere Portal Express is writing to a file, the values for $CLOSE_TIME and $CREATE_TIME are the same.

2. Specify the date format for tokens in the log file names.

   SiteAnalyzerFileHandler.dateFormat
   Specify a value to format the date and time for the $CLOSE_TIME and

$CREATE_TIME tokens. For example, SiteAnalyzerFileHandler.dateFormat=yyyy.MM.dd-HH.mm.ss.

3. Specify the interval to back up log files.

   SiteAnalyzerFileHandler.minutesPerLogFile
   Sets the backup interval in minutes. Specify a value between 1 and 60.

   SiteAnalyzerFileHandler.hoursPerLogFile
   Sets the backup interval in hours. Specify a value between 1 and 24.

   SiteAnalyzerFileHandler.daysPerLogFile
   Sets the backup interval in days. Specify any value that indicates the number of days between backups.

   Notes:

   - If you enable more than one date format interval, the smallest interval is used.
   - If you specify 60 minutes, the file is backed up after 60 minutes. If you specify 1 hour, the file is backed up on the next full hour; for example, 01:00, 02:00 and so on. If you specify an interval of days, the file is backed up at 24:00 (midnight). For more information about the date format, see the Javadoc documentation for java.text.SimpleDateFormat.

4. Activate loggers as appropriate: To activate a logger, select the logger that you plan to activate and set the value to true. For example, SiteAnalyzerSessionLogger.isLogging=true. For a list of available loggers, refer to *Analysis loggers reference*.

   Tip: Site analysis logging can affect performance. For this reason, you might choose to disable loggers when necessary. To disable loggers, set the value to false.

5. Restart WebSphere Portal Express.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Analysis loggers reference

The following table lists and describes available loggers.

Table 242. Description and activity logged for each available logger

WebSphere Portal Express Server Analysis Logger	Description and Activity Logged
SiteAnalyzerSessionLogger.isLogging	HTTP requests that include the URLs: /Command/Login /Command/Logout
SiteAnalyzerUserManagementLogger.isLogging	HTTP requests that include the URLs: /Command/UserManagement/CreateUser /Command/UserManagement/DeleteUser /Command/UserManagement/CreateGroup /Command/UserManagement/DeleteGroup

Table 242. Description and activity logged for each available logger (continued)

WebSphere Portal Express Server Analysis Logger	Description and Activity Logged
SiteAnalyzerPageLogger.isLogging	HTTP requests that include the URLs: `/Page/*` `/Command/Customizer/CreatePage` `/Command/Customizer/EditPage` `/Command/Customizer/DeletePage`
SiteAnalyzerPortletLogger.isLogging	HTTP requests that include the URLs: `/Portlet/*`
SiteAnalyzerPortletActionLogger.isLogging	HTTP requests that include the URLs: `/PortletAction/*`
SiteAnalyzerErrorLogger.isLogging	HTTP requests that include the URLs: `/Error/Portlet` `/Error/Page` The referrer field points to a portlet name or a page name. Examples of such referrer fields are: `http://your.server.name/Portlet/1/PortletName` `http://your.server.name/Page/1/PageName`
SiteAnalyzerApplicationActionLogger.isLogging	HTTP requests that include the URLs: `/ApplicationAction/*`
SiteAnalyzerJSRPortletLogger.isLogging	web content viewer

Understanding the site analysis log

Learn more details about how to read the site analysis log.

The IBM WebSphere Portal Express site analysis log information is collected in the following file:

wp_profile_root`/logs/app_server_name/sa_`*date_time*`.log`

where *wp_profile_root* is the WebSphere Portal Express root directory and *date_time* is the date and time when the file was created. The current (active) log file is named **sa.log**. As explained in the previous section, the dateFormat parameter determines the interval at which the file is created.

The site analysis log uses the NCSA Combined log format, which is a combination of NCSA Common log format and three additional fields: the referrer field, the user_agent field, and the cookie field. The following example displays a sample entry in the site analysis log. The table describes each field of the log format:

```
9.37.3.88 - customer2 [10/Apr/2002:21:33:16 +0000]
"GET /Portlet/146/Welcome_Portlet?PortletPID=146&PortletMode=View&PortletState=Normal
HTTP/1.1" 200 -1 "http://myserver.company.com/Page/110/Welcome"

"JSESSIONID=OXDFAPVR4SXYZOIHSLVGA2Y"
```

Table 243. Log fields, names, and explanations

Field in the Example	Log Field Name and Explanation
9.37.3.88	**host** The IP address of the HTTP client that sent the request. **Important:** If there is a reverse proxy server between the client and the portal, the IP address logged is that of the reverse proxy server rather than the HTTP client. To log the IP address of the HTTP client, you must remove the reverse proxy server from the environment.
-	**rcf931** The identifier used to identify the client making the request. This field is always set to the hyphen character (-).
customer2	**username** The user ID for the client. If the user ID is not known, the field is set to the hyphen character (-).
[10/APR/2002:21:33:16 +0000]	**date:time: timezone** The date and time of the HTTP request.
"GET /Portlet/146/Welcome_Portlet? PortletMode= View&PortletState= Normal HTTP/1.1"	**request** The HTTP method, the URL of the requested resource and the version of HTTP used by the client.
200	**statuscode** The HTTP status code for the request.
-1	**bytes** The number of bytes of data transferred from the client as part of the request. A value of -1 means unknown.
"http://myserver.company.com/Page/110/ Welcome"	**referrer** The URL that linked the client to the website. For some requests, the referrer might not be logged. In such cases, the field is set to empty double quotes: ""
"Firefox/2.0 (compatible; MSIE 5.5; Windows NT 4.0)"	**user_agent** The type of Web browser used by the client.
"JSESSIONID=OXDFAPVR4SXYZOIHSLVGA2Y"	**cookies** The name and value of a cookie that was sent to the client browser. If multiple cookies were sent, the list is delimited by the semicolon character.

Logging custom details of business events for site analysis

From a business point of view, you might want to log custom details of business events.

About this task

Note: The procedures that are described in the following topic and subtopics work only for standard (JSR 168 or JSR 286) portlets.

The following are main components of the site analysis logging framework for standard portlets:

com.ibm.portal.portlet.service.siteanalyzer.PortletSiteAnalyzerLoggingServiceHome
This portlet service exposes methods to obtain a logger instance that is specialized on the specified request type. This request type is either an

ActionRequest, an EventRequest, a RenderRequest, or a ResourceRequest. Portlets obtain this portlet service through a JNDI lookup.

com.ibm.portal.portlet.service.siteanalyzer.PortletSiteAnalyzerLogger
Portlets can retrieve an instance of this logger from the PortletSiteAnalyzerLoggingServiceHome. It is valid for the request for which it was created, and it provides methods to query the logger state - enabled or disabled - and to create a site analysis log entry.

com.ibm.portal.portlet.service.siteanalyzer.ParameterNamesProcessor
Request parameter names, in particular render parameter names that are encoded into portal URLs, should be as short in length as possible. This recommendation does not leave much space for meaningful parameter names in site analysis log entries. The ParameterNamesProcessor interface allows portlet developers to provide the PortletSiteAnalyzerLogger with a callback for processing request parameter names. Implementations of this interface are called by the site analytics framework before assembling the query string section of the request URI that is written to the site analysis log file.

To activate and use the portal site analyzer logger for standard portlets, proceed by the following steps:

Procedure

1. Activating the site analyzer logger for standard portlets: You enable the site analyzer logger for standard portlets by using the portal configuration service SiteAnalyzerLogService. You set its property in the WebSphere Integrated Solutions Console through the resource environment provider WP SiteAnalyzerLogService. To do this step, proceed as follows:
   a. Select the appropriate WebSphere Integrated Solutions Console, depending on your environment:
      - If your portal runs stand-alone, use the local WebSphere Integrated Solutions Console.
      - If your portal is installed in a cluster, use the WebSphere Integrated Solutions Console of the Deployment Manager.
   b. Start the WebSphere Integrated Solutions Console by entering the following URL in the **URL location** field of a web browser:
      `http://your_server.com:admin_port/ibm/console`

      where *your_server.com* is the name of your server and *admin_port* is the port that is assigned to the WebSphere Integrated Solutions Console.
   c. In the navigation click **Resources** > **Resource Environment** > **Resource Environment Providers**.
   d. In the **Resource environment providers** page, elect the appropriate node or cluster from the scopes pull-down list, or clear the **Show Scope** selection drop-down check box and select one of the following options, depending on your portal environment:
      - If your portal runs as a single server, select **Browse Nodes** and select the node.
      - If your portal is installed in a cluster, select **Browse Clusters** and select the portal cluster.
   e. Select **WP SiteAnalyzerLogService**.
   f. Click **Custom Properties**.
   g. Do one of the following:

- Select the property **SiteAnalyzerJSRPortletLogger.isLogging** and change its value to `true`.
- Create a new property named `SiteAnalyzerJSRPortletLogger.isLogging` and set its value to `true`. Use `java.lang.String` as the property type.

h. When you are done, click **Save** at the start of the page under **Message(s)**.

i. Click **Save** again when prompted to confirm your change.

j. Optional: If your portal runs in a cluster configuration, replicate your changes to the cluster.

k. Restart the portal to activate the changes.

2. Implementing a parameter names processor: The following code sample shows an implementation of the `ParameterNamesProcessor` interface. The sample holds a static mapping of possible request parameter names to a more descriptive expression that you want to be used in site analysis log entries instead. If there is no replacement for a parameter name, the original key is returned to be written to the site analysis log entry as it is. Sample:

```
public class ParameterNamesProcessorSample implements ParameterNamesProcessor
{
   private static final Hashtable<String, String> PARAM_NAMES_MAP;

   static
   {
      PARAM_NAMES_MAP = new Hashtable();
      PARAM_NAMES_MAP.put("ci", "CurrentItem");
   }

   public String processParameterName(String paramName)
   {
      // get a replacement for the given parameter name or
      // return the original key if not replacement exists
      if (PARAM_NAMES_MAP.containsKey(paramName))
      {
         return PARAM_NAMES_MAP.get(paramName);
      }
      else
      {
         return paramName;
      }
   }
}
```

3. Retrieving the portlet site analyzer logging service: The following code sample shows the `init` method of a portlet class that looks up an instance of the portlet site analyzer logging service. Additionally, it instantiates a sample implementation of the `ParameterNamesProcessor` interface.

```
private PortletSiteAnalyzerLoggingServiceHome iSALogServiceHome = null;

private ParameterNamesProcessorSample iParamNamesProc = null;

public void init() throws PortletException
{
  com.ibm.portal.portlet.service.PortletServiceHome psh;
  javax.naming.Context ctx = new javax.naming.InitialContext();

  try
  {
    psh = (PortletServiceHome) ctx
      .lookup(PortletSiteAnalyzerLoggingServiceHome.JNDI_NAME);
  }
  catch (javax.naming.NameNotFoundException e) {
    // error handling
  }
```

```
  // obtain the service object
  PortletSiteAnalyzerLoggingServiceHome iSALogServiceHome =
    (PortletSiteAnalyzerLoggingServiceHome) psh
      .getPortletService(PortletSiteAnalyzerLoggingServiceHome.class);

  // instantiate the sample parameter names processor
  paramNamesProc = new ParameterNamesProcessorSample();
}
```

4. Retrieving an instance of the portlet site analyzer logger: A logger instance depends on a specific portlet request. Its validity is bound to a single request processing cycle. Therefore, the instantiation of the logger must be performed for each request. The service object that is obtained in the previous step makes various methods available to obtain a logger instance. The following code sample shows the retrieval of a logger instance for the current `RenderRequest` in the `doView` method of a portlet. The instance of the parameter names processor sample is also passed in to the `getLogger` method in order to register this callback for the returned logger. Sample:

```
protected void doView(final RenderRequest renderRequest, final RenderResponse
       renderResponse) throws PortletException, IOException
{
  final PortletSiteAnalyzerLogger saLogger = iSALogServiceHome.getLogger(renderRequest,
       renderResponse, iParamNamesProc);

  // further request processing
}
```

5. Logging a business event: When a portlet uses the instance of the `PortletSiteAnalyzerLogger`, the logger writes a business event to the site analysis log file. Sample:

```
protected void doView(final RenderRequest renderRequest, final RenderResponse
     renderResponse) throws PortletException, IOException
{
  // request processing

  // check whether the logger is enabled
  if (saLogger.isLogging())
  {
    // create a site analysis log entry
    saLogger.log("A sample business event");
  }

  // further request processing
}
```

Results

Understanding the site analysis log for standard portlets: The site analysis log entries that are written by the PortletSiteAnalyzerLogger have the same structure as log entries created by other types of site analyzer loggers that the portal provides. This particular logger also implements the industry standard NCSA Combined.

The following example shows how the URL-encoded representation of the custom business event is written to the request URI section of a log record. Non-ASCII characters are first encoded as sequences of 2 or 3 bytes, by using the UTF-8 algorithm, before they are encoded as `%HH` escapes. The encoded business event precedes the query fragment, which starts with a question mark. Furthermore, the name of the request parameter `ci`was replaced by `ParameterNamesProcessorSample` and now displays in the query fragment of the request URI section of the log record as `CurrentItem`.

```
9.37.3.88 - jdoe [22/Nov/2008:22:11:27 +0100] "GET /Portlet/
5_8000CB1A00U6B02NVSPH1G20G1/SamplePortlet/A%20sample%20business%20event
?PortletPID=5_8000CB1A00U6B02NVSPH1G20G1&PortletMode=view&PortletState=normal
&RequestType=render&CurrentItem=9783000216008 HTTP/1.1" 200 -1
```

```
"http://myserver.company.com/Page/6_8000CB1A00UR402F0JC25U1O25/SamplePage"
"Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.18) Gecko/20081029
Firefox/2.0.0.18" "JSESSIONID=0000JwIm04xm7btVLwzCj9Qo-uj:-1"
```

Analyzing user behavior by Active Site Analytics

You can collect data about user behavior in your portal and send that data to a service for analysis. For this purpose the portal provides Active Site Analytics (ASA).

About this task

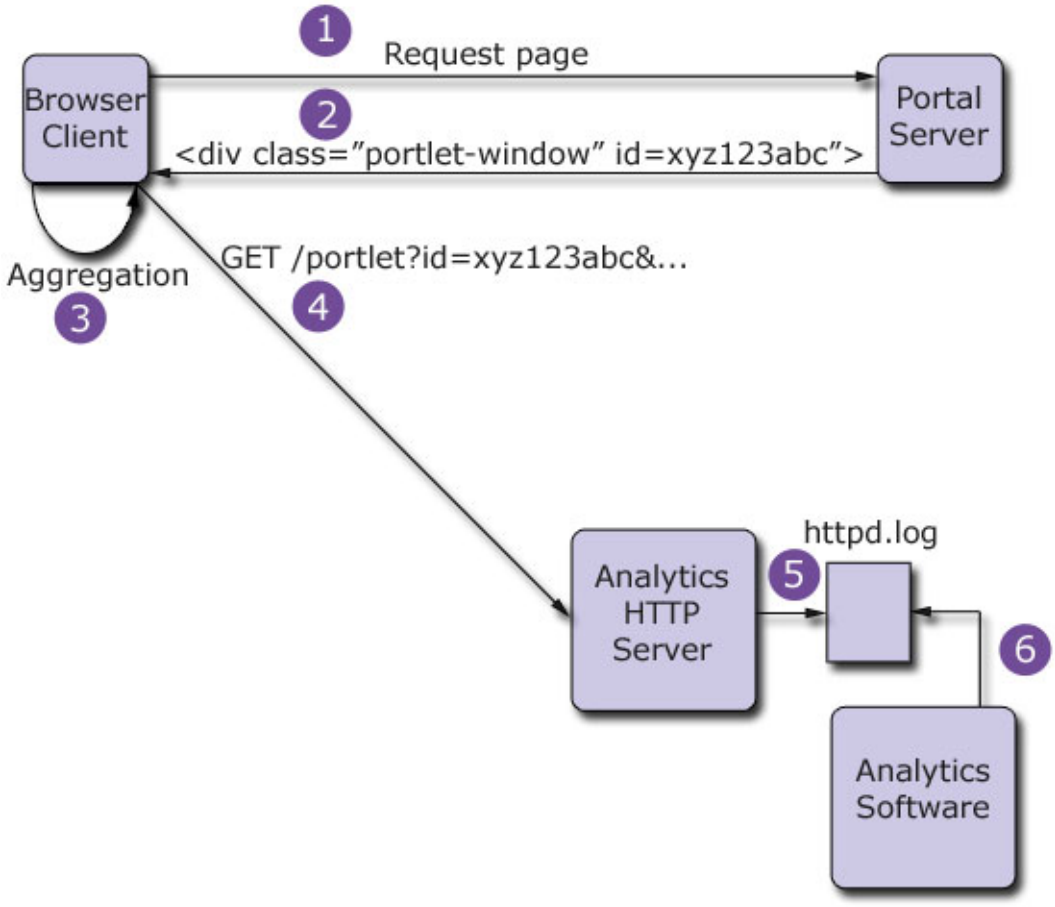
The portal provides a rich set of page metadata as part of its themes and skins. Examples are:

- Page title
- Page identifier
- Portlet title
- Portlet identifier.

You can write scripts to retrieve the data. Such scripts are called aggregators. WebSphere Portal Express provides several sample aggregators. They are located in the *PortalServer_root*/doc/js-samples directory of your portal installation. You can customize these or write your own aggregators to collect page metadata and more. For the samples, refer to the topic Writing an aggregator for active site analytics.

Administrators add the aggregator scripts to pages as required. When they do this, the aggregator is injected into markup of the page the next time it is rendered. You can select the point of injection arbitrarily; for example, this can be just before the

closing body tag of the HTML markup. Refer to the following flow graphic:



"Collecting analytics data" on page 1698
Before you can send data about user behavior in your portal to a service for analysis, you need to collect that data. See the following topics for information about how to do this.

"Displaying overlay analytics reports" on page 1714
You can use Active Site Analytics to show graphical statistics reports about individual portal resources, such as pages or portlets. These reports are called Active Site Analytics overlay reports.

"Analytics tags and site promotions" on page 1724
To obtain further analytics information from your portal, you can use analytics tags for your portal resources. You can also use analytics tags for site promotions.

Related tasks:

"Writing an aggregator for Active Site Analytics" on page 1701
You can write your own scripts to retrieve the data for Active Site Analytics from the portal themes and skins and connect to your analytics service provider. Such scripts are called aggregators.

Related information:

IBM WebSphere Portal V 8 Integration with IBM Digital Analytics (Coremetrics)

Collecting analytics data

Before you can send data about user behavior in your portal to a service for analysis, you need to collect that data. See the following topics for information about how to do this.

"How Active Site Analytics data is represented in the portal"
The data for the analysis of user behavior is retrieved from markup embedded in the portal pages.

"Writing an aggregator for Active Site Analytics" on page 1701
You can write your own scripts to retrieve the data for Active Site Analytics from the portal themes and skins and connect to your analytics service provider. Such scripts are called aggregators.

"Adding an Active Site Analytics aggregator to a portal page" on page 1709
Portal administrators can manage the aggregators. They can assign an aggregator to one or more portal labels or pages.

"Instrumenting a theme for Active Site Analytics" on page 1710
The portal theme provided with IBM WebSphere Portal Express is prepared and suitable for use with Active Site Analytics. You can also create your own custom theme to use the Active Site Analytics functionality.

"Configuring an aggregator at run time" on page 1713
If you want to configure the behavior of an aggregator at run time, you can pass additional meta data values to the aggregator.

How Active Site Analytics data is represented in the portal:

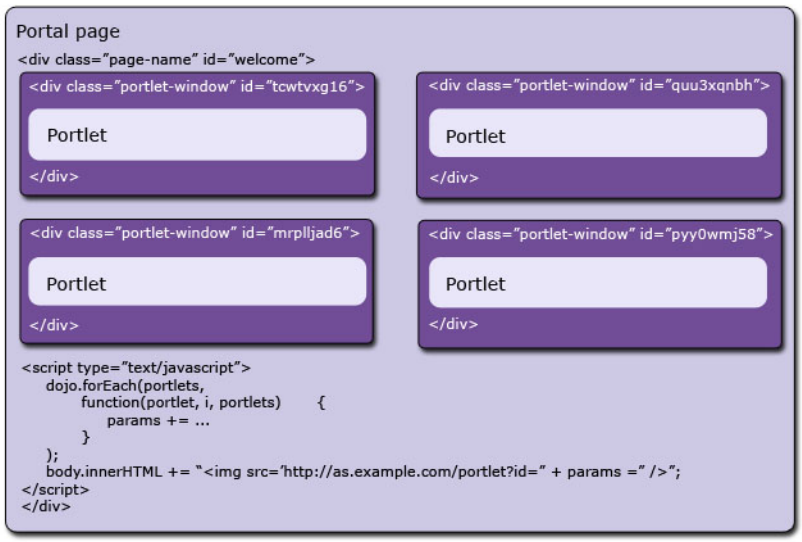
The data for the analysis of user behavior is retrieved from markup embedded in the portal pages.

The aggregator associated with the page formats the data, so that it corresponds to the requirements of the external analytics service to which it is finally submitted. By changing the Javascript the administrator can change the information that is retrieved from the page and submitted for analysis.

The data is represented as a microformat in the portal page HTML DOM tree and tagged with CSS classes. An example of such microformat is:

```
<span class="asa.portlet.title">My Portlet</span>
```

The structure of a page with annotated portlets is shown here:



"Supported aggregator tags"
The portal supports the aggregator tags listed in the following for Active Site Analytics.

Supported aggregator tags:

The portal supports the aggregator tags listed in the following for Active Site Analytics.

Page Breadcrumb
Page breadcrumb is the ObjectID for the breadcrumb of the current page. Tag it as `asa.page.breadcrumb`. It contains the path to the page within the hierarchical tree of pages. The path is made of the names of the pages, which are separated by a forward slash character.

Page ID
Page ID is the ObjectID of the page. Tag it as `asa.page.id` and add it to the theme markup.

Page Title
Page Title is the title of the page in the portal default language. Tag it as `asa.page.title` and add it to the theme markup.

Page Locale
Page locale is the locale of the page. Tag it as `asa.page.locale` and add it to the theme markup.

Page Locale Direction
Page locale direction is the direction of the locale of the page. Tag it as `asa.page.direction` and add it to the theme markup.

Friendly URL
Friendly URL is the URL of the current page without navigational state. Tag it as `asa.url` and add it to the theme markup. This element is empty if no friendly URL is set for the page.

Visitor ID
Visitor ID is the object ID of the user who currently logged in. If the user is anonymous, the visitor ID remains empty. Tag it as `asa.visitor` and add it to the theme markup.

Portlet Window Title
Portlet window title is the title of the portlet as it is delivered to the client. Tag it as `asa.portlet.title` and add it to the skin markup.

Portlet Locale
Portlet locale is the locale of the portlet as it is delivered to the client. Tag it as `asa.portlet.locale` and add it to the skin markup.

Portlet Locale Direction
Portlet locale direction is the direction of the locale of the portlet as it is delivered to the client. Tag it as `asa.portlet.direction` and add it to the skin markup.

Portlet Window ID
Portlet window ID is the unique identifier of the portlet. Tag it as `asa.portlet.id`. It is published to the Dojo topic queue by the topic name `com.ibm.portal.theme.portlet_ready`.

Portlet Screen ID
Portlet screen ID is the unique identifier of the screen or view that is

displayed in a portlet. Tag it as `asa.portlet.screen.id`. The portlet developer can add it to the custom portlet.

Portlet Screen Title
Portlet screen title is the localized title of the screen or view that is displayed in a portlet. Tag it as `asa.portlet.screen.title`. The portlet developer can add it to the custom portlet.

Selected Portlet
This tag marks the portlet with which the user interacted. Tag it as `asa.portlet.selected` and add it to the skin markup by the portal framework. For developers: do not manually add this tag to the skin.

Analytics Tags
The analytics tags are associated with a resource. Tag them as `asa.tag`*`analyticstag_name`* and add them to the page or portlet markup.

Site Promotions
These tags are the site promotions that are associated with a resource. Tag them as `asa.tag.SitePromotion` and add them to the page or portlet markup.

Web Content Manager Content Querystring
The Content Querystring is the unique identifier of the Web Content Manager content item that is displayed in a portlet. Tag it as `asa.wcm.content_item.path` and add it to the markup of the Web Content Manager Rendering portlet.

Web Content Manager Content Title
The Content title is the title of the Web Content Manager content item that is displayed in the Web Content Manager Rendering portlet. This tag must not be identical to the portlet window title. Tag it as `asa.wcm.content_item.title` and add it to the Web Content Manager Rendering portlet.

Web Content Manager Content ID
The Content ID is the unique identifier of the Web Content Manager content item that is displayed in a portlet. Tag it as `asa.wcm.content_item.id` and add it to the markup of the Web Content Manager Rendering portlet.

Web Content Manager Content Authors
The Content authors is the name of the author of the Web Content Manager content item that is displayed in a portlet. Tag it as `asa.wcm.content_item.authors` and add it to the markup of the Web Content Manager Rendering portlet. This tag can occur more than once.

Search term
Tag the search term as `asa.search.query` and add it to the markup of the Search Center portlet.

Number of search results
This the number value of the search results. Tag it as `asa.search.results` and add it to the markup of the Search Center portlet.

Search Scope ID
Tag the search scope ID as `asa.search.scope.id` and add it to the markup of the Search Center portlet.

Search Scope Label
Tag the search scope label as `asa.search.scope.label` and add it to the markup of the Search Center portlet.

Writing an aggregator for Active Site Analytics:

You can write your own scripts to retrieve the data for Active Site Analytics from the portal themes and skins and connect to your analytics service provider. Such scripts are called aggregators.

About this task

The portal themes that come with WebSphere Portal Express write the data that Active Site analytics needs into the markup. The data is provided in the form of microformats, which are injected into the markup during rendering.

The scripts that retrieve this data from the page markup are called aggregators. After retrieving the relevant microformat instances, the aggregator typically submits the collected data to an external analytics service where the data is then recorded, processed, and formatted in the form of reports.

The portal provides a client side JavaScript SPI that you can use to implement aggregators. The SPI is named **Active Site Analytics Mediator SPI**.

Aggregators that are based on the Active Site Analytics Mediator SPI behave consistently in WebSphere Portal Express, irrespective of the render mode of the portal page to which the aggregator is applied. This includes server side aggregation rendering and client side aggregation rendering modes. An aggregator can capture the latest data that is relevant for Active Site Analytics even in the client side aggregation mode that uses Ajax technologies to refresh individual page parts that are part of the DOM. Beyond that, the SPI also supports custom Ajax applications that can be part of a portal page.

From a programming model perspective, the Active Site Analytics Mediator SPI allows aggregator implementations to register callback functions; the portal framework calls these functions to notify the aggregator about DOM changes that can be relevant for Active Site Analytics. Upon receiving such notifications, aggregators can parse the DOM to find the relevant microformats to send this information to the analytics service.

For typical portal pages use the Active Site Analytics Mediator SPI for implementing your aggregators. However, for simpler pages you can also implement your aggregator by using a simple inline script which is executed at the end of your portal page. In this case the page must fulfill both of the following conditions:

- The page does not exploit client side aggregation rendering.
- Your applications on the page do not use Ajax technologies to refresh single page parts that might contain analytics-relevant microformats.

The SPI does not affect the way you can configure aggregators. For details about how to configure aggregators see the topic about *Adding an Active Site Analytics aggregator to a portal page*.

"The Active Site Analytics Mediator SPI" on page 1702
The portal provides a client side JavaScript SPI named Active Site Analytics Mediator SPI. You can use it to implement aggregators. The Active Site Analytics Mediator SPI allows aggregators to register callback functions; the portal framework calls these functions to notify the aggregator about DOM changes that can be relevant for Active Site Analytics.

"Guidelines for implementing an aggregator" on page 1703
When you implement an aggregator by using the Active Site Analytics Mediator SPI, the following guidelines can be helpful.

"Aggregator patterns and samples" on page 1704
The aggregator patterns and samples section provides common aggregator patterns and samples that you might want to adopt to implement your own aggregator.

"How to identify and resolve problems with your aggregator" on page 1708
If your custom aggregator is not working correctly, perform the checks listed here.

Related tasks:

"Adding an Active Site Analytics aggregator to a portal page" on page 1709
Portal administrators can manage the aggregators. They can assign an aggregator to one or more portal labels or pages.

The Active Site Analytics Mediator SPI:

The portal provides a client side JavaScript SPI named Active Site Analytics Mediator SPI. You can use it to implement aggregators. The Active Site Analytics Mediator SPI allows aggregators to register callback functions; the portal framework calls these functions to notify the aggregator about DOM changes that can be relevant for Active Site Analytics.

You can get access to the Active Site Analytics Mediator SPI by using the following global JavaScript variable:

com.ibm.portal.analytics.SiteAnalyticsMediator
Note that the `SiteAnalyticsMediator` object is not available until the page has finished loading. The `SiteAnalyticsMediator` object defines the following JavaScript functions:

register: function(/*Function*/ listener)
Use this function to register a listener function. You can invoke this listener function by using the portal framework in case of DOM changes that might be relevant for the aggregator. Typically, an aggregator registers one function. The register function returns a string identifier that you can use for unregistering the listener later on.

The provided listener function must conform to the following function signature:
`function (/*DOMNode[]?*/ node, /*Function?*/ callback)`
The optional argument `DOMNode[]` indicates the areas in the DOM that have changed. If no `DOMNode` array is provided, the aggregator can assume that major parts of the DOM have changed. In this case make the aggregator operate on the global document object. You can use the second argument by the framework to optionally pass in a callback function. That callback function must be called by the aggregator after finishing the DOM parsing process.

deregister : function(/*String*/ listenerID)
Use this function to unregister a listener function with the identifier that you have obtained during listener registration.

notify : function(/*DOMNode[]?*/ node, /*Function?*/ callback)
Use this function to notify all registered listeners about DOM changes. If you know the specific DOM area that has changed, you can pass in the corresponding DOM nodes. This function is

typically called by the portal framework to notify the registered aggregators about DOM changes. It can also be used by AJAX applications that partially update the DOM. If no specific DOM nodes are provided, the aggregator may assume that the notification refers to the portal page. If DOM nodes are provided, the notification refers to portal page elements. In addition to the `DOMNode` array you can optionally provide a callback function.

Guidelines for implementing an aggregator:

When you implement an aggregator by using the Active Site Analytics Mediator SPI, the following guidelines can be helpful.

1. Implement the function that parses the DOM to collect and format all Active Site Analytics-specific data.
2. Subscribe for the DOM notifications. Register the function that you implemented in the previous step with the object `SiteAnalyticsMediator`.
3. Implement the logic that sends the collected data to the external analytics service.

When performing these steps, apply the following considerations:

1. Implement the function that parses the DOM to collect and format all Active Site Analytics-specific data.
   - The implementation for collecting the data greatly depends on the themes, skins, and all the other components that might produce analytics-specific microformats. So, there is a dependency between the theme and its skins on the one hand and the aggregator.
   - The aggregator implementation can affect the performance, as it increases the page rendering time in the browser. To mitigate this time increase, take care that the DOM processing required to collect the data is implemented efficiently. To achieve a good performance, you can use implementation details of your themes and skins. For example, you can use HTML identifiers in the theme markup to isolate the nodes that contain the Active Site Analytics-specific data into a DOM subtree that your aggregator can access efficiently. In any case, avoid traversing the entire DOM.
2. Subscribe for the DOM notifications. Register the function that you implemented in the previous step with the object `SiteAnalyticsMediator`.
   - It is only during this registration step that you directly invoke the Active Site Analytics Mediator SPI. Register your function with the `SiteAnalyticsMediator` before the browser sends the onload event.
3. Implement the logic that sends the collected data to the external analytics service.
   - To have your data analyzed, you send them to an external analytics service. This step is separated from step 1, as the aggregator cannot make assumptions as to how often it is notified about DOM changes. In a Web 2.0 portal environment with many Ajax components, multiple notifications per portal page or even per portlet interaction can occur. Therefore, this step requires a deep understanding of the analytics service. Even the contract with your analytics service provider can influence when and how frequently you want to send the collected data to the analytics service.
   - If it does not matter how frequently data is transmitted to the analytics service, you can combine steps 1 and 3. For example, you can send the data directly to the analytics service after or even during the DOM parsing process. If the transmission frequency is relevant, store the data in a buffer, for example a JSON object or a cookie, during the DOM parsing process in

step 1. To trigger sending the buffer contents to the analytics service, you can either use an external event or a specific data change. The trigger can be based on a time interval, for example a JavaScript timeout. You can also use the `beforeunload` event that the browser fires whenever the entire browser page is refreshed, or when a browser window or browser tab is closed.

- You have several options for how to send the collected data to the external analytics service. A widely spread method is to use a so-called tracking image. In this case, the aggregator adds an HTML image element to the DOM, which is hidden via CSS. The image source URL that points to the external analytics service is used to transport the collected data. After adding the image element to the DOM, the browser initiates an HTTP GET request to load the tracking image. As an alternative, you can also use an asynchronous `XmlHttpRequest` (XHR) to send the data to the external analytics service. This method is useful if you want to use HTTP POST requests to submit a large amount of data.

The following list gives a summary of the guidelines:

1. Register your handler before the page finishes loading, that is, before the browser sends the onload event.
2. Do not make assumptions about the frequency by which your aggregator gets notified about DOM changes. For a single page, multiple notifications can occur.
3. Use separate steps for collecting and formatting the data, and for submitting the data to the external analytics server.
4. The processing that is done by your aggregator affects the page rendering time in the browser.
5. To ensure a better performance, use theme and skin implementation details. For example, use HTML identifiers for efficient DOM node lookups if available. Avoid traversing the entire DOM.
6. Carefully decide when and how often you want to send the collected data to the analytics service. This decision might depend on your analytics service provider.
7. Use an asynchronous way to send the collected data to the analytics service, for example a tracking image or an asynchronous XHR.
8. You can use a JavaScript framework to implement your aggregator.

For code samples see *Aggregator patterns and samples.*

Related reference:

"Aggregator patterns and samples"
The aggregator patterns and samples section provides common aggregator patterns and samples that you might want to adopt to implement your own aggregator.

Aggregator patterns and samples:

The aggregator patterns and samples section provides common aggregator patterns and samples that you might want to adopt to implement your own aggregator.

JavaScript module pattern template

You can use the following JavaScript module pattern as a general template:

```
/**
 * Check if the aggregator module is already defined.
 */
if (typeof(ibm_analytics_MyAggregator) === "undefined") {
```

```
/**
 * Function to construct the aggregator
 */
ibm_analytics_MyAggregator = function(/* Arguments needed internally */) {

  /**
   * The aggregator instance
   */
  var aggregator = {};

  // define your internal variables
  var a = ...;
  var b = ...;
  // ...

  /**
   * Public function which parses the given DOM nodes to detect
   * the microformats the aggregator is interested in.
   */
  aggregator.parse = function(/*DOMNode[]*/ nodes, /*Function?*/ callback) {
    // check if a specific part of the DOM should be parsed
    if (nodes && nodes.length > 0) {
      /*
       * Page element notification
       */
      // invoke your internal parsing code here
      // ...
    } else {
      /*
       * Page notification
       */
      // invoke your internal parsing code here
      // ...
    }
    // invoke the callback (if any)
    if (callback) {
      callback();
    }
  };

  /**
   * Public function to submit the collected data to the
   * analytics service.
   */
  aggregator.submit = function() {
    // send the collected data off to the external analytics service
    // use your preferred technique here (tracking image, XHR etc.)
    // ...
  };

  // define your internal functions
  var parseNode = function(/*DOMNode*/ node) {
    // ...
  };

  // ...

  // return the aggregator
  return aggregator;

};

};
```

By using this pattern, the aggregator module exports two functions:

1. The `parse()` function is responsible for handling the DOM notifications that are sent by the SiteAnalyticsMediator. It complies with the function signature that is defined by the Active Site Analytics Mediator SPI as discussed in the topic about *The Active Site Analytics Mediator SPI*. Use this function to parse the DOM to find the analytics-specific microformats that interest you. Typically, the parse function stores the found microformats in an internal bucket, for example a JSON object, that is transmitted to the external analytics service later on. In the JavaScript module pattern template, you can also see how the aggregator can distinguish between page notifications and page element notifications (a page element can either be a portlet or an Ajax component).
2. The `submit()` function is responsible for sending the collected data to the external analytics service. Use your preferred technique to initiate an HTTP request, which carries the data to the analytics service. For performance reasons, communicate the data asynchronously, for example by a tracking image or an asynchronous XmlHTTPRequest.

Note: The registration aspect and the logic that triggers the data transmission are not part of the aggregator module. These aspects need to be handled outside of the aggregator module.

Registering with the Site Analytics Mediator

Register your `parse()` function with the Site Analytics Mediator immediately after the definition of the aggregator module:

```
// instantiate and initialize the aggregator
var ibm_analytics_aggregator = new ibm_analytics_MyAggregator(/* arguments as needed...*/);
// register the parse function of the aggregator with the mediator
com.ibm.portal.analytics.SiteAnalyticsMediator.register(function() {
  ibm_analytics_aggregator.parse.apply(ibm_analytics_aggregator, arguments);
});
```

Submitting the analytics data

You can submit the data in various ways. You can submit the data internally by using the aggregator. For example, you can submit data upon receiving a notification or based on the internal state of the aggregator. Or you can trigger data submission from outside based on an external event. This decision can depend on the business model of your external analytics service provider. For more information, see *Guidelines for implementing an aggregator*.

To trigger the data submission that is based on an external event, you can either use the DOM `beforeunload` event of the browser or a time-based solution. The time-based solution sends the data periodically. You can also combine the two approaches.

To use the DOM `beforeunload` event of the browser, subscribe to the `beforeunload` event by using the `submit()` function of your aggregator as shown by the following sample:

```
// instantiate and initialize the aggregator
var ibm_analytics_aggregator = new ibm_analytics_MyAggregator(/* arguments...*/);

// register the parse function of the aggregator with the mediator
com.ibm.portal.analytics.SiteAnalyticsMediator.register(function() {
  ibm_analytics_aggregator.parse.apply(ibm_analytics_aggregator, arguments);
});

// callback function to submit the collected data to the external analytics service
var ibm_analytics_MyAggregator_submit = function() {
```

```
  ibm_analytics_aggregator.submit.apply(ibm_analytics_aggregator, arguments);
}

// register the callback functions
if (window.addEventListener) {
  // W3C
  window.addEventListener("beforeunload", ibm_analytics_MyAggregator_submit, false);
} else if (window.attachEvent) {
  // Microsoft
  window.attachEvent("onbeforeunload", ibm_analytics_MyAggregator_submit);
}
```

If you use client-side aggregation rendering for your portal pages, full page refreshes and therefore DOM unload events do not occur often. In this case and with this approach, the aggregator collects data for a long time without sending the collected data to the external analytics service. To avoid this behavior, you can combine the unload approach with a time-based approach. For example, you can submit the collected data periodically, in addition to sending it off upon receiving a `beforeunload` event. To implement the periodic data submission, use the `setInterval()` JavaScript API that all major browsers provide. In this case, you register the `submit()` function as the interval handler within your onload handler. The following code sample uses a time interval of 30 seconds. This action means that the data is submitted every 30 seconds at the latest if no `beforeunload` event is received.

```
// instantiate and initialize the aggregator
var ibm_analytics_aggregator = new ibm_analytics_MyAggregator(/* arguments...*/);

// register the parse function of the aggregator with the mediator
com.ibm.portal.analytics.SiteAnalyticsMediator.register(function() {
  ibm_analytics_aggregator.parse.apply(ibm_analytics_aggregator, arguments);
});

// callback function to initialize and register the aggregator
var ibm_analytics_MyAggregator_init = function() {
  // register the interval handler
  setInterval(ibm_analytics_MyAggregator_submit, 30000);
};

// callback function to submit the collected data to the external analytics service
var ibm_analytics_MyAggregator_submit = function() {
  ibm_analytics_aggregator.submit.apply(ibm_analytics_aggregator, arguments);
}

// register the callback functions
if (window.addEventListener) {
  // W3C
  window.addEventListener("load", ibm_analytics_MyAggregator_init, false);
  window.addEventListener("beforeunload", ibm_analytics_MyAggregator_submit, false);
} else if (window.attachEvent) {
  // Microsoft
  window.attachEvent("onload", ibm_analytics_MyAggregator_init);
  window.attachEvent("onbeforeunload", ibm_analytics_MyAggregator_submit);
}
```

If you have Ajax applications on your portal sites that change markup dynamically by using JavaScript, you need to notify the SiteAnalyitcsMediator framework manually. You can notify by calling the `notify()` function that is provided by the SiteAnalyticsMediator object.

```
// determine the root nodes of the DOM subtrees that have changed
var nodes = ...;
com.ibm.portal.analytics.SiteAnalyticsMediator.notify(nodes, callback);
```

Aggregator samples

WebSphere Portal Express provides several sample aggregators. They are in the *PortalServer_root*/doc/js-samples directory of your portal installation. Although the provided sample code does not require any special JavaScript library, you might want to use the JavaScript framework of your choice. With the JavaScript framework, you can reduce the size and complexity of your aggregator implementations. And the JavaScript framework might support you in achieving a good performance. To explain the concepts that are discussed in the previous topics, the sample aggregators are simplified with regards to the data that is evaluated. You can install each of the provided sample aggregators on WebSphere Portal Express as described in the topic about *Adding an Active Site Analytics aggregator to a portal page*. WebSphere Portal Express provides the following sample aggregators:

CoremetricsAggregator.js
: This sample aggregator collects the analytics data and sends it to the IBM Coremetrics Web Analytics service. It uses a Coremetrics specific JavaScript API to submit the data. You need to configure this API as an aggregator dependency. The target URL is `http://libs.coremetrics.com/eluminate.js`. Before you can use the aggregator, you also need to edit the aggregator code to specify your Coremetrics client ID, cookie domain, and data collection domain.

SampleAggregator.js
: This aggregator is a generic aggregator. It collects all the analytics data and sends it to the URL `http://example.org` by using a tracking image. The collected data is appended to the image URL as a query string. The names of the URL parameters correspond with the microformat names as defined in *Supported aggregator tags*.

Related tasks:

"Adding an Active Site Analytics aggregator to a portal page" on page 1709
Portal administrators can manage the aggregators. They can assign an aggregator to one or more portal labels or pages.

Related reference:

"Guidelines for implementing an aggregator" on page 1703
When you implement an aggregator by using the Active Site Analytics Mediator SPI, the following guidelines can be helpful.

"Supported aggregator tags" on page 1699
The portal supports the aggregator tags listed in the following for Active Site Analytics.

How to identify and resolve problems with your aggregator:

If your custom aggregator is not working correctly, perform the checks listed here.

1. Verify that the HTML markup of the portal page contains the microformats that the aggregator expects.
2. Verify that the tagging method that is used in the themes, skins, and portlets matches the expected behavior of the aggregator.
3. If you use the SiteAnalyticsMediator API and your aggregator is not notified about DOM changes as expected, you can enable client side tracing by using the resource environment provider **WP CommonComponentConfigService**. Set the following two configuration properties:

   ```
   cc.isDebug=true
   cc.traceConfig=["com.ibm.wps.analytics.*"]
   ```

After you have done this, save your changes and restart the portal server.

4. To review what occurs in the server side aggregator inclusion code, specify the trace `com.ibm.wps.theme.extensions.*=debug` .

Remember that aggregators are regular JavaScript files. Consequently, you can also use all JavaScript debugging tools and browser developer tools for debugging an aggregator.

Adding an Active Site Analytics aggregator to a portal page:

Portal administrators can manage the aggregators. They can assign an aggregator to one or more portal labels or pages.

About this task

As a portal administrator, you assign an aggregator to a page by editing the page properties and adding a new parameter. To do this by using the portal administration portlets, proceed by the following steps:

Procedure

1. Add the aggregator to the directory `js` in the theme directory. For example, for the portal theme, the WebDAV location is as follows:

   ```
   mycontenthandler/dav/themelist
   ```

   Take a note of the name of the aggregator file. You need it in later steps. Go to the **Analytics** section of *Modules that are provided with the modularized theme* for information.
2. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
3. Locate the page to which you want to assign the aggregator. Use the Manage Pages portlet to locate the page.

   Note: The page must be located as a child of the portal content root.
4. Click **Edit Page Properties** for the page that you selected.

   Note: You cannot add an Active Site Analytics aggregator to the content root. **Edit Page Properties** is not available for the content root.
5. To expand the available choices, click the plus sign (**+**) icon next to **Advanced Options**.
6. Click **I want to set parameters**.
7. In the field **New parameter**, type a string that starts with `asa_aggregator` or `asa_dependency`. Values that correspond to names that start with the string `asa_aggregator` are added to the page body, names that start with `asa_dependency` are added to the head. Both the aggregators and dependencies are added to the portal page in alphabetical order according to the Java method `Collections.sort()`.
8. In the **New value** field, type the name of the aggregator script file.
9. Click **Add**.
10. Verify that the new parameter is added to the list.
11. Repeat steps 6 - 9 for all aggregators and dependencies.
12. Click **OK** to return to the main Page Properties screen.
13. Click **OK** to save your changes and return to the **Manage Pages** screen.

What to do next

Note: Children pages inherit the script that is set on the parent page. If you want to use a different aggregator on a child page, follow the same procedure that is previously addressed for the child page to make the appropriate assignment. If you want to block inheriting the aggregator setting from the parent page, follow the same procedure, but leave the value empty. As a result, the page has no aggregator that is assigned anymore and all children of the page inherit the new setting.

Related concepts:

"Modules that are provided with the modularized theme" on page 2608
WebSphere Portal Express provides a set of ready-to-use modules.

Instrumenting a theme for Active Site Analytics:

The portal theme provided with IBM WebSphere Portal Express is prepared and suitable for use with Active Site Analytics. You can also create your own custom theme to use the Active Site Analytics functionality.

About this task

In order make Active Site Analytics work in a custom theme, you need to do the following tasks:

1. Add metadata to the page.
2. Include an aggregator with the page.
3. Include dependencies with the page.
4. Include microformats of interest, that is related to the statistical data that you want to collect.

For more details about these tasks refer to the following sections:

Adding metadata to the page

An aggregator picks up information stored in the DOM (document object model) tree of a page. This information must be present in the HTML source of the page. The recommended approach to add this information to a page is to implement the theme or skin so that they write all the necessary information into the DOM tree of the page. Examples:

1. You can add the identifier of a portlet to the DOM tree by adding the following line to the file `skin.html` in the theme WebDAV folder:

   ```
   <span class="asa.portlet.id" style="display:none;"><%= myPortletID %></span>
   ```

   Note: If the `span` element is styled in an external CSS file, you need to escape each period in the class name by a backslash. Example: `asa\.portlet\.id{display:none;}`
   With the code sample, the aggregator can fetch the identifiers of all portlets on a page by iterating over all occurrences of `"span"` elements with a class attribute of `"asa.portlet.id"`.

   The correct function of the code sample depends on the definition of `myPortletID` earlier in the file `skin.html`. By default, it is defined by using the following JSP code:

   ```
   <portal-skin:portletID var="myPortletID"/>
   <jsp:useBean id="myPortletID" class="java.lang.String" scope="page"/>
   ```

2. You can add the portlet title to the DOM tree by wrapping the `<portal-skin:portletTitle>` statement in the file `skin.html` with a span element that has a class attribute of `"asa.portlet.title"`:

```
<span class="asa.portlet.title"><portal-skin:portletTitle></portal-skin:portletTitle</span>
```

Including aggregators with the page

The portal includes a default implementation of the theme extension point `com.ibm.portal.theme.plugin.ActiveSiteAnalyticsAggregators`. The extension point locates a JavaScript file that is specified in the page metadata. The extension point includes the content of that file in the HTML code of the page. The metadata key must start with `asa_aggregator` or `asa_js` , and its value is the JavaScript file name. For details see the topic about *Adding an Active Site Analytics aggregator to a portal page*. Add the following code somewhere near the closing body tag (`</body>`) of the file *themeName*`\nls\theme_`*languageCode*`.html` of the theme WebDAV folder:

```
<portal-theme-ext:themeExtension id="com.ibm.portal.theme.plugin.ActiveSiteAnalyticsAggregators">
   <portal-theme-ext:themeExtensionLoop>
      <portal-theme-ext:themeExtensionItemText />
   </portal-theme-ext:themeExtensionLoop>
</portal-theme-ext:themeExtension>
```

This code loops over every implementation of the theme extension point and executes it. The default implementation included with the portal follows the approach previously outlined.

Note: WebSphere Portal Express Version 8.5 does not support the extension point with the ID `com.ibm.portal.theme.plugin.ActiveSiteAnalyticsItems` from previous versions. If you have migrated a theme from an earlier portal version, make sure to use the extension ID `com.ibm.portal.theme.plugin.ActiveSiteAnalyticsAggregators` .

Including dependencies with the page

If you want dependencies to be used by more than one aggregator, add them to the head of the html page. The portal includes a default implementation of the theme extension point `com.ibm.portal.theme.plugin.ActiveSiteAnalyticsDependencies` . The extension point locates a JavaScript file that is specified in the page metadata. The extension point includes the content of that file in the HTML code of the page. The metadata key must start with `asa_dependency` , and its value is the JavaScript file name. For details see the topic about *Adding an Active Site Analytics aggregator to a portal page*. Add the following code to the head of the theme:

```
<portal-theme-ext:themeExtension id="com.ibm.portal.theme.plugin.ActiveSiteAnalyticsDependencies">
   <portal-theme-ext:themeExtensionLoop>
      <portal-theme-ext:themeExtensionItemText />
   </portal-theme-ext:themeExtensionLoop>
</portal-theme-ext:themeExtension>
```

This code loops over every implementation of the theme extension point and executes it. The default implementation included with the portal follows the approach previously outlined.

Including an aggregator JSP with the page

Active Site Analytics can include a dedicated JSP fragment as part of the aggregator inclusion on the page. The portal appends the contents of the JSP fragment to the content that is injected into the `theme.html` of the theme.

You configure the JSP similarly to the aggregator itself: Add page metadata with the key of `asa_jsp` to the page. The portal interprets the value as a file name relative to the `theme.html` of the theme in WebDAV. For example, if you want to add markup from a JSP in a file named `page.jsp` , you can

put this JSP next to `theme.html` using WebDAV and configure a new page property with the key `asa_jsp` and the value `page.jsp` .

Example: If the key of a piece of metadata is `asa_jsp` and its value is set to `asa_sample.jsp`, then the portal searches for a JSP with a file name `asa_sample.jsp` and, if it finds such a file, it includes it in the page contents.

You can use the extension point `com.ibm.portal.theme.plugin.ActiveSiteAnalyticsInclude` to include a JSP into the page content. The metadata key must start with `asa_jsp` . Add the following code in the theme:

```
<portal-theme-ext:themeExtension id="com.ibm.portal.theme.plugin.ActiveSiteAnalyticsInclude">
   <portal-theme-ext:themeExtensionLoop>
      <portal-theme-ext:themeExtensionRenderInclude/>
   </portal-theme-ext:themeExtensionLoop>
</portal-theme-ext:themeExtension>
```

This code loops over each implementation of the theme extension point and executes it. The default implementation included with the portal follows the approach outlined earlier.

Including microformats of interest

You might need to modify the skins used by a theme so that all of the microformat information that you want to be captured is present when the page is rendered. For example, you can add this code to the `skin.html` file of a skin to ensure the portlet ID of all portlets on the page is available for the aggregator script to discover:

```
<span class="asa.portlet.id" style="display:none;"><%= myPortletID %></span>
```

In this case the file `asa_sample.js` looks for elements with the class `asa.portlet.id` to find the portlet ID of all portlets present on the page. Portlet titles can be rendered by using code similar to this:

```
<portal-skin:portletTitle />
```

To capture such portlet titles, you can add an additional span element with a class defined for titles that the analytics JavaScript file will look for:

```
<span class="asa.portlet.title"><portal-skin:portletTitle /></span>
```

In this case, `asa.portlet.title` is recognized as the class for all span elements that encompass the rendered text of the portlet titles.

"Injecting custom aggregators"
By implementing a new theme extension point, you can apply different approaches to Active Site Analytics.

Related tasks:

"Adding an Active Site Analytics aggregator to a portal page" on page 1709
Portal administrators can manage the aggregators. They can assign an aggregator to one or more portal labels or pages.

Related information:

"Tags used by the portal JSPs" on page 2790
Learn about the most commonly used tags in the portal JSPs. Use these tags to modify the appearance and layout of the portal page.

Injecting custom aggregators:

By implementing a new theme extension point, you can apply different approaches to Active Site Analytics.

About this task

To do this, define your implementation within the following theme extension point in the file `plugin.xml`:

`com.ibm.portal.theme.plugin.ActiveSiteAnalyticsAggregators`

If the theme extension loop is in place, it picks up the custom implementation of the extension point, and its output is added to the markup of the portal page. If you want to have the output added to the head of the HTML page, implement the following extension point:

`com.ibm.portal.theme.plugin.ActiveSiteAnalyticsDependencies`

For details about the extension loop refer to the topic about Instrumenting a special theme for Active Site Analytics under the section about Including microformats of interest.

Alternatively, you can use the following JSP code to iterate over the page metadata and add the aggregator JavaScript include statement to the page:

```
<portal-logic:pageMetaData varname="pageMetaData">
   <portal-logic:urlFindInTheme file='js/${pageMetaData["asa_aggregator"]}'
         id="asa_aggregator_file"/>
   <c:if test="${asa_aggregator_file != null}">
      <script type="text/javascript" src='<%=asa_aggregator_file%>'></script>
   </c:if>
</portal-logic:pageMetaData>
```

Note: The result of `urlFindInTheme` is cached. To clear the cache, restart the portal. Redeploying the theme is not sufficient to trigger a reevaluation of `urlFindInTheme`.

Related information:

"Tags used by the portal JSPs" on page 2790
Learn about the most commonly used tags in the portal JSPs. Use these tags to modify the appearance and layout of the portal page.

Configuring an aggregator at run time:

If you want to configure the behavior of an aggregator at run time, you can pass additional meta data values to the aggregator.

About this task

All page metadata with names that start with `asa_js` is an attribute of the global Javascript object called `ibm_page_metadata`. This object is only defined if at least one page metadata exists the key of which starts with `asa_js` .

There are two different ways by which metadata values are evaluated:

1. If the value starts with an equals sign (`=`), the value is interpreted as a JavaScript snippet. This evaluation starts with the first character after the equals sign.

   Example: If the key of a metadata is `<code>asa_js_foo</code>` and its value is set to `<code>='foo' + 'bar'</code>` , then the page meta data object `<code>ibm_page_metadata</code>` defines an attribute `<code>foo</code>` with a value of `<code>foobar</code>` . Refer to the following code snippet:

   ```
   <script language='JavaScript'>
        var ibm_page_metadata = {
             'foo' : 'foo' + 'bar'     }
   </script>
   ```

The aggregator can then access the metadata by either of the following ways: <code>ibm_page_metadata['foo']</code> or <code>ibm_page_metadata.foo</code> . It will evaluate to foobar .

The value part of page metadata is evaluated at run time in the client browser.

2. All other values are treated as JavaScript strings. No further escape mechanism is applied.

   Example: If the key of a piece of metadata is <code>asa_js_bar</code> , and its value is set to <code>foo</code> , then the page metadata object <code>ibm_page_metadata</code> defines an attribute <code>bar</code> with a value of <code>foo</code> .

   ```
   <script language='JavaScript'>
        var ibm_page_metadata = {
             'bar' : 'foo'     }
   </script>
   ```

   The aggregator can then access the meta data by either of the following ways: <code>ibm_page_metadata['bar']</code> or <code>ibm_page_metadata.bar</code> .

   Note: As the page metadata section requires administrative rights for the page, the portal performs no further checks. It passes all values entered as metadata keys and values to the JavaScript object literally.
   If an error in the metadata value prevents the page from being rendered, you can still change or remove the value by using other administrative ways to access the portal. For example, this can be WebDAV, the XML configuration interface, or the Portal Scripting Interface.

Displaying overlay analytics reports

You can use Active Site Analytics to show graphical statistics reports about individual portal resources, such as pages or portlets. These reports are called Active Site Analytics overlay reports.

About this task

The statistics graph is shown on the portal page in the format of an overlay over the portal resource to which they apply, such as the portlet. Examples of what statistical graphs can show:

- A simple view count of a portal page, that is how many users visited and viewed the page on a particular day.
- A statistical curve of the usage for a specific portlet over time.

The statistics are generated as follows:

1. The data collected by the portal site analytics are collected and forwarded to your business partners for portal analytics.
2. Your business partner analyses and evaluates the portal data and sends back a report to your portal.
3. The portal then displays the report as the overlay graph.

You can define which users can view the reports by setting the appropriate access rights.

The user must at least be in the **USER** role on the virtual resource **OVERLAYREPORTS** and at least in the **USER** role of the resource he wants to view reports for.

Note: You can enable overlay statistics only if you already use **Active Site Analytics** on your portal site.

To set up Active Site Analytics overlay reports, perform the procedures given here.

Procedure

1. Enable data collection. For example, you can do this by adding an Active Site Analytics aggregator to a portal page:
   a. Access the IBM Coremetrics Web Analytics website or contact your Coremetrics representative.
   b. Retrieve the appropriate aggregator file from Coremetrics.
   c. Upload the aggregator file to your portal theme folder, for example, by using WebDAV.
   d. Perform the procedure described under *Adding an Active Site Analytics aggregator to a portal page*.

   As a result of these steps, analytics data is sent from browsers of your portal users to the Coremetrics data collection servers. You can log on to Coremetrics and analyze the portal usage.
2. Enable inline display of Coremetrics reports by enabling overlay reports:
   a. Establish trust with Coremetrics servers. For more information, see *Configuring overlay reports*.
   b. Optional: Set up security for overlay reports. For details see *Configuring security for overlay analytics reports*.
   c. Configure your Coremetrics user ID. For more information, see *Configuring a Credential Vault for overlay reports*.
   d. Configure your theme to integrate the overlay report features into the **Actions** menu. For more information, see *Configuring the theme for overlay reports*.
   e. Display overlay reports by choosing **Show Portlet Reports** or **Show Page Reports**. For more information, see *Viewing overlay analytics statistics*.
   f. Optional: Customize the overlay report by providing additional configuration parameters. For more information, see *Customizing overlay reports*.
3. Optional: To further customize the tagging of your site, assign site promotions to pages and portlets as required.

What to do next

For more information about enabling inline display of overlay reports, see the following topics.

1. "Configuring overlay reports" on page 1716
   In order to activate overlay reports for your site, you need to configure some security-related settings.
2. "Configuring security for overlay analytics reports" on page 1717
   You can administer which users can view overlay reports. To do this, you use the virtual resource `OVERLAY_REPORTS`.
3. "Configuring a Credential Vault for overlay reports" on page 1718
   To access the IBM Coremetrics Web Analytics system, you have to store your user information in a Credential Vault slot. If you do not do this, the portal overlay reports cannot show data from the Coremetrics system.

4. "Configuring the theme for overlay reports" on page 1720
   To integrate the overlay analytics features into your theme, add a theme module to the theme profiles of your modular theme. If you do not add the theme, the overlay menu items are not displayed.
5. "Viewing overlay analytics statistics" on page 1721
   Users with the appropriate access rights can use Active Site Analytics to view graphical statistics about individual portal resources, such as pages or portlets.
6. "Customizing overlay reports" on page 1721
   You can customize your own overlay reports by setting specific parameters as required. Learn about the parameters and the levels at which you can specify them.

Related tasks:

"Adding an Active Site Analytics aggregator to a portal page" on page 1709
Portal administrators can manage the aggregators. They can assign an aggregator to one or more portal labels or pages.

Related information:

IBM WebSphere Portal V 8 Integration with IBM Digital Analytics (Coremetrics)

Configuring overlay reports:

In order to activate overlay reports for your site, you need to configure some security-related settings.

About this task

To allow connections with the report provider, you need to configure the AJAX Proxy. Proceed as follows:

Procedure

1. Add the IBM Coremetrics Web Analytics site to the default dynamic policy. To do this, proceed with the following steps:
   a. Open the WebSphere Integrated Solutions Console.
   b. Click **Resources** > **Resource Environment** > **Resource Environment Providers** > **WP ConfigService** > **Custom Properties** > **New...**.
   c. Use the property name `wp.proxy.config.urlreplacement.default_policy.1` and the value `https://welcome.coremetrics.com/*` .

      Note: If this name is already in use, increment the counter of the key `wp.proxy.config.urlreplacement.default_policy` .
   d. Click **Apply**.
   e. Click **Save**.
2. Import the Coremetrics certificate into your server truststore. You can complete this step either by using the `wsadmin` command `retrieveSignerFromPort` of the `AdminTask` object, or manually as described here:
   a. Open the WebSphere Integrated Solutions Console.
   b. In a stand-alone enviroment: Click **Security** > **SSL certificate and key management** > **SSL Configurations** > **NodeDefaultSSLSettings** > **Key stores and certificates** > **NodeDefaultTrustStore** > **Signer certificates** > **Retrieve from port**.

In a portal cluster environment: Click **Security** > **SSL certificate and key management** > **SSL Configurations** > **CellDefaultSSLSettings** > **Key stores and certificates** > **CellDefaultTrustStore** > **Signer certificates** > **Retrieve from port**. By alternative, you can also select each node individually and apply the following settings to each node in the cell.

c. Fill the fields as follows:
   - For the host: `welcome.coremetrics.com`
   - For the port: `443`
   - For the alias: `coremetrics` or another identifying string of your choice.

d. Click **Retrieve signer information**.

e. Click **Apply**.

f. Click **Save**.

3. Store your Coremetrics user information in a credential vault slot as described under Configuring credential vaults for overlay reports.
4. Restart the portal server for the changes to take effect.
5. Optional: If required, configure the limit of outbound HTTP connections. The data for the overlay reports is retrieved through the AJAX proxy. The AJAX proxy has a configuration parameter for setting the maximum number of concurrent outbound connections. This limit also affects the number of overlay reports that can be retrieved from the Coremetrics API server concurrently. If you want to have more overlay reports retrieved, configure the limit as described under *Configuration metadata for outbound HTTP connections*.

What to do next

Note: Additional to the limit set by the AJAX proxy, Coremetrics also enforces a limit on the number of concurrent connections from the same host. When that limit is reached, the portal displays message `EJQGB0000E` instead of the overlay report. In this case the user can refresh the portal page until the error message disappears; the successful analytics results are cached in the browser.

Related reference:

"Configuration metadata for outbound HTTP connections" on page 3008
You can add general proxy configuration parameters to the file `proxy-config.xml` by using `meta-data` settings.

Configuring security for overlay analytics reports:

You can administer which users can view overlay reports. To do this, you use the virtual resource `OVERLAY_REPORTS`.

About this task

This resource allows you to determine the user rights that are related to the overlay reports. The following description explains which roles a user requires to view overlay reports. The user actions correspond to the normal portal roles. Privileges are inherited.

In a default portal installation the `wpsadmins` group has the `USER` role. There is no role assigned to the anonymous user or to the group All authenticated users.

Notes:

- To view overlay reports, users need at least the USER role to the virtual resource OVERLAY_REPORTS , ADMIN_SLOTS , and to the resource on which they want to see the overlay report.
- In a default portal installation only the group wpsadmins has the USER role on OVERLAY_REPORTS.

USER Can view overlay reports on all resources, for example pages or portlets where at least the USER role is given.

Configuring a Credential Vault for overlay reports:

To access the IBM Coremetrics Web Analytics system, you have to store your user information in a Credential Vault slot. If you do not do this, the portal overlay reports cannot show data from the Coremetrics system.

About this task

You can set up the Credential Vault by one of the following three methods:

- Manually by using the Credential Vault portlet in the section for Access and Managing credential information of the portal administration. For details, refer to the portlet help about *Managing credential information* and *Adding a vault slot*.
- By using the XML configuration interface portlet. For details refer to the Import XML portlet help *Importing an XML configuration file*.
- By importing an XML configuration interface script. For details, refer to the topics about working with the portal XML configuration interface.

Also refer to *Portlet authentication*.

Make sure you store the following Coremetrics user information:

- The Coremetrics client ID
- The Coremetrics user name
- The Coremetrics authentication key.

Notes:

1. The Credential Vault slot used for storing the user information must have the slot name com.ibm.portal.asa.coremetrics.slot .
2. In the credential slot, you need to enter the Coremetrics user name and client ID, separated by a hash sign (#), either as a shared user ID in the portal Administration user interface, or as an external-id in the XML configuration interface script.

For example: If you have a Coremetrics user name Bob and a Coremetrics client ID 123456789, enter these values as "Bob#123456789" in the credential slot. Here is a sample XML configuraion interface script for creating required Credential Vault entries:

```
<request
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
   type="update" create-oids="true">
   <!-- Sample for updating the Credential Vault.
        This script creates a new segment containing one slot in the portal Credential Vault.
        The credentials (user ID and password pairs) that are stored in the vault cannot be
        accessed by using the XML configuration interface. You can only set the credentials
        by using the portal administration portlets for the Credential Vault. -->
   <portal action="locate">
      <credential-segment action="update" adapter-type="default"
```

```
                    name="com.ibm.portal.asa.coremetrics.segment" user-mapped="false">
        <description>Segment containing credentials for Coremetrics</description>
        <credential-slot action="update" name="com.ibm.portal.asa.coremetrics.slot"
                    active="false" system="true" resource="none"
                    secrettype="userid-password">
          <localedata locale="en">
            <description>Credentials for accessing the CoreMetrics server</description>
            <keywords>CoreMetrics</keywords>
          </localedata>
          <password-secret action="create" user="user_id"
            external-id="coremetrics_user_name#coremetrics_client_id">coremetrics_auth_key
          </password-secret>
        </credential-slot>
      </credential-segment>
   </portal>
</request>
```

In the sample XML script, replace the following variables with your actual Coremetrics user data values:

user_id
: Replace this variable with the user ID that is defined under the Resource Environment Provider **WP_VaultService** in the entry `systemcred.dn` .

coremetrics_user_name
: Replace this variable by your Coremetrics user name.

coremetrics_client_id
: Replace this by your Coremetrics client ID.

coremetrics_auth_key
: Replace this by your authentication key used by Coremetrics. This allows URL queries to the Coremetrics system.

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related tasks:

"Importing pages or page hierarchies by using the XML Import portlet" on page 1063
You can import an XML configuration file by using the XML Import portlet.

Related information:

Portlet authentication
For resources protected by the portal, IBM WebSphere Portal Express uses CORBA credentials and an encrypted LTPA cookie to authenticate users. However, for backend systems that require their own authentication, portlets need to provide some form of authentication to access these remote applications. To provide a single sign-on user experience, portlets must be able to store and retrieve user credentials for their particular associated application. Then, they use those credentials to log in on behalf of the user. WebSphere Portal Express supports the use of a credential vault where users and administrators can safely store credentials for authentication. Portlets that are written to extract the user's credentials from the vault can hide the login challenge from the user.

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Configuring the theme for overlay reports:

To integrate the overlay analytics features into your theme, add a theme module to the theme profiles of your modular theme. If you do not add the theme, the overlay menu items are not displayed.

Before you begin

Locate the **overlay analytics features** that are provided by the theme module `wp_analytics_overlay_reports`. This theme module contains a menu contribution, which defines the menu entries that display or hide the overlay reports in the portal user interface. In the default theme, these menu entries are integrated into the **Actions** menu of the theme.

About this task

The theme module for analytics overlay reports is not part of the theme profiles that are provided by the default theme. You need to explicitly add this module to one or multiple theme profiles.

Procedure

1. Open the menu definition file for the **Actions** menu. It is in WebDAV at `fs-type1/themes/Portal8.5/menuDefinitions/pageAction.json`
2. In the menu definition file, find the menu entry of type `ModuleRef`, which references the theme module `wp_analytics_overlay_reports`. For example:

```
{
        "type":"ModuleRef",
        "id":"wp_analytics_overlay_reports"
},
```

3. To add the theme module for analytics overlay to a deferred theme profile, add the module code to the deferred section of the profile in WebDAV to `fs-type1/themes/Portal8.5/profiles/profile_deferred.json`. In the following example, add the `deferredModuleIDs` code between the `ModuleIDs` and the `wp_analytics_overlay_reports` line.

```
{
   "moduleIDs": [
        ... modules that are loaded immediately...
        ...
   ],
   "deferredModuleIDs": [
       ... modules that are deferred...
       ...
       "wp_analytics_overlay_reports",
       ...
   ],
   ...
}
```

4. Restart your Portal server or invalidate the resource aggregator cache. To invalidate your cache, click **Theme Analyzer** > **Utilities** > **Control Center** > **Invalidate Cache**. CF06 Auto invalidation recognizes your changes automatically for WebDAV based themes. No further action is required. For more information, see Utilities.

What to do next

The theme module menu entries are displayed only if the theme module `wp_analytics_overlay_reports` is part of the theme profile. The theme profile is associated with the portal page that is being rendered.

If you have a custom modular theme, you can use the same approach to integrate the overlay features into a different menu.

Viewing overlay analytics statistics:

Users with the appropriate access rights can use Active Site Analytics to view graphical statistics about individual portal resources, such as pages or portlets.

About this task

To enable overlay statistics, the user opens the page actions menu and selects either **Show Portlet Reports** or **Show Page Reports**. The portal refreshes and shows boxes with the statistics coupled with the corresponding resource.

When the user clicks a report type in the actions menu, the **display reports** setting is persisted in the navigational state. This means that the user can now navigate through the portal and view statistics on every page that the user visits. To view more details about the resource or the statistics, the user clicks the **Details** link in the overlay box. The portal redirects the user to the website of the analytics business partner.

To disable overlay statistics, the user opens the page actions menu again and clicks the menu item **Close Analytics Reports**. This disables the display of overlay statistics and removes the setting from the navigational state.

Notes:

- Analytics providers might have a policy in place that limits the amount of concurrent API calls against their servers. Such a policy can affect the maximum number of overlay reports that can be displayed on a single portal page. In such cases work with the analytics provider to raise that limit.
- To be able to view graphical statistics about portal resources, the user needs to have the appropriate roles and access rights, both for viewing the reports and for the portal resources for which the reports are shown. The menu items for displaying statistics are only added to the page actions menu if the user has the access rights to work with analytics and overlay reports.
- Depending on the height of the portlet window on the portal page, the overlay window can display different graphics. For smaller portlet windows the graphic is smaller than for larger portlet windows.
- The option to view overlay reports is not available for pages that are currently being updated with new content. For the option to become available, save the page or draft of the page.

Customizing overlay reports:

You can customize your own overlay reports by setting specific parameters as required. Learn about the parameters and the levels at which you can specify them.

The scheme described here applies to all parameters that are available for overlay reports.

Report parameters for customizing overlay reports

To customize the display of overlay reports, you can provide the additional query parameters listed in the following. You can provide all these parameters at six different levels. The levels are listed and described in a table later in this topic. Depending on the level, you have to prefix the parameters with `asa.report` . Also depending on the level, you specify the parameters and their values in different locations and ways, for example as part of code with an equal sign, or by typing them into fields in a user interface.

metrics [page|portlet]

You can specify metrics for the page only, for the portlets only, or for a page and all portlets on it:

metrics
This affects both page and portlet metrics.

metrics.page
This affects only page metrics.

metrics.portlet
This affects only portlet metrics.

For these parameters, specify a comma separated list of metrics that you want to show in the overlay report. Do not include spaces between the items in the comma separated list. Specify the metrics by their column ID. You can use all metrics that are available in an IBM Coremetrics Web Analytics trend report. The default value is `SESSIONS,PAGE_VIEWS`. Note that the type of the overlay reports is optimized for displaying two metrics. Specifying fewer than one or more than two metrics may yield unexpected display results. The list of applicable metrics depends on the type of your Coremetrics account.

granularity
Specify the interval granularity of the displayed metrics. Specify an uppercase letter. The default value is `W` for weekly reports. Possible values are as follows:

D Daily

W Weekly. This is the default.

M Monthly

Q Quarterly

Y Yearly

numPeriods
Specify the number of periods that you want to have reported. This starts backwards from the value that you specify for the parameter `period_a`. Specify a numerical value larger than `1` . The default value is `4` .

period_a
Specify the ending date for reports in the format `YYYYMMDD` . Specify a valid date in the past. Reports will show data from this date backwards, with the number of periods as specified by the parameter `numPeriods` and the

granularity as specified by the parameter `granularity` . The default value is `yesterday` ; it is computed at the time of the HTTP request in the format `YYYYMMDD` .

Providing parameters at multiple levels

You can set the parameters for the customized display of overlay reports at six different levels as shown by the following table. Depending on the level, you have to prefix the parameters with `asa.report` , for example, `asa.report.granularity` .

Table 244. Customized display of overlay reports: levels for specifying parameters. Setting parameters for the customized display of overlay reports at multiple levels

Level hierarchy	Specify the parameter at this level:	Prefix the parameter with the following string:	Configure the parameter at this layer:	Set this parameter by using the following method:
1	HTTP request parameter	—	HTTP request	Custom code in the theme or skin
2	Portlet parameter - metadata	`asa.report.`	portlet window	XML configuration interface
3	Portlet preference	`asa.report.`	portlet	Portlet Edit mode, XML configuration interface
4	Page parameter - metadata	`asa.report.`	page	Page Properties dialog, XML configuration interface
5	Portal Configuration Service parameter	`asa.report.`	WP Config Service	WebSphere Integrated Solutions Console
6	Default value	—	WebSphere Portal Express product code	N/A - product code. You cannot change this default value, but you can override it in the layers listed previously.

If you provide parameters at multiple levels, the settings are applied in the hierarchical order of levels as given in the first table column Level hierarchy. Parameters specified at levels shown in earlier rows in table 1 override parameters shown in last table rows.

Customization examples

The following examples for customizing overlay reports show how the override precedence works between different levels in the hierarchy of levels. The examples use the granularity parameter. The precedence hierarchy works the same way for all other report parameters.

Example 1:

Scenario: No configuration provided.

Parameter settings:

Table 245. Example 1 parameter settings. No configuration provided.

Level hierarchy	Level	Interval granularity key	Interval granularity value
1	Request parameter	—	—
2	Portlet parameter - metadata	—	—
3	Portlet preference	—	—
4	Page parameter - metadata	—	—
5	Configuration parameter	—	—
6	Default value	`granularity`	`W` (weekly)

Result: All defaults apply. All overlay reports will show sessions and page views in weekly interval granularity for four weeks, starting backwards from yesterday.

Example 2:

Scenario: Interval granularity is overridden on the page level.

Parameter settings:

Table 246. Example 2 parameter settings. Interval granularity is overridden on the page level.

Level hierarchy	Level	Interval granularity key	Interval granularity value
1	Request parameter	—	—
2	Portlet parameter - metadata	—	—
3	Portlet preference	—	—
4	Page parameter - metadata	`asa.report.granularity`	`M` (monthly)
5	Configuration parameter	—	—
6	Default value	`granularity`	`W` (weekly)

Result: As the granularity setting at the higher level overrules the one at the subordinate level, overlay reports at the page where `asa.report.granularity` is set show sessions and page views with monthly granularity for four weeks, starting backwards from yesterday. Reports on all other pages will show as specified by the default configuration.

Example 3:

Scenario: Interval granularity is overridden at request level.

Parameter settings:

Table 247. Example 3 parameter settings. Interval granularity is overridden at request level.

Level hierarchy	Level	Interval granularity key	Interval granularity value
1	Request parameter	`granularity`	`W` (weekly)
2	Portlet parameter - metadata	—	—
3	Portlet preference	—	—
4	Page parameter - metadata	`asa.report.granularity`	`M` (monthly)
5	Configuration parameter	—	—
6	Default value	`granularity`	`W` (weekly)

Result: As the request parameter has the highest precedence, overlay reports will show sessions and page views with weekly interval granularity for four weeks, starting backwards from yesterday. Reports on all other pages will show as specified by the default configuration.

Analytics tags and site promotions

To obtain further analytics information from your portal, you can use analytics tags for your portal resources. You can also use analytics tags for site promotions.

"Analytics tags" on page 1725
You can obtain specific analytics information from your portal by using analytics tags for your portal resources. You might want to know which types of users visit your portal site most frequently, for example by age group or

other characteristics. If you want to address mostly young professionals, you can create an analytics tag `targetAudience:youngProfessionals` ; you can then associate the tag with all resources that contain content for this target user group. By looking at the visits on these pages you can determine whether the main user group of your portal site is young and new in their job.

"Site promotions" on page 1727
You can use analytics tags to set up site promotions. A site promotion is part of a marketing campaign that has the objective to introduce a new product, service, or event, or better position an existing one. A marketing campaign can consist of mailings, posters, banners, articles, games or other web content. In IBM WebSphere Portal Express, a site promotion covers the web content part of the marketing campaign. For example, a site promotion named "Christmas 2011" can reference pictures, blogs, and gift-shops on a website.

"Security for analytics tags and site promotions" on page 1729
Security for analytics tags allows you to administer which users can view, create, or assign analytic tags, including site promotions. The portal provides a virtual resource for site promotions and roles on this virtual resource.

"Using the XML configuration interface to administer analytics tags" on page 1729
You can use the XML configuration interface to manage analytics tags and site promotions in the portal.

Analytics tags:

You can obtain specific analytics information from your portal by using analytics tags for your portal resources. You might want to know which types of users visit your portal site most frequently, for example by age group or other characteristics. If you want to address mostly young professionals, you can create an analytics tag `targetAudience:youngProfessionals` ; you can then associate the tag with all resources that contain content for this target user group. By looking at the visits on these pages you can determine whether the main user group of your portal site is young and new in their job.

An analytics tag in the portal consists of a name-value pair that you can associate with portal or Web Content Manager resources. When you add a resource to an analytics tag, a microformat is injected into the markup of the resource. Analytics provider applications can read this microformat and track which resources associated with the analytics tag users visited.

The association of a resource to an analytics tag is called an analytics tag mapping. Resources that can be associated to an analytics tag include portal pages, portlets, and Web Content Manager content items. The relation between a resource and an analytics tag is as follows:

- One resource can be added to several general analytics tags, but it can be added only once to each analytics tag.
- One analytics tag can contain several resources but only one of each resource.

To associate a resource with an analytics tag, you add the tag name and value to the resource. For example, the tag name can be `season promotion` , and values can be `christmas` or `easter` . This way you can use the analytics tags to group your portal resources by purpose and then analyze which user groups visit these resources most frequently.

The portal stores analytics tags as regular portal tagging and rating tags, but with the following specific naming convention for analytics tags only:

```
name#value
```

where the tagging and rating tag is made up by a combination of the `name` and `value` of the analytics tag, separated by a hash sign character (`#`). Tags that are encoded like this show in the Tag Center under the tab **Analytics**.

"Working with analytics tags"
To work with analytics tags, the portal provides a dialog. You can create new analytics tag mappings and view and delete existing ones.

Working with analytics tags:

To work with analytics tags, the portal provides a dialog. You can create new analytics tag mappings and view and delete existing ones.

About this task

You can open this dialog from the resource that you want to associate with an analytics tag.

Procedure

1. To **add** a page or portlet to an analytics tag, proceed as follows:
   a. Open the page **Actions** menu or the portlet menu. The menu provides a section for Analytics.
   b. Select the menu entry **Analytics Tags**. The analytics tag dialog opens; it offers the following controls:
      - Two fields for entering the **Analytics tag name** of the tag to which you want to add the resource, and the **Analytics tag value** for that tag. Your input is matched against a regular expression.

        Notes:

        1) Special characters and all of the following characters are not valid: parentheses, brackets, angle brackets, hash signs, single and double quotes: `( , ) , [ , ] , < , > , # , ' , " .`
        2) The maximum allowed length for a name or a value is 100 characters.
        3) You cannot associate the same analytics tag with the same resource twice.
      - A list of existing analytics tag mappings, that is analytics tags that are associated with the portal resource. They have Delete icons next to them. You can view these icons and delete mappings only if you have the appropriate access rights. For details, see the topic about security for analytics tags.
      - An **Add** button. To save the mapping between the entered analytics tag and the currently viewed portal resource, click **Add**.
      - A **Done** button. Click **Done** when you have completed working with analytics tags for the currently viewed resource.
   c. To add a new analytics tag to the resource, type the analytics tag in the tag name field. The field provides a typeahead feature. After you type three characters, it lists the existing tags that start with these three characters.
   d. Type the value for the analytics tag in the second field. You must fill both fields.

e. To add and save the analytics tag mapping, click **Add**. The portal saves your tag mapping. If this mapping was not used before, it is created new. The tag and its value are shown in the dialog and added to the markup immediately .

2. To **delete** an existing tag mapping, proceed as follows:
   a. Open the page **Actions** menu or the portlet menu. The menu provides a section for Analytics.
   b. Select the menu entry **Analytics Tags**.
   c. Click the **Delete** icon for the tag mapping that you want to delete.
   d. Click **Done**. The portal removes the tag mapping from the resource.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Site promotions:

You can use analytics tags to set up site promotions. A site promotion is part of a marketing campaign that has the objective to introduce a new product, service, or event, or better position an existing one. A marketing campaign can consist of mailings, posters, banners, articles, games or other web content. In IBM WebSphere Portal Express, a site promotion covers the web content part of the marketing campaign. For example, a site promotion named "Christmas 2011" can reference pictures, blogs, and gift-shops on a website.

In WebSphere Portal Express, a site promotion can be associated with WebSphere Portal Express or IBM Web Content Manager resources. If a resource is assigned to a site promotion, a microformat is injected into the markup of the resource. This microformat can be picked up by an aggregator of the analytics provider. The analytics provider is able to track which site promotion resources were visited or which links were most often followed to get to the campaign content.

The assignment of a resource to a site promotion is called a site promotion mapping. Resources that can be assigned to a site promotion include portal pages, portlets, and Web Content Manager content items. The relation between a resource and a site promotion is as follows:

- One resource can be assigned to several site promotions, but can be assigned only once to each site promotion.
- One site promotion can contain several resources but only one of each resource.

For information about the security of site promotions, refer to the topic about Security for analytics tags and site promotions.

"Working with site promotions"
To work with site promotions, the portal provides a dialog. You can create new site promotion mappings and view and delete existing ones.

Working with site promotions:

To work with site promotions, the portal provides a dialog. You can create new site promotion mappings and view and delete existing ones.

About this task

You can open this dialog from the resource that you want to associate with a site promotion.

Procedure

1. To **add** a page or portlet to a site promotion, proceed as follows:
   a. Open the page **Actions** menu or the portlet menu. The menu provides a section for Analytics.
   b. Select the menu entry **Site promotions**. The site promotion dialog opens; it offers the following controls:
      - A field for entering the **Site promotion name** of the site promotion to which you want to add the resource. Your input is matched against a regular expression.

        Notes:

        1) Special characters and all of the following characters are not valid: parentheses, brackets, angle brackets, hash signs, single and double quotation marks: `( , ) , [ , ] , < , > , # , ' , " .`
        2) The maximum allowed length for a site promotion name is 100 characters.
        3) You cannot associate the same site promotion with the same resource twice.
      - A list of existing site promotion mappings, that is site promotions that are associated with the portal resource. They have Delete icons next to them. You can view these icons and delete mappings only if you have the appropriate access rights. For details, see the topic about security for site promotions.
      - An **Add** button. To save the mapping between the entered site promotion and the currently viewed portal resource, click **Add**.
      - A **Done** button. Click **Done** when you complete working with site promotions for the currently viewed resource.
   c. To add a new site promotion to the resource, type the site promotion name in the site promotion name field. The field provides a type-ahead feature. After you type 3 characters, it suggests existing site promotions that start with these 3 characters.
   d. To add and save the site promotion mapping, click **Add**. The portal saves your site promotion mapping. If this mapping was not used before, it is created new. The site promotion is shown in the dialog and added to the markup immediately.
2. To **delete** an existing site promotion mapping, proceed as follows:
   a. Open the page **Actions** menu or the portlet menu. The menu provides a section for Analytics.
   b. Select the menu entry **Site promotions**.
   c. Click the **Delete** icon for the site promotion mapping that you want to delete.
   d. Click **Done**. The portal removes the site promotion mapping from the resource.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Security for analytics tags and site promotions:

Security for analytics tags allows you to administer which users can view, create, or assign analytic tags, including site promotions. The portal provides a virtual resource for site promotions and roles on this virtual resource.

The portal provides the virtual resource **ANALYTICS_TAGS** to administer which users can manage analytics tags. This resource allows you to determine user rights that are related to analytics tags. The following list explains which roles users need in order to manage analytics tags. The user actions correspond to the usual portal roles. Privileges are inherited.

In a default portal installation the group **wpsadmins** has the role **EDITOR**. There is no role assigned to the anonymous user or to the group **All authenticated users**. Note that all users need to have at least the **EDITOR** role on the virtual resource **ANALYTICS_TAGS** and the **USER** role on the resource to be able to edit analytics tags assignments.

USER
- Can view existing analytics tag mappings.
- Can view existing analytics tag assignments on all resources, for example pages or portlets, on which the user has at least the **USER** role.

EDITOR
- Includes all **USER** actions.
- Can create analytics tags and analytics tag mappings.
- Can add and remove analytics tag mappings on all resources, for example pages or portlets, on which the user has at least the **USER** role.

MANAGER
- Includes all **USER** and **EDITOR** actions.
- Can delete analytics tags.

Note: In a default portal installation only the group **wpsadmins** has the **MANAGER** role on **ANALYTICS_TAGS**.

Using the XML configuration interface to administer analytics tags:

You can use the XML configuration interface to manage analytics tags and site promotions in the portal.

The XML resource related to analytics tags is `tag` . The attributes are listed in the following.

Notes:
1. When you create tags, you need to specify all attributes except the attribute `locale` marked as optional in the attributes list.

2. When you export tags, the export result file will contain all the tags that exist in your portal system, including the tags created for tagging and rating.
3. Tag names of analytics tags must start with `com.ibm.portal.asa.` and must match the following pattern: `com.ibm.portal.asa.`*`tag_name#tag_value`* .
4. The tag name for site promotions must be `com.ibm.portal.asa.SitePromotion` .
5. You must escape all special characters. This includes the blank space.

tag This is the XML resource tag for analytics tags. Use the following attributes with this tag:

resourceref = "object_ID"
Use this attribute to specify the reference to the resource that is being tagged.

domain = "comm"
For analytics tags, specify a value of `"comm"` for this attribute. For tagging and rating this attribute is used to specify the database domain for the tagged resource. The only accepted value for analytics tags is `"comm"` .

owner = "*user*"
Use this attribute to specify the owner of the tag.

locale = "*locale*"
Use this attribute to specify the locale of the tag. This attribute is optional. If you do not specify this attribute, it defaults to `null` .

For examples refer to the following code samples:

Example: Exporting analytics tags

```
<?xml version="1.0" encoding="UTF-8"?>
<request
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
   type="export">
   <!-- This sample exports all tags.
        In addition to all analytics tags (starting with 'com.ibm.portal.asa.'),
        all tags (see tagging and rating) are exported.
   -->
   <portal action="locate">
      <tag action="export" objectid="*"/>
      <!-- Export all tags with a specific locale in the system -->
      <!-- <tag action="export" objectid="*" locale="SPECIFIC_LOCALE"/> -->
   </portal>
</request>
```

Example: Creating analytics tags

```
<?xml version="1.0" encoding="UTF-8"?>
<request
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
   type="update">
   <!-- This sample creates analytics tags and site compositions.
        NOTE: This sample file needs to be modified before execution.
        Update the value of the 'owner' attributes of the 'access-control',
        and 'tag' tags, and specify an existing user.
   -->
   <portal action="locate">
      <!-- Locate the page. -->
      <content-node action="locate" objectid="gettingStartedPage"
                    uniquename="ibm.portal.Home.Getting Started"/>
      <!--  NOTE:
            Pattern:  <tag-name>#<tag-value>
```

```
            Special characters (including 'blank space') must be escaped!
            All tag names must start with 'com.ibm.portal.asa.'
            Tag name for site promotions must be 'com.ibm.portal.asa.SitePromotion'
      -->
      <!-- Assignment of the site promotion "Christmas 2011" to the gettingStartedPage -->
      <tag action="update" objectid="ZCI_B1L68B1A00DO80IG7PCV0I6000"
           resourceref="gettingStartedPage" domain="comm"
           owner="uid=wpsadmin,o=defaultwimfilebasedrealm"
           locale="en">com.ibm.portal.asa.SitePromotion#Christmas%202011</tag>
      <!-- Assignment of the analytics tag 'Target Audience#Young Professionals'
           to the gettingStartedPage -->
      <tag action="update" objectid="ZCI_B1L68B1A00DO80IG7PCV0I7000"
           resourceref="gettingStartedPage" domain="comm"
           owner="uid=wpsadmin,o=defaultwimfilebasedrealm"
           locale="en">com.ibm.portal.asa.Target%20Audience#Young%20Professionals</tag>
   </portal>
</request>
```

Example: Deleting analytics tags

```
<?xml version="1.0" encoding="UTF-8"?>
<request
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
   type="update">
   <!-- This sample deletes analytics tags.
        NOTE: This sample assumes that the CreateAnalyticsTags.xml
        sample was executed before.
   -->
   <portal action="locate">
      <!-- Delete the analytics tags created by sample CreateAnalyticsTags.xml -->
      <tag action="delete" objectid="ZCI_B1L68B1A00DO80IG7PCV0I6000"/>
      <tag action="delete" objectid="ZCI_B1L68B1A00DO80IG7PCV0I7000"/>
      <!-- Delete all tags with a specific locale in the system -->
      <!-- <tag action="delete" objectid="*" locale="SPECIFIC_LOCALE"/> -->
  </portal>
</request>
```

Related reference:

"Types of portal resources" on page 1085
The portal resources are represented by the following XML tags.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Auditing

IBM WebSphere Portal Express includes an auditing function that allows users to log certain events and their originators in to a separate log file. This file can then be used to track administrative activities.

The following details are logged for each event:

- The time stamp
- An optional transaction ID
- An optional project ID
- The user action
- Individual event details

If the user who does the action (for example, Bob) is impersonated by another user (for example, Alice), the user is shown as [Bob[Alice]] in the log file.

You can use the auditing function on the following events:

- Creating, modifying, and deleting users and groups
- Assigning and revoking roles to and from users
- Modifying role blocks
- Modifying resource ownership
- Creating, modifying, and deleting protected resources
- Externalizing and internalizing resources
- Installing and uninstalling web modules
- Creating and deleting application roles
- Assigning and revoking application roles to and from users
- Adding and deleting roles to application roles
- Initializing a database domain
- Creating, modifying, and deleting portlet applications by using IBM Lotus Component Designer
- Starting and ending impersonating a user or impersonating a user without the appropriate permission.

To activate and configure the auditing function, modify the auditing service settings in the Auditing Service by following the steps that are provided in the Setting service configuration properties file.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Chapter 14. Setting up a website

Setting up a website includes, creating pages, adding navigation, setting up search, and adding content to the site. Themes are used to customize the portal's look-and-feel. Out-of-the-box templates and the site wizard can help you set up your portal site faster. You can add wikis and blogs to your site and let users tag and rate content on your site.

"Social rendering" on page 2106
IBM WebSphere Portal Express page editors can use social rendering to feature social data that is hosted on a remote IBM Connections server in the context of portal pages.

"Setting up marketing campaigns" on page 2240
Use these tools and features to setup marketing campaigns for your website.

"IBM Web Content Manager Multilingual Solution" on page 2442
The Web Content Manager Multilingual Solution is a set of tools that are used to manage translated versions of localized and regionalized websites.

"IBM Syndicated Feed Portlet for WebSphere Portal Express" on page 2471
The new IBM Syndicated Feed Portlet for IBM WebSphere Portal Express offers enhanced feed subscription and presentation capabilities.

Website building blocks

Unless you are familiar with the IBM WebSphere Portal Express and IBM Web Content Manager website model, review the building blocks. Building blocks are the different parts that make up the website framework, such as pages, navigation, themes, content, content libraries, and more.

"Themes, profiles, and skins"
The theme controls the presentation layer of your website. The theme profile contains modules and controls which modules load per page. A skin surrounds the portlet on a page.

"Pages" on page 1736
A page is an organization element that contains portlets. Since version 8.0, pages are stored in the Portal Site web content library and managed by IBM Web Content Manager. Pages are stored as content in the library. By managing portal pages from within Web Content Manager, you can apply features like workflow, version control, and syndication to portal pages.

"Portlets" on page 1744
Portlets are small applications that are independently developed, deployed, managed, and displayed. After the portlet is deployed, you can use it multiple times on different pages.

"Content" on page 1746
Rich content and elegant user experiences drive the success of websites. Each digital experience consists of many different types of content from many various sources.

Themes, profiles, and skins

The theme controls the presentation layer of your website. The theme profile contains modules and controls which modules load per page. A skin surrounds the portlet on a page.

To some extent, the theme governs aspects of the entire website. In the following wireframe, the theme controls the navigation placement, the styles and branding, the visual treatment of portlets on the page, the JavaScript modules that are available, and more.

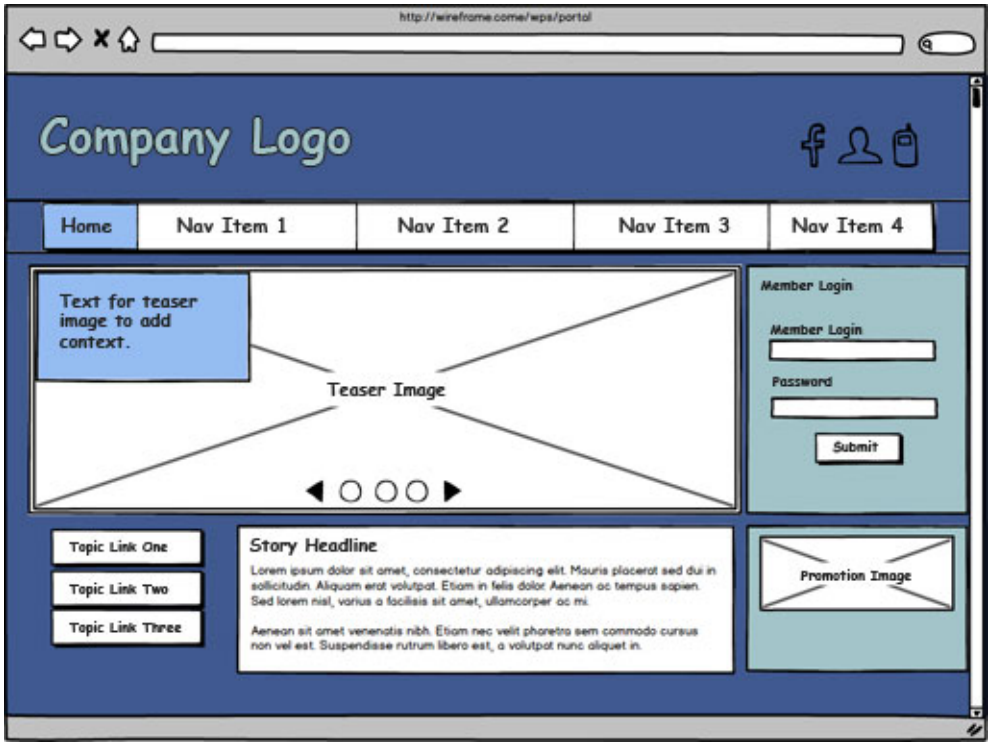


Figure 17. Wireframe of website

Themes

In addition to branding, themes provide a wide range of capability. The theme includes page layouts, styles, and modules. The page layouts and styles are visible to content authors from the site toolbar. Modules within a theme are grouped into profiles. Profiles are associated with pages to ensure each page loads the appropriate modules.

The default theme, Portal 8.5, is designed using Responsive Web Design principles. Therefore, the site renders well on mobile devices. You can make a copy of this theme and customize it.

Theme profiles

Use profiles to load only the modules that are required for a page. If a portlet require specific extra modules, you can group those modules into a profile. Then, assign the profile to the page that contains the portlet. For example, the Web Dock Portlet requires extra modules to render third-party content in an iFrame. Those modules are grouped into a profile called Web Dock profile. Only the pages that contain Web Dock Portlets need to load the additional modules. Assign the Web Dock profile to pages that contain the Web Dock Portlet. Pages that do not contain a Web Dock Portlet do not need to waste resources loading extra modules.

Note: CF03 Starting with CF03, the Web Dock profile no longer exists. If you are using the Resource Aggregator for Portlets, no additional steps are necessary. If you are not using the Resource Aggregator for Portlets, add the **wp_webdock** module to an existing profile on your page. You do not need any extra profiles when module autoLoad is turned on.

Skins

Skins include more capability and can be used to visually indicate portlets on a page. However, depending on your website design that might not be wanted. The skin that is immediately available and active is a hidden skin. On the rendered website, the skin is not visible. However, the hidden skin is visible when Edit Mode is on. The outline of each portlet helps the author identify the portlets on the page. The skin also includes capability for inline editing.

"JavaScript libraries"
WebSphere Portal Express includes JavaScript libraries to support various features.

Related concepts:

"Included profiles" on page 2606
Portal includes profiles that contain modules that change how your portal site is rendered.

"Modules that are provided with the modularized theme" on page 2608
WebSphere Portal Express provides a set of ready-to-use modules.

"Default skins" on page 2687
The default theme includes these skins that can be used as a basis for creating custom skins: Hidden, Standard, and NoSkin.

"Customizing the theme" on page 2658
The module framework allows themes to be customized in order to provide flexibility, enhance the user experience, and maximize performance.

JavaScript libraries

WebSphere Portal Express includes JavaScript libraries to support various features.

Dojo

WebSphere Portal Express contains an instance of the IBM Dojo Toolkit, a JavaScript library that is based on the Dojo toolkit (http://dojotoolkit.org). All Dojo packages can be used by IBM and non-IBM components.

jQuery

WebSphere Portal Express includes a jQuery module for the jQuery core so you can use jQuery in a theme.

Java messaging services for web content

Web Content Manager provides support for the notification of events such as item state changes, or services starting and stopping. These notifications can be delivered as messages to the Java messaging service.

Pages

A page is an organization element that contains portlets. Since version 8.0, pages are stored in the Portal Site web content library and managed by IBM Web Content Manager. Pages are stored as content in the library. By managing portal pages from within Web Content Manager, you can apply features like workflow, version control, and syndication to portal pages.

In the example wireframe, the home page portion is highlighted. The page contains five portlets. Most of the portlets render content by using the Web Content Viewer portlet. The Member Login portlet is providing access to the LDAP user registry

for member authentication.

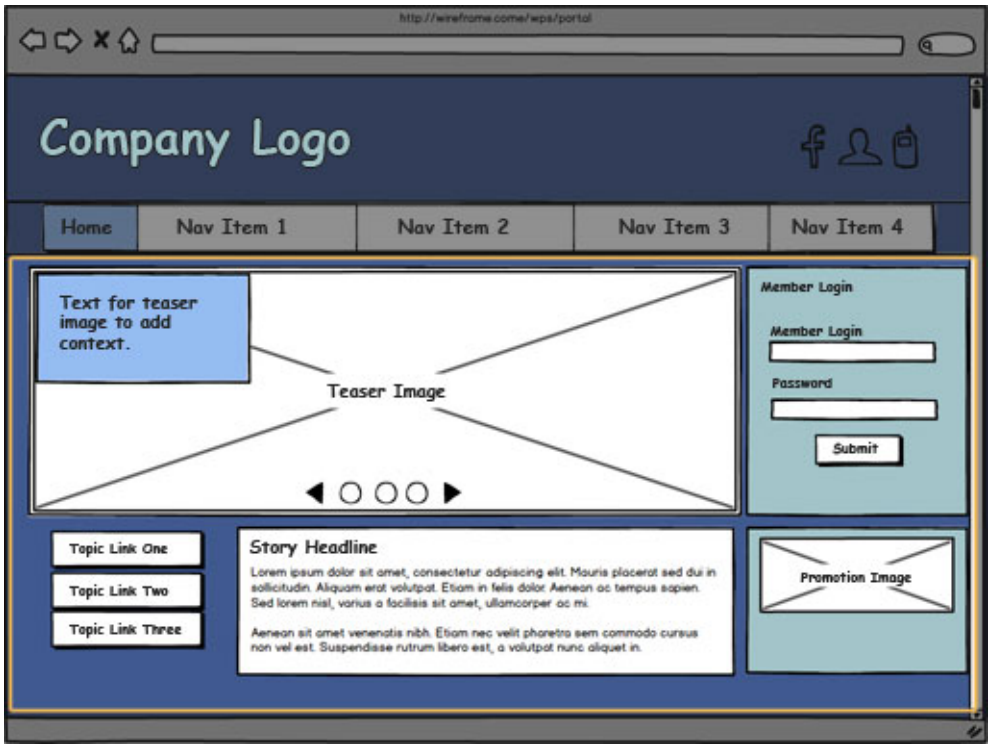


Figure 18. Wireframe with a page highlighted.

Pages and navigation

You can use the page hierarchy to establish the navigation structure in your website. The underlying information architecture is revealed in the navigation. There are other navigation approaches, but the page hierarchy is the simplest.

Types of pages

Page types include static, dynamic, derived, private, and hidden.

Static pages
Static pages use HTML templates to organize layout containers and controls, and are used as the default page type since version 7.0.

Dynamic pages
Dynamically created based on the definition of an existing page. In most cases, a dynamic page is a transient copy of a template page, often referred to as base page. This transient copy behaves like a snapshot of the base page from the time when the copy was created. It contains all portlets of the base page and all its properties. The benefit of using dynamic pages instead of static pages is that you can create multiple copies or instances of the base page. A user can then manually switch or be redirected between these instances.

Derived pages
Derived pages are child pages of the original page and inherit the properties of the original page. Creating a derived page is equivalent to creating a new, specialized layer on the original page. The original page and the new layer are aggregated together at rendering time. The new

layer is contained within and controlled by the original page. Reference an existing page to give administrative access to the other users, but still maintain the content and layout from the original page.

Private pages
A private page can be accessed only by its owner, who must be a Privileged User. A Privileged User can explicitly create new private pages that are accessible only by themselves. Additionally, a Privileged User on a non-private page can personalize the page and create new private pages underneath it. Customizing a non-private page usually creates a private copy of the corresponding non-private page. Any changes that a Privileged User makes to a non-private page are not accessible by other users.

Hidden pages
Hidden pages do not show in the portal, but contain portlets that can be opened from other pages. These hidden pages do not appear in the site navigation, but are opened from generated links in portlets or theme code. For ease of administration and conserving system resources, you can place and manage such pages in one place, for example, in the Edit Page Properties portlet: users can open from a link in the theme, but the portlet instance itself is on a hidden page in the content model.

Page templates are stored as hidden pages under the context root, **Context Root** > **Hidden Pages** > **Page Templates**.

Pages and the portal theme

Using the portal theme that is available for immediate use, you can

- Create, reorder, delete, and edit the properties of pages, labels, and URLs
- Reorder pages, labels, and URLs
- Assign access to pages, labels, and URLs
- Move pages to a new location in the portal hierarchy

Both administrators and users with appropriate access can create and delete pages. Users can delete only the pages they create or the pages for which they have at least Manager access.

Versioning and syndication of pages

You can create version and use syndication to publish the following portal artifacts:

- Portal page definitions that are stored in the release domain, including page properties, metadata, and layout settings
- Content targeting rules
- Access control that you grant to portal pages
- Public wires that connect portlets on the same page or on different pages
- Portlet preferences that are made in the **Edit Shared Settings** mode
- Vanity URLs

You cannot create version or use syndication to publish the following portal artifacts:

- Explicitly derived pages
- Private pages
- Personalization rules that are defined in Personalization and not in Web Content Manager

- Tags and ratings
- Themes and skins
- URL mappings
- Artifacts that are stored in the WebDAV file store by portlets or iWidgets
- WSRP Producers

Pages and the authoring portlet

By default, the display name of the portal page site area is based on the title of the portal page. Web Content Manager assigns a unique name in the library for each portal page site area. You can have pages with the same title organized in separate portal page site areas.

The operations that you can do on the portal page site areas in the authoring portlet are restricted. You can set Web Content Manager properties in the authoring portlet, and you can also move portal pages through the authoring portlet. But you cannot change the name, title, or description of Portal pages. For changing the title or description, you must use the portal user interface.

Note: If you delete a portal page, the portal page site area is deleted from the web content library. If the portal page site area contains any other site areas or content items, they are also deleted.

Pages and portal page site areas

The portal page site area is a special site area. In addition, to the usual Web Content Manager data site area pages contain an XML document that represents the portal page in the portal release domain database. This XML document is updated whenever a new version of the page is created or during a JCR import.

"System content associations"
System content associations are used to associate a portal page with its corresponding artifacts in IBM Web Content Manager. A system content association is an extension of the standard content association.

"Best practices for pages" on page 1740
Use these tips and guidelines to develop and deploy pages more effectively.

"Troubleshooting pages" on page 1741
When you work with pages, you might encounter problems that are related to projects, access rights, or other issues.

"Known issues for pages" on page 1743
You can review known issues for pages.

System content associations

System content associations are used to associate a portal page with its corresponding artifacts in IBM Web Content Manager. A system content association is an extension of the standard content association.

A standard content association maps a web content page or web content viewer to content in a web content library. For a system content association, there is an additional `system` flag that distinguishes the association from a content association. Like web content associations, system content associations point to objects in a web content library. However, the objects are associated with a page rather than content. System content associations are managed by the portal. The system content association acts as an automatic default content mapping of the portal page to the portal page site area and allows to automatically find content underneath

that portal page site area. The content mappings can be used in addition to the system mapping and define extra content contributions to the page and it is also possible to change the default from using the system mapping to another content mapping.

The `system` flag is a private, read-only flag. You can use the public API or the REST API to query a content association to determine whether it is a system content association or standard content association. You cannot use these interfaces to modify the `system` flag. To modify `system` data, use the XML configuration interface (XML Access).

Best practices for pages

Use these tips and guidelines to develop and deploy pages more effectively.

Create links in web content to portal pages

When you enable pages, you can create links to portal pages from within the authoring portlet in IBM Web Content Manager. You can create links to portal pages in two ways:

- By editing a content item in the rich text editor and inserting a link.
- By creating a link component.

To select the portal page, click **Browse content** in the **Link** field, and browse to the page in the Portal Site library.

Referential integrity applies for links to portal pages. You cannot delete a portal page if a link pointing to that page exists. You can view or remove such link references in the following ways:

- Edit the page properties in the portal user interface and select **View References**.
- Select the page item in the authoring portlet in Web Content Manager and click **More** > **View References**.

When users click a link, the link is resolved according to the system content mapping for the portal page item in the Portal Site library. Based on the system content mapping, the appropriate portal page is displayed.

Important: You cannot change system content mappings through typical operations with the user interface. However, it is possible to change system content mappings through programmatic interfaces, like the XML configuration interface (**`xmlaccess`** command), or other low-level database operations. If a system content mapping is changed or corrupted through such a method, the link can no longer be resolved.

Use unique friendly URLs with pages

When you create pages, it is not possible to programmatically enforce uniqueness of friendly URLs in all circumstances. Because of this behavior, it is possible to create multiple pages that have the same friendly URL, which can produce unexpected results. To prevent potential confusion, ensure that all friendly URLs that you create are unique.

Use transaction processing with the XML configuration interface

Because pages are stored in the Portal Site library in Web Content Manager, each page has corresponding objects in the JCR database. You must be aware of this relation when you create, update, or delete pages with the XML configuration

interface. If **xmlaccess** processing is interrupted, it can result in a mismatch between the page state and database state.

WARNING: If you redeploy your site daily, your JCR size increases because of page versions. Periodically clean up your versions to reduce the JCR size. Go to "Clearing version history" on page 1194 for information.

To ensure that page and database information for a page remain synchronized, use the `transaction-level` attribute of the `request` element in the XML file. For more information about using the `transaction-level` attribute, go to *XML configuration reference.*

Example:

```
<request
    type="update"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
    transaction-level="resource">
```

Troubleshooting pages

When you work with pages, you might encounter problems that are related to projects, access rights, or other issues.

User *A* cannot view project *X*

Ensure that User *A* has the following access rights:

- User on the project. Specify this access by editing the project and adding User *A* to the **User** list in the **Access** section.
- User on the WCM_REST_SERVICE virtual resource. Specify this access in the portal.Click the **Administration menu** icon. Then, click **Access** > **Resource Permissions**. From the list of Resource Types, select **Virtual Resources** by clicking it. On the Resource Permissions page, click on the **Assign Access** icon for the WCM_REST_SERVICE resource. Then, click on the **Edit Role** icon to specify the access for the user.

User *A* cannot modify a published or draft page

Check the following issues as potential causes:

- If the page is part of a workflow, ensure that User *A* has Editor access to the current workflow stage.
- Is access control inheritance enabled for the portal page site area? The setting is enabled by default but can be disabled as needed. Verify the setting by editing the portal page site area and viewing the **Access** section of the properties to determine whether the **Inheritance** setting is selected.

  If access control inheritance is disabled, ensure that User *A* has Editor access to the portal page site area.

Why does User *A* receive the message "You are customizing this page. Changes are only visible to you"?

This message is generated because User *A* has Privileged User access to the page. This access level is the default access for a newly created user, and changes that the user makes are visible only to this user.

If you want the changes that User *A* makes to be visible to all users of that page, User *A* requires Editor access on the page.

User *A* is in a project and receives a message rather than creating a draft

The message "You are customizing this page. Changes are only visible to you" is generated because User *A* has Privileged User access to the page. This access level is the default access for a newly created user, and changes that the user makes are visible only to this user.

To create a draft in a project, User *A* requires either of the following access:

- Editor or higher rights on the page.
- User access on the page and Draft creator or higher access on the corresponding web content page item.

User *A* cannot drag content from the site toolbar onto the page

When the user uses the site toolbar and attempts to add content from the **Content** category in the toolbar, the following access is required for User *A*:

- Editor rights on the page.
- User rights on the web content viewer portlet that User *A* wants to add.

Portal pages are not synchronized with portal page site areas in Portal Site library

Typically, the pages in the portal and their corresponding portal page site areas in the Portal Site library are automatically synchronized. However, in some cases, these artifacts can become unsynchronized. For example, this situation can occur when data is restored from a backup or from errors when the portal page site area is created after the portal page is created.

You can resynchronize the Portal Site library based on the current portal page structure that is stored in the portal database. When you do the synchronization, the portal database acts a master repository. Any portal page site areas in the Portal Site library that do not correspond with existing portal pages are removed from the Portal Site library. Any content site areas or content items within the affected portal page site areas are also moved to the lost and found section of the Portal Site library.

To do the resynchronization, run the `create-page-nodes` configuration task, as described in *Enabling managed pages*.

Important: This task also removes any draft pages that are not found in the Portal Site library.

Related tasks:

"Enabling vanity URL support" on page 2100
In a new WebSphere Portal Express Version 8.5 installation, vanity URL support is enabled. If you upgrade your WebSphere Portal Express from a previous version to Version 8.5, vanity URL support is disabled. You can enable and disable vanity URL support as required by using a portal configuration task.

"Synchronizing the vanity URL database" on page 2104
Vanity URLs are stored as part of the page in the JCR database in the portal page site area of Web Content Manager. For performance reasons, the data is also stored in the WebSphere Portal Express database. When the data is modified, the portal synchronizes the data between both sides. However, under certain circumstances it can happen that the data is not synchronized. For such cases, the portal provides a

configuration task that synchronizes the data.

Known issues for pages

You can review known issues for pages.

Remote portlet entities are not aware of projects

`PortletEntity` objects that are stored in the Release domain of the portal database are project-aware. Changes to such a portlet entity, such as setting and modifying preferences, are reflected as a change limited to the active project. Changes to a portlet made in the active project are not visible on the published site until the changes are syndicated.

This ability to modify a portlet in a project does not apply to remote portlets that are produced with WSRP. As the remote system, the WSRP Producer is responsible for managing the portlet entities. However, because the WSRP Producer is not project-aware, the Producer cannot differentiate between the following changes:

- Changes made to the remote portlet entity directly on the published site.
- Changes made to the remote portlet entity when viewing and interacting with a project.

Because of this limitation, changes made in a project are displayed immediately on the published site through the remote portlet entity.

Authoring portlet issues

Search results for page items in the authoring portlet
When you search in the authoring portlet with the **Titles** or **Descriptions** filter, no results are returned for page items.

Sorting order of pages
When displayed in the authoring portlet, pages are listed according to the page IDs rather than the page titles. If you attempt to sort pages by title, this behavior causes the pages to display in an unexpected order. This sorting behavior also applies to any window or view, such as the Manage Project window, that is based on the authoring portlet.

Changing the order of pages in the portal

You can modify the ordering and hierarchy of pages in the portal only with the site toolbar in the portal interface. If you move the portal page site area in the authoring portlet, the page order in the portal is not affected.

Personalization rules

- Personalization rules that you create by using the Personalization editor are not managed in Web Content Manager and so are not available for versioning or included in syndication. These rules must be published by using the **pznload** command or by publishing with Personalization.
- Personalization rules are not aware of projects and the status of items in a project. Because of this characteristic, rules operate only on published content and do not include draft items.

Limited support for derived pages

Explicitly derived pages that are in the release domain can be managed by Web Content Manager. However, if you modify an explicitly derived page, that change does not generate drafts for all of the derived pages. If you want the change to

occur for all of the derived pages, you must edit each derived page separately. For more information, see the documentation about derived pages.

Portlet configuration settings

Configure mode

If you change the configuration settings for a portlet in **Configure** mode, these changes are global and are not limited to the page. Because the changes are global, the changes cannot be managed in Web Content Manager and so cannot be syndicated to another server. To transfer these changes to another server, use the XML configuration interface (**xmlaccess** command).

Edit Shared Settings mode

If you change the configuration settings for a portlet in **Edit Shared Settings** mode, these changes are part of the page. Because the changes are part of the page, the changes are managed in Web Content Manager and are automatically syndicated to other servers.

Automatic publishing and deleted items

When you specify automatic publishing for a project, the project is published as soon as all the items in the project reach a state of "pending." Deletions do not go through an explicit approval stage and are available for publishing immediately. If your project consists of only deletions, automatic publishing of the project can occur prematurely.

To prevent this automatic publishing, you can complete the following steps:

- Ensure that the project contains new pages or changes to pages, which require approval before publishing.
- Set the project to use manual publishing.

Syndication and versioning

- The versioning feature of Web Content Manager also applies to pages and is used to run different tasks with page versions. Versioning tasks include saving, deleting, and restoring versions. The Page Properties window lists the versions of the page on the **Advanced** tab. However, if you create a page and syndicate the page for the first time, the version information is empty when you view the page properties on the delivery server. After subsequent syndication operations, the version information is listed.

Portlets

Portlets are small applications that are independently developed, deployed, managed, and displayed. After the portlet is deployed, you can use it multiple times on different pages.

Almost everything on a page is rendered through a portlet. The portlet can provide access to enterprise data, content that is stored in IBM Web Content Manager, and external content from third-party sources. In the following image, there are five portlets on the page, but only one is highlighted. The highlighted Member Login Portlet is providing access to the enterprise LDAP user registry to enable members to authenticate.

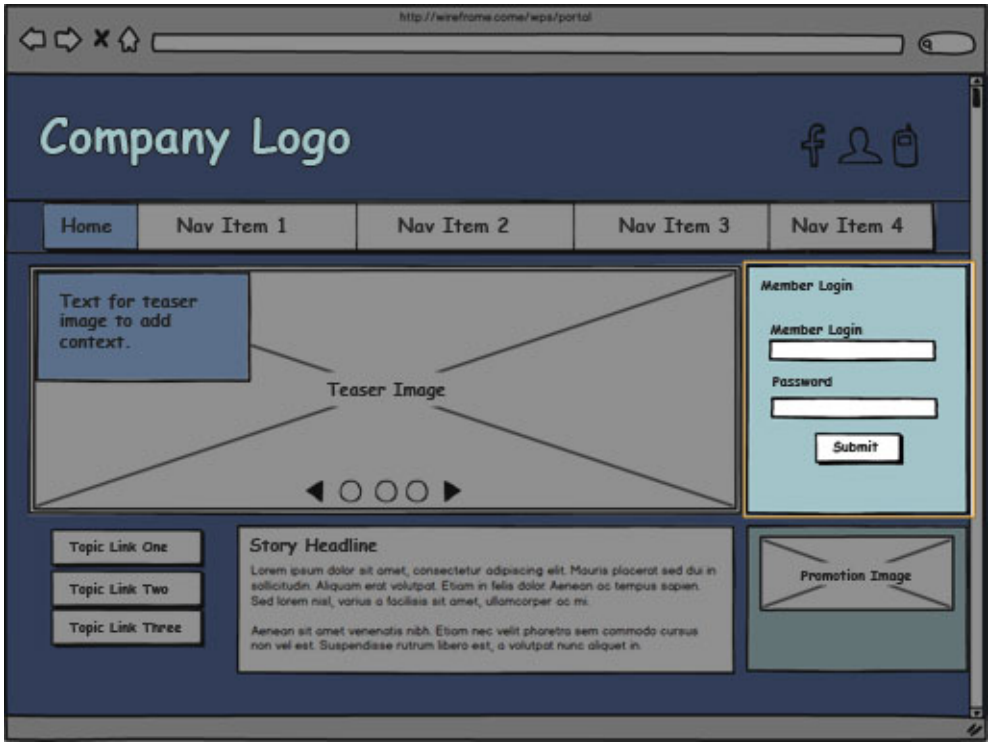


Figure 19. Wireframe of a website with portlet highlighted

Application palette

Content authors access portlets from the **Applications** palette (**Create** > **Applications**).

Content portlets

Specialized portlets are available to render content.

Web Content Viewer
: Renders content from Web Content Manager. When an author drags content from the **Content** palette onto a page, the content is rendered by using the Web Content Viewer Portlet.

Web Dock Portlet
: Renders content from third-party sources. And administrator must configure the content provider source and associate it with a Web Dock application. Then, content authors can add the Web Dock Portlet to a page.

Syndicated Feed Portlet
: Gets feeds from third-party feeds. The portlet is configured to retrieve weekly podcasts from IBM developerWorks.

Administration portlets

Use administrations portlets to configure portal. Administrations portlets are immediately available after you install the portal. To get to the administration portlets, click the **Administration menu** icon in the toolbar. Administration portlets are also available in the **Applications** palette.

Content

Rich content and elegant user experiences drive the success of websites. Each digital experience consists of many different types of content from many various sources.

In the example wireframe, content is highlighted. Depending on your point of view, all of the information in the page could be considered content. The highlighted article is the most obvious example of content. The content and associated presentation template is rendered through a portlet.

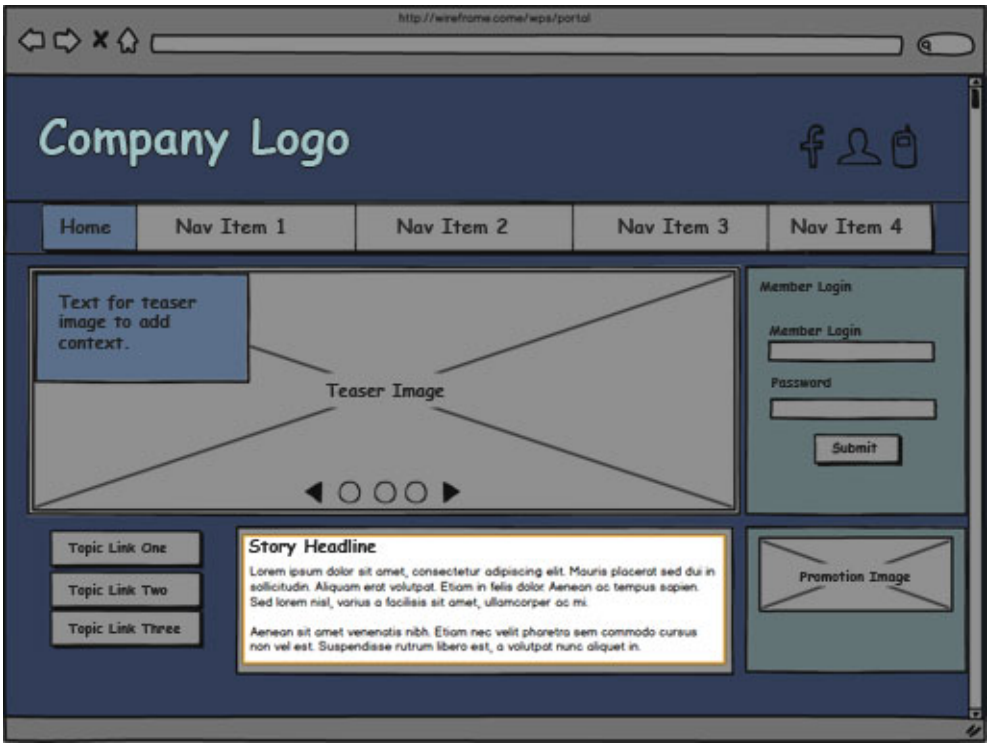


Figure 20. Wireframe with content highlighted

"Content stored in Web Content Manager" on page 1747
Use Web Content Manager to store and manage content. Web Content Manager separates the content from the presentation layer. Content authors enter information in authoring templates. The authoring template is then associated with a presentation template. The presentation template defines how the content is rendered and presented to the site visitor. As a result, you can rapidly change the look and feel of your content without editing the same elements repeatedly. Use the Web Content Viewer Portlet to render content from Web Content Manager.

"External content" on page 1747
Many capabilities are included to retrieve content from external sources. You can use Web Content Integrator, REST services for IBM Web Content Manager, IBM Digital Data Connector (DDC) for WebSphere Portal Express, specialized portlets, WSRP, and other mechanisms to retrieve and render content from sources outside your digital experience.

"Social Content" on page 1747
Integrating social content into your digital experiences adds to the rich user experience. Capabilities such as community pages and IBM Connections portlets make that integration possible.

Content stored in Web Content Manager

Use Web Content Manager to store and manage content. Web Content Manager separates the content from the presentation layer. Content authors enter information in authoring templates. The authoring template is then associated with a presentation template. The presentation template defines how the content is rendered and presented to the site visitor. As a result, you can rapidly change the look and feel of your content without editing the same elements repeatedly. Use the Web Content Viewer Portlet to render content from Web Content Manager.

External content

Many capabilities are included to retrieve content from external sources. You can use Web Content Integrator, REST services for IBM Web Content Manager, IBM Digital Data Connector (DDC) for WebSphere Portal Express, specialized portlets, WSRP, and other mechanisms to retrieve and render content from sources outside your digital experience.

IBM Web Content Integrator

The Web Content Integrator is a solution for integrating externally managed Web content with WebSphere Portal Express. Through the use of standard content syndication feed technologies based on RSS 2.0, the Web Content Integrator provides a loosely-coupled mechanism for transferring published content and metadata to the portal after they have been approved in the source system. Once the content and metadata have been transferred to the portal, it is possible to use the built-in content management features of WebSphere Portal Express to secure, personalize, and display the content to users.

REST service for Web Content Manager

The REST service for Web Content Manager is a collection of web services that are compliant with the Atom Publishing Protocol. They provide access to web content, including versions and workflow states, through HTTP. The service is designed according to the REST (REpresentational State Transfer) architectural style.

REST services make it easy to build interactive content, which can be modified directly by your site users. Responsive, integrated editing tools can be created by embedding HTML and JavaScript in web content components, which bind to the REST service to display or update content asynchronously. (Ajax)

Digital Data Connector

Use the IBM Digital Data Connector (DDC) for WebSphere Portal Express to integrate data from external data sources on portal pages by using IBM Web Content Manager presentation components. With Digital Data Connector, content authors and designers can use Web Content Manager presentation components to generate the web page markup for external data.

Social Content

Integrating social content into your digital experiences adds to the rich user experience. Capabilities such as community pages and IBM Connections portlets make that integration possible.

Integrate Connections into your site using community pages and Connections Portlets.

Community pages

Portlets on community pages are automatically scoped to the community membership and display content from the community in the portlets. For example, if your community contains a forum you can add the forum portlet to a community page. The portal site visitors can view and interact with the forum content from the portal site.

You can automatically create new communities for your pages during the page template instantiation.

Connections Portlets

Connection portlets are not installed with the portal but you can easily download them from IBM Lotus and WebSphere Portal Business Solutions Catalog.

The portlet includes activities, blogs, blog summary, bookmarks, bookmarks summary, profiles, profiles summary, wikis, forums, forums summary, community overview, and tags.

Planning a website

Before you build a website, take the time to analyze, plan, and design the entire system that supports the website. You must plan not only the website, but the web content system that manages the website and the infrastructure that is required for the system. You must also define in advance the roles and users that are involved in building the website and installing the infrastructure.

"Website objectives" on page 1749
It is important to have an understanding of the objectives, deliverables, and scope of a web content system. Your website definition outlines the "what, why, and who" of the project and can be used throughout the life of your website.

"Human resource planning" on page 1751
These roles are examples of the type of work performed by the people who create and manage a website. A single person can be responsible for more than one of the roles outlined in this section. The roles you implement in your organization to support your website are determined by the size and complexity of the system being deployed. Not all the following roles are required for every website, but all aspects of these roles must be considered during any system deployment.

"Plan your site with an analysis document" on page 1766
An analysis document is used to record the information that is gathered from stakeholders. It describes the design of the website, its content, and its features.

"How to design a prototype website by using HTML" on page 1768
Before you create a design document for your web content system, it can be useful to create a prototype of your site by using HTML. This prototype is based on the outline that is defined in your project plan and the data that is gathered in your analysis document. The site structure, design, and HTML code you develop for your prototype can be used as the basis for many of the items that are defined in your design document.

"The design document" on page 1769
When you have defined the project and created an analysis document, you then define the requirements of your web content system in a design document. The design document outlines what types of content is needed for your site, how they are structured, how content is authored, and what the final website looks like.

"Roadmap to building a web content system" on page 1781
To build a web content system you need to deploy hardware, configure servers,

design an authoring system, configure a delivery environment and enable syndication. Get an overview of the steps required to build your web content system. Keep in mind the analysis and design documents developed during the planning phase of a project as you review the roadmap.

Website objectives

It is important to have an understanding of the objectives, deliverables, and scope of a web content system. Your website definition outlines the "what, why, and who" of the project and can be used throughout the life of your website.

You define the following information in your project plan:

Background

Define the background information for the project. This information can include why the project is being undertaken and any history of previous and current related projects.

Mission

What is the mission of the project? Writing a mission statement can help in determining the requirements of the website.

Objectives

It is essential to determine the objectives of the project. These objectives can be provided by the project team, which is determined from meetings with the project stakeholders, and usability workshops with users. All stakeholders should agree on the final objectives of the site. Each objective must be clear and concise. There is no room for assumptions or varied interpretations.

Business objectives

These objectives define what the business wants to achieve. They focus on concerns such as profit and brand value.

The following are examples of business objectives for an internet site:

- Reduce costs of distributing press releases and sales materials
- Reduce the number of phone calls taken by the support team
- Strengthen existing customer loyalty
- Discover potential customers online

The following are examples of business objectives for an intranet:

- Provide specialized and tailored content to key groups within the company
- Ensure that employees feel valued
- Reduce business costs by making staff more productive by improving their core tasks

Operational objectives

Operational objectives can be grouped according to short-term and long-term objectives.

The following are examples of operational objectives:

- Provide information to company employees
- Provide information to customers

- Develop skills within the company to administer a website
- Develop single sign-on function

User objectives

These objectives define the needs of the user of the website and are crucial to developing the site objectives, structure, and function.

The following are examples of a user objective for an internet site:

- Make it easy for me to find what I want
- Make the information understandable and relevant
- Let me know where I am in the site
- Retain my privacy

The following are examples of a user objective for an intranet:

- Find up to date, relevant information as quickly as possible
- Keep me informed of latest news and updates
- Help me perform core business functions such as completing time-sheets and applying for leave
- Reduce my frustrations
- Let me publish information
- Help me feel connected, supported, and valued

Site objectives

Site objectives are derived from the business, operational, and user objectives. They might be the result of the intersection of the business, operational, and user objectives, or they might be extra objectives that result from analysis.

The following are examples of site objectives for an internet site:

- Provide clear and easy to understand navigation that enables users to find information quickly
- Provide a search feature
- Write content for the web so that it is easy to read and understand
- Provide an FAQ section that addresses the most frequently asked questions of the support team
- Provide a framework that structures content for the user and not the business division

The following are examples of site objectives for an intranet:

- Provide clear and easy to understand navigation that enables users to find information quickly
- Support key common tasks such as timesheet entry
- Provide a customized news section
- Provide an FAQ section that addresses the most frequently asked questions of the support team
- Provide a framework that structures content for the user and not the business division
- Enable staff to enter content that is then put through a workflow before it is published on the intranet

Project teams

Define the role of each of the teams that are involved in the project and the organization of the teams. The following are some project team examples:

Executive sponsors
The owners and drivers of the project.

Project team
Responsible for the day-to-day management, analysis, and construction of the new site.

Reference group
Business unit representatives that are consulted to ensure that their needs are addressed.

Focus group
User representatives that are consulted to ensure that the new site is user-focused.

Deliverables

Document all the deliverables of the project. Give a description of what the deliverables are and the expected time frame for delivery.

Human resource planning

These roles are examples of the type of work performed by the people who create and manage a website. A single person can be responsible for more than one of the roles outlined in this section. The roles you implement in your organization to support your website are determined by the size and complexity of the system being deployed. Not all the following roles are required for every website, but all aspects of these roles must be considered during any system deployment.

"The project manager" on page 1752
Large complex deployments require a project manager to ensure that there are sufficient resources available to deploy a system, that the correct users have been assigned to the project and that the tasks required to deploy the system occur at the correct time in the overall deployment process.

"The business analyst" on page 1753
A business analyst is responsible for developing an analysis document.

"The architecture and design team" on page 1753
Based on the data collected in the analysis document, the architecture and design team documents all the things required by the WebSphere Portal system. The design documents the team develops are used by the deployment and development teams to deploy and manage the WebSphere Portal system.

"The deployment team" on page 1757
Based on the design documents developed by the technical design team, the deployment team builds and manages the WebSphere Portal system.

"The development team" on page 1761
Based on the design documents developed by the technical design team, the development team is responsible for creating custom applications using the product API.

"The website creation team" on page 1762
Based on the design documents developed by the technical design team, the website creation team is responsible for creating the website and related systems.

"The content acquisition team" on page 1764
Based on the content identified in the analysis document, the content acquisition team is responsible for importing content from legacy systems into the new website. The team consists of a mixture of administrators and developers. Team members need to be experts in both legacy products and WebSphere Portal Express

"The maintenance team" on page 1765
Based on the maintenance architecture developed by the technical design team, the maintenance team is responsible for the ongoing monitoring and maintenance of the overall system.

The project manager

Large complex deployments require a project manager to ensure that there are sufficient resources available to deploy a system, that the correct users have been assigned to the project and that the tasks required to deploy the system occur at the correct time in the overall deployment process.

Defining the project:

It is important to have an understanding of the objectives, deliverables, and scope of a web content system. The project manager is responsible for creating a project plan to document the overall scope and goals of the project.

- "Website objectives" on page 1749

Overseeing the development of a site prototype and design documents:

After defining the project and creating an analysis document, the project manager is responsible for overseeing the development of a site prototype and design documents.

- "How to design a prototype website by using HTML" on page 1768
- "Plan your site with an analysis document" on page 1766

Ensuring that the project has the optimum level of human and physical recourses:

The project manager is responsible for ensuring that sufficient human and physical resources have been planned for the initial design, development, and ongoing maintenance of the website and related infrastructure.

Project managing the deployment and installation of the hardware and software used to manage and deliver the website:

The project manager is responsible for overseeing the deployment and installation of the hardware and software used to manage and deliver the website. The project manager monitors the progress of the overall deployment, and takes appropriate action to ensure that milestones are reached on time and on budget.

- Chapter 4, "Installing," on page 101

Project managing the ongoing maintenance and upkeep of the hardware and software used to manage and deliver the website:

The project manager also develops a plan for the ongoing monitoring and maintenance of the website and related infrastructure.

- "Web content administration tools" on page 1176

Related information:

Planning to install WebSphere Portal

"Planning a website" on page 1748
Before you build a website, take the time to analyze, plan, and design the entire system that supports the website. You must plan not only the website, but the web content system that manages the website and the infrastructure that is required for the system. You must also define in advance the roles and users that are involved in building the website and installing the infrastructure.

The business analyst

A business analyst is responsible for developing an analysis document.

The business analyst is responsible for developing an analysis document. This document is used to record the information gathered from stakeholders that determine the design of the website, its content, and its features. The analysis document includes information on:

- The types of users that will use the system
- What features and applications will be required on the website
- What existing content will need to be imported into the new web content system, and what new content will need to be created

Related concepts:

"Plan your site with an analysis document" on page 1766
An analysis document is used to record the information that is gathered from stakeholders. It describes the design of the website, its content, and its features.

Related information:

"Planning a website" on page 1748
Before you build a website, take the time to analyze, plan, and design the entire system that supports the website. You must plan not only the website, but the web content system that manages the website and the infrastructure that is required for the system. You must also define in advance the roles and users that are involved in building the website and installing the infrastructure.

The architecture and design team

Based on the data collected in the analysis document, the architecture and design team documents all the things required by the WebSphere Portal system. The design documents the team develops are used by the deployment and development teams to deploy and manage the WebSphere Portal system.

"The technical architecture team" on page 1754
The technical architects are responsible for designing the overall server topology required for the entire system and the architecture of the servers that make up the system. The technical architecture team can include technical architects, database architects, security architects, performance engineers, and other resources as required.

"The information architect" on page 1755
The information architect determines the overall informational structure of the website based on requirements gathered in the analysis document.

"The website designer" on page 1756
A website designer is responsible for designing the layout and style of the web pages in a website. A graphic designer might also be employed to create graphics and color schemes for the website.

"The authoring system architect" on page 1756
The authoring system architect is responsible for determining the architecture of the authoring system used to create and manage web content.

The technical architecture team:

The technical architects are responsible for designing the overall server topology required for the entire system and the architecture of the servers that make up the system. The technical architecture team can include technical architects, database architects, security architects, performance engineers, and other resources as required.

Determining a server architecture:
The technical architect is responsible for designing a server architecture describing all the servers and related software required to deliver the website.

- "Server architecture" on page 1770

Determining a database architecture:
The database architect is responsible for designing a database architecture describing all the configuration settings required for the databases used in the system.

- "Server architecture" on page 1770
- "Database" on page 119
- "User registry considerations" on page 128

Determining syndication strategies:
The technical architect determines what syndication strategies are required between the different environments in the web content system.

- "Server architecture" on page 1770
- "Syndication relationships" on page 449

Determining a security architecture:
The security architect is responsible for designing a security architecture describing all groups and roles, and access levels required to ensure that different types of users have access only to the functions and content appropriate to their roles.

- "Security architecture" on page 1771
- "Users, Groups and Roles" on page 1557
- Managing users and groups
- "Controlling access" on page 1524

Determining a content acquisition architecture:
The technical architect is responsible for designing a content acquisition describing the technical details of the methods to use when importing existing content in the new web content system. The content to import is based on the information collected in the analysis document.

- "Content acquisition architecture" on page 1779
- "Plan your site with an analysis document" on page 1766
- "The IBM Web Content Manager API" on page 3154
- "IBM Web Content Integrator" on page 1936
- "WebDAV" on page 1986

Determining delivery strategies:
The technical architect is responsible for determining which delivery methods are best suited for the types of websites being delivered.

- "Delivery architecture" on page 1779
- "Types of websites" on page 45

- "Delivering web content" on page 2016

Determining system maintenance strategies:

The technical architecture team is also responsible for determining the ongoing maintenance strategies and procedures of the system.

- "Maintenance architecture" on page 1780

Related information:

Server topologies

Installing WebSphere Portal

"Planning a website" on page 1748

Before you build a website, take the time to analyze, plan, and design the entire system that supports the website. You must plan not only the website, but the web content system that manages the website and the infrastructure that is required for the system. You must also define in advance the roles and users that are involved in building the website and installing the infrastructure.

The information architect:

The information architect determines the overall informational structure of the website based on requirements gathered in the analysis document.

Determining the website structure:

The web site structure designed by the information architect determines:

- What Portal pages will need to be created if using portlet delivery.
- What site areas will need to be created

Determining what content types are required:

The content types required for each section in the website structure determine:

- What authoring templates are required
- What presentation templates are required
- What template mappings are required

Determining a content profiling architecture:

The information architect determines what content profiling is required for features such as menus.

Related concepts:

Item management features

Web Content Manager includes a range of features that help you manage the web content items that are used in your system.

Related information:

Server topologies

Installing WebSphere Portal

"Delivering web content" on page 2016

The type of delivery method you use to deliver web content to your viewers depend on the type of content that is being delivered, and the type of viewers your website is intended for.

The website designer:

A website designer is responsible for designing the layout and style of the web pages in a website. A graphic designer might also be employed to create graphics and color schemes for the website.

Determining what themes, style sheets, and templates are required:
The website designer determines what themes, style sheets, and presentation templates are required to deliver the website.

- "Design architecture" on page 1776
- "How to design a prototype website by using HTML" on page 1768
- Creating a presentation template

Determining what elements and components are required:
The page types and presentation templates required for your site determine:

- What components are required
- What elements are required for each authoring template
- What elements are required to be stored in site areas
- "Components" on page 1795

Determining personalization strategies:
The website designer determines what personalization strategies are required for the website.

- "Personalization" on page 2240

The authoring system architect:

The authoring system architect is responsible for determining the architecture of the authoring system used to create and manage web content.

Designing the library architecture:
The authoring system architect determines what libraries are required for the website.

Determining what authoring templates are required:
The authoring system architect documents the configuration of each authoring template used in the authoring system and what template mappings are required to link authoring templates and presentation templates.

- "An overview of authoring templates" on page 1787
- "Template mappings" on page 1792

Designing a folder architecture:
The authoring system architect determines what folders are required to store item-types within logical groupings.

- Web content folders

Design a change management model:
The authoring system architect determines what workflows are required to manage approval processes and what items require a workflow, and also design project templates.

- "Workflow and change management" on page 1929
- "Projects" on page 1923

Determine version control strategies:

The authoring system architect determines what version control strategies are applied to the web content system.

- How to manage versions of items

Related concepts:

Item management features
Web Content Manager includes a range of features that help you manage the web content items that are used in your system.

Related information:

"Delivering web content" on page 2016
The type of delivery method you use to deliver web content to your viewers depend on the type of content that is being delivered, and the type of viewers your website is intended for.

The deployment team

Based on the design documents developed by the technical design team, the deployment team builds and manages the WebSphere Portal system.

"The database administrator"
A database administrator is responsible for deploying the database servers and data repositories based on the technical architecture developed by the database architect.

"The WebSphere Portal administrator" on page 1758
A WebSphere Portal administrator is responsible for the overall deployment and management of the servers in a WebSphere Portal deployment based on the architecture developed by the technical architect.

"The web content administrator" on page 1758
A web content administrator is responsible for configuring the web content servers within a WebSphere Portal system.

"The security administrator" on page 1760
A security administrator is responsible for securing the overall system including access control strategies and firewalls.

The database administrator:

A database administrator is responsible for deploying the database servers and data repositories based on the technical architecture developed by the database architect.

The database administrator is responsible for installing all databases used by the website and related systems. The databases the administrator sets up are based on the technical architecture developed by the technical and database architects. The database administrator will need to become familiar with the user documentation of the data and user repositories used by the web content system.

Related concepts:

"Server architecture" on page 1770
Your technical architects define what web content environments are required for your system and the servers that are required for each environment. This information ensures that you have sufficient hardware to support your web content system

Related information:


Server topologies

Installing WebSphere Portal

Database considerations

User registry considerations

The WebSphere Portal administrator:

A WebSphere Portal administrator is responsible for the overall deployment and management of the servers in a WebSphere Portal deployment based on the architecture developed by the technical architect.

Installing and configuring WebSphere Portal:
The WebSphere Portal administrator is responsible for installing and configuring all the instances of WebSphere Portal used in the overall system.

- Chapter 4, "Installing," on page 101
- Chapter 5, "Configuring," on page 231

Migration:
The WebSphere Portal administrator is responsible for migrating data and configuration settings from older versions of WebSphere Portal.

- Chapter 9, "Migrating," on page 785

Related concepts:

"Security architecture" on page 1771
The security architecture describes what groups are required for your site and what access is required for different groups to the authoring portlet and rendered website.

Related information:

 Server topologies

Installing WebSphere Portal

The web content administrator:

A web content administrator is responsible for configuring the web content servers within a WebSphere Portal system.

Web content authoring environments

In a web content authoring environment, a web content administrator performs the following tasks:

Create new pages
An administrator might need to create new pages to display additional authoring portlets used by different users, or for displaying web content viewer portlets to preview sites within.

- "Pages" on page 1736
- "Custom portal pages for authoring" on page 1897

Configure an authoring portlet
Each authoring portlet in an authoring environment needs to be configured to ensure that it has been configured correctly for the users using each authoring portlet.

- "Authoring portlet settings" on page 1915

Configure a web content viewer portlet
Each web content viewer portlet in an authoring environment needs to be configured to display the content that is being previewed.

- "Web Content Viewers" on page 2017
- "Displaying content with Web Content Viewers" on page 2034

Create web content libraries
The web content administrator creates a set of web content libraries based on the recommendations of the web content architect.

Clone a web content repository
Before enabling syndication, a web content administrator, with a database administrator, clones a web content repository from the authoring environment to other environments.

- "How to clone a web content repository" on page 1214

Manage syndication
Syndication relationships are normally created on subscriber servers. As you typically syndicate only out from an authoring environment to other environments, you would not normally create a syndication relationship from an authoring environment, but a web content administrator might need to manage syndication settings and configurations from time to time.

- "Syndication relationships" on page 449
- How to manage syndicators and subscribers
- "Syndication tuning" on page 456

Create and manage feed configurations and jobs
Sometimes web content is stored and maintained in external systems. You use the Web Content Integrator to consume content from external systems using feeds.

- Web content feed management

Tune an authoring environment configuration settings
Authoring environment configuration settings can be tuned for specific features.

- "Configuring a web content authoring environment" on page 393

Web content delivery environments

In a web content delivery environment, a web content administrator performs the following tasks:

Create new pages
An administrator might need to create new pages when displaying content using web content viewer portlets.

- "Pages" on page 1736

Configure a web content viewer portlet
Each web content viewer portlet in a delivery environment needs to be configured to display the correct content.

- "Web Content Viewers" on page 2017
- "Displaying content with Web Content Viewers" on page 2034

Manage syndication
A web content administrator creates syndication relationships to the delivery environment from the web content staging and authoring environments.

- "Syndication relationships" on page 449
- How to manage syndicators and subscribers
- "Syndication tuning" on page 456

Tune a delivery environment configuration settings

Delivery environment configuration settings can be tuned for specific features.

- "Configuring a web content delivery environment" on page 411

User acceptance test environment

A user acceptance testing environment is set up like authoring and delivery environments depending on the type of user acceptance test being performed. The tasks required to set up the UAT environment are the same as those for the authoring and delivery environments.

Related concepts:

"Delivery architecture" on page 1779
Your technical architect and information architect need to define what delivery methods are most appropriate for the website you are delivering.

"Server architecture" on page 1770
Your technical architects define what web content environments are required for your system and the servers that are required for each environment. This information ensures that you have sufficient hardware to support your web content system

Related information:

Server topologies

Installing WebSphere Portal

The security administrator:

A security administrator is responsible for securing the overall system including access control strategies and firewalls.

Implementing the security plan outlined in the security architecture document:

The security administrator is responsible for deploying the security features and processes outlined in the security architecture document developed by the security architect.

- "Security and authentication considerations" on page 1519
- "Security architecture" on page 1771

Managing the user registry:

The security administrator maintains the security registry by running various update delete tasks.

- "User registry" on page 562

Managing users and groups:

The security administrator maintains the users and groups stored in the user registry.

- Managing users and groups

Managing access control

The security administrator manages access controls to pages, portlets, modules, and other applications.

- "Controlling access" on page 1524

Additional security tasks

There are various other tasks carried out by the security administrator depending on what environment is being used.

- "Enabling step-up authentication and the **Remember me** cookie" on page 1587

Related information:

 Server topologies

Installing WebSphere Portal

The development team

Based on the design documents developed by the technical design team, the development team is responsible for creating custom applications using the product API.

"The portlet developer"
The portlet developer is responsible for developing new portlets.

"The theme developer"
A theme developer is responsible for creating new themes based on the designs developed by the information architect and the graphic designer.

"The web content developer" on page 1762
A web content developer is responsible for extending Web Content Manager by using the Web Content Manager API, developing JSP components and creating web content plug-ins.

The portlet developer:

The portlet developer is responsible for developing new portlets.

The portlet developer is primarily responsible for creating custom portlets used to aggregate content and function from external applications. The portlets they develop are based on the portlets identified in the design document.

Related concepts:

"Design architecture" on page 1776
The design architecture describes what your website looks like, and what components are needed to build your site. You need to define presentation templates, components, and themes.

Related information:

Developing

The theme developer:

A theme developer is responsible for creating new themes based on the designs developed by the information architect and the graphic designer.

The theme developer is primarily responsible for creating custom themes based on the Web site designs outlined in the design architecture and the HTML used in the website prototype. Themes are a set of JSPs, images, and style sheets packaged together in a common directory structure. They can either be packaged directly in the wps.war or inside a separate WAR file of their own, or with other themes, skins, and resources.

Related concepts:

"Design architecture" on page 1776
The design architecture describes what your website looks like, and what components are needed to build your site. You need to define presentation templates, components, and themes.

"How to design a prototype website by using HTML" on page 1768
Before you create a design document for your web content system, it can be useful to create a prototype of your site by using HTML. This prototype is based on the outline that is defined in your project plan and the data that is gathered in your analysis document. The site structure, design, and HTML code you develop for your prototype can be used as the basis for many of the items that are defined in your design document.

The web content developer:

A web content developer is responsible for extending Web Content Manager by using the Web Content Manager API, developing JSP components and creating web content plug-ins.

The Web Content Manager API
You can use the Web Content Manager API to extend the functions of Web Content Manager.

- "The IBM Web Content Manager API" on page 3154

Custom authoring interfaces
Custom authoring interfaces are used by content authors as an alternative to using the authoring portlet.

- "How to create a custom launch page" on page 3192
- "How to use remote actions" on page 3195

Custom plug-ins
You can create custom plug-ins to add custom features to your site such as custom workflow actions and text providers for multi-locale sites.

- "How to create custom plug-ins" on page 3205

Related concepts:

"Design architecture" on page 1776
The design architecture describes what your website looks like, and what components are needed to build your site. You need to define presentation templates, components, and themes.

The website creation team

Based on the design documents developed by the technical design team, the website creation team is responsible for creating the website and related systems.

"The website creator"
A website creator is responsible for building a website by creating presentation templates, authoring templates, site areas, components and categories. The web site creator is also responsible for creating content management items such as folders and workflows. The items created by a website creator are based on the designs created by the information architect and graphic designer.

"The web content author" on page 1764
A web content author is responsible for creating web content for the sites developed and managed using Web Content Manager.

"The web content manager" on page 1764
A web content manager is responsible for performing ongoing site maintenance activities.

The website creator:

A website creator is responsible for building a website by creating presentation templates, authoring templates, site areas, components and categories. The web site

creator is also responsible for creating content management items such as folders and workflows. The items created by a website creator are based on the designs created by the information architect and graphic designer.

Creating web content management items

Before creating a website, the web content creator begins by creating any required content management items:

Creating folders
Folders are used to store item-types within logical groupings.

- Working with folders

Creating workflows
Workflows are required to manage the approval processes of items.

- "Workflow and change management" on page 1929

Creating projects
Projects are used to manage the approval process of a collection of items.

- "Projects" on page 1923

Creating a website

To create a website the following Web Content Manager items will need to be created:

Site areas
Site areas are used to define the site framework of a site. Content items that form part of the site framework will be stored within the site framework.

- Creating site areas

Authoring templates
Authoring templates are required for the different types of pages in a website. Content items created by content authors will be based in the authoring templates created by the website creator.

- Creating authoring templates

Presentation templates
Presentation templates determine the layout of the pages in a website and are linked to authoring templates by the template maps defined in site areas.

- Creating a presentation template

Elements and components
Elements are added to authoring templates and site areas to store different types of content. Components are used to create single elements that can be reused across a website.

- "Elements" on page 1804

Categories
Categories are used to profile items that have been configured to allow content profiling.

- "Taxonomies, Categories, and keywords" on page 1847

Related concepts:

"Information architecture" on page 1774
The information architecture describes the information structure of the site and

how users browse through the site.

"Design architecture" on page 1776
The design architecture describes what your website looks like, and what components are needed to build your site. You need to define presentation templates, components, and themes.

The web content author:

A web content author is responsible for creating web content for the sites developed and managed using Web Content Manager.

Content authors create content either by accessing an authoring portlet, or by using a custom content authoring interface.

Creating content items
Presentation templates determine the layout of the pages in a website and are linked to authoring templates by the template mappings defined in site areas.

- "Content items" on page 1794

Working with elements and components
Elements are added to authoring templates and site areas to store different types of content. Components are used to create single elements that can be reused across a website.

- "Elements" on page 1804

The web content manager:

A web content manager is responsible for performing ongoing site maintenance activities.

Moving, linking and copying items:
From time to time a web content manager will need to redesign the structure of a website by moving or copying items.

Managing authoring templates:
From time to time a web content manager will need to reapply or change the authoring template used by content items and site areas.

Restoring items:
From time to time a web content manager will need to restore an item to a previously saved version.

Editing user profiles:
Profiling information specific to a web content system can be added to a user profile to allow you to personalize content for a specific set of users.

Managing access controls
From time to time a web content manager will need to reapply or change the access controls of a set of items.

The content acquisition team

Based on the content identified in the analysis document, the content acquisition team is responsible for importing content from legacy systems into the new website. The team consists of a mixture of administrators and developers. Team members need to be experts in both legacy products and WebSphere Portal Express

The Web Content Manager API:

You can use the Web Content Manager API to import content into Web Content Manager.

- "The IBM Web Content Manager API" on page 3154

Web content feeds

You use the Web Content Integrator to consume content from external systems using feeds.

- "IBM Web Content Integrator" on page 1936

Using WebDAV for web content:

With WebDAV for WebSphere Portal Express, you can use standard operating system tools to create, modify, and delete web content rather than the standard authoring portlet.

- "WebDAV" on page 1986

Related concepts:

"Content acquisition architecture" on page 1779
The content acquisition architecture is used to define what existing content will be imported into your web content system, how it is imported, and what content needs to be created.

The maintenance team

Based on the maintenance architecture developed by the technical design team, the maintenance team is responsible for the ongoing monitoring and maintenance of the overall system.

The database administrator

After all databases have been setup, the database administrator is then responsible for the ongoing monitoring and maintenance of all databases in the system.

The WebSphere Portal administrator

The WebSphere Portal administrator is responsible for ongoing maintenance of the WebSphere Portal servers in the system.

- Chapter 11, "Administering," on page 1041

The web content administrator

Manage syndication:

Syndication relationships are normally created on subscriber servers. As you typically syndicate only out from an authoring environment to other environments, you would not normally create a syndication relationship from an authoring environment, but a web content administrator might need to manage syndication settings and configurations from time to time.

- "Syndication relationships" on page 449
- Working with syndicators and subscribers
- "Syndication tuning" on page 456

Manage feed configurations and jobs:

Sometimes web content is stored and maintained in external systems. You use the Web Content Integrator to consume content from external systems using feeds.

- Web content feed management

Maintain an authoring system:
From time to time a web content administrator needs to perform some maintenance tasks.

- "Web content administration tools" on page 1176

The security administrator

Managing the user registry:
The security administrator maintains the security registry by running various update delete tasks.

- "Updating your user registry" on page 585

Managing the users and groups:
The security administrator maintains the users and groups stored in the user registry.

- Managing users and groups

Managing access control:
The security administrator manages access controls to pages, portlets, modules, and other applications.

- "Controlling access" on page 1524

Additional security tasks:
There are various other tasks carried out by the security administrator depending on the environment being used.

- "Enabling step-up authentication and the **Remember me** cookie" on page 1587

The performance engineer

The performance engineer is responsible for ensuring that all the hardware and software in the system are tuned for maximum performance and reliability.

- Chapter 17, "Troubleshooting," on page 3537

Related concepts:

"Maintenance architecture" on page 1780
You need to plan for the tasks that maintain the health and integrity of your web content system. Your technical architect and database architect need to define what maintenance procedures are required for your system, and when and how often they need to be run.

Plan your site with an analysis document

An analysis document is used to record the information that is gathered from stakeholders. It describes the design of the website, its content, and its features.

Here are some examples of the analysis that can be undertaken when you design a web content system.

User analysis

To design a website that supports the needs of the company and the users, you must know who your audience is. It is important to determine your users at this early stage of the project. Some of the things you want to discover are:

- Who they are?
- Who are the most important groups?
- What do they want to do on the site?

- What makes them return to the site?
- What is their level of experience with the web?

To help you understand your main user groups even further, you can develop personas and scenarios:

Personas

A persona is a fictional person who represents a major user group for your site. By using the information that is gathered about your users, create a person who represents each main user group. Give them:

- A name and picture
- Demographics, such as age, education, and family status
- Job role and responsibilities
- Their goals and tasks in relation to the site
- A background on their computer and web usage

Scenarios

A scenario is a story of how users might experience the site. They help you visualize the site and its users. They can help you view the navigation process as a whole. Scenarios are also useful in validating the website design after it is finished and can be used in usability testing. Use your personas, and give them a task to accomplish on the site. Write a story about how the character uses the site to finish the task. Be creative.

Competitive Analysis

If you are building a public website, it is useful to look at what the competition is doing. Generate a list of competitors and document things you like and dislike about their Internet sites. For intranet sites where you cannot compare your site with websites of competitors, you can instead ensure that your intranet meets current standards.

Website requirements

Website requirements describe the features and functions of a website. They document what the site must have and also what users can do. The requirements are not used to describe how to build the website, which is detailed in the design document.

For example:

General features:

- Search
- Contact details

Allow users to:

- Purchase a product
- Sign up for a newsletter
- Complete a timesheet online

Include the following content and site areas:

- Press releases
- Policies and guides
- Links to related articles

Content Inventory

It is useful to identify the types of content that make up the site. As your new website might be a redesign of an existing site, identify what content exists and what new content needs to be written. Create a content inventory, and add any existing web pages and potential types of content that you can think of. Types of content include:

- Static content such as copyright notices and privacy statements
- Dynamic content such as latest news and product campaigns
- Transactional content such as logon pages and registration pages for email newsletters

When you create a content inventory, you can collect the following information:

- A brief description
- Topic area or category
- Priority
- Format, such as a web page, a file, or on paper
- Intended audience
- Related content
- Created date
- Last modified date
- Owner
- Author
- Expiration date

How to design a prototype website by using HTML

Before you create a design document for your web content system, it can be useful to create a prototype of your site by using HTML. This prototype is based on the outline that is defined in your project plan and the data that is gathered in your analysis document. The site structure, design, and HTML code you develop for your prototype can be used as the basis for many of the items that are defined in your design document.

By creating a prototype website by using HTML, you get to see what your actual website looks like, plus you develop many of the features that are used by your web content system and website.

Building the prototype

- The prototype website includes example pages of each page type that is defined in your project plan and analysis document. It also includes the basic site structure of your proposed website.
- The prototype can be a functional website. It can include functioning navigational features so you can browse through the prototype. Examples of real content are used to populate the prototype.
- Take care to ensure that the HTML and other code you write to create the prototype meets coding best practices because much of the code you develop for the prototype is reused in your Web Content Manager site.

How to use HTML to plan the design of your site

A useful method to begin the design of your content management system is to use HTML to develop a prototype of your site. Do this task in consultation with the web content information architect.

The HTML site can include:

- The design for the main home pages of each section of your site
- The design for content pages that are stored in each section
- The navigation features of your site

The HTML site can then be used to help you determine what web content items need to be created for your web content management system.

For example, the design of each web page in the HTML mock-up can determine different parts of your Web Content Manager site:

- The layout of each HTML page determines the layout of your presentation templates.
- The design of the navigation features determines what menus and navigators you need to create.

The presentation template is used to combine elements from different parts of the web content system, such as site areas, content items, and components.

Other features of your design, such as images and style sheets, determine what components need to be created and whether you need to develop a new WebSphere Portal theme.

The design document

When you have defined the project and created an analysis document, you then define the requirements of your web content system in a design document. The design document outlines what types of content is needed for your site, how they are structured, how content is authored, and what the final website looks like.

"Server architecture" on page 1770
Your technical architects define what web content environments are required for your system and the servers that are required for each environment. This information ensures that you have sufficient hardware to support your web content system

"Security architecture" on page 1771
The security architecture describes what groups are required for your site and what access is required for different groups to the authoring portlet and rendered website.

"Information architecture" on page 1774
The information architecture describes the information structure of the site and how users browse through the site.

"Design architecture" on page 1776
The design architecture describes what your website looks like, and what components are needed to build your site. You need to define presentation templates, components, and themes.

"Authoring architecture" on page 1777
The authoring architecture describes what types of content are required for the site and what change management strategies are applied when you update content and design. You need to define what authoring templates are required

for your system, what workflows are used to manage changes and what folders are used to group items in the authoring portlet.

"Content acquisition architecture" on page 1779
The content acquisition architecture is used to define what existing content will be imported into your web content system, how it is imported, and what content needs to be created.

"Delivery architecture" on page 1779
Your technical architect and information architect need to define what delivery methods are most appropriate for the website you are delivering.

"Maintenance architecture" on page 1780
You need to plan for the tasks that maintain the health and integrity of your web content system. Your technical architect and database architect need to define what maintenance procedures are required for your system, and when and how often they need to be run.

Server architecture

Your technical architects define what web content environments are required for your system and the servers that are required for each environment. This information ensures that you have sufficient hardware to support your web content system

Environmental architecture

Every Web Content Manager system consists of at least one authoring environment and one delivery environment. The authoring environment is used to create and manage the individual items that make up your website. The delivery system is used to deliver the web content to your viewers. In addition to these environments, you might also require testing environments where you undertake user acceptance testing (UAT).

The technical architecture team documents the overall environmental architecture that is required for your Web Content Manager system including:

- What authoring, delivery and UAT environments are required
- What WebSphere Portal servers or clustered are required in each environment
- What database servers are required to store the data repositories that are used by each WebSphere Portal server or cluster
- What syndication relationships are required between each environment
- What user repositories, such as LDAP, are required

Server architectures

The technical architecture team also documents a server architecture for each server that is described in the environmental architecture. A server architecture describes the detailed setup of each server, including:

- What hardware is required for each server
- What operating system is required for each server
- What software is required
- Configuration settings for all hardware, software, and operating systems
- Other relationships such as remote data repositories, and user repositories

Security architecture

The security architecture describes what groups are required for your site and what access is required for different groups to the authoring portlet and rendered website.

Note: The following example describes the type of security architecture that is required for an authoring environment. In most cases, the security architecture for a staging or delivery environment would be much simpler with only the **All Portal User Groups** group assigned user access to the library. This strategy prevents users from being able to edit content and disables features like authoring tools from being displayed on the published site.

In this example, item type roles are applied to the following groups:

Table 248. Groups

Group	Details
WCMAdmins	Members of this group require access to all features of the authoring portlet.
SiteAdmins	Members of this group require access to all features of the authoring portlet except workflow.
SiteDesigners	Members of this group require access to content items, presentation templates, authoring templates, and components.
ContentAuthors	Members of this group require editor access to content items only.
ContentReviewers	Members of this group require Reviewer access to content items only.

Library access

The simplest method of setting library access is to grant contributor access to all your groups. This access gives all users and groups contributor access to the library and authoring portlet. Extra access is then granted to each group by using resource permissions. You can also grant the Anonymous Portal User group user access to ensure that all anonymous users can access the library if anonymous access is required for your website.

Table 249. Library access

Roles	Allow propagation	Allow inheritance	User or group
Administrator	Yes	Yes	
Manager	Yes	Yes	
Editor	Yes	Yes	
User	No	Yes	Anonymous Portal User
Contributor	Yes	Yes	WCMAdmins SiteAdmins SiteDesigners ContentAuthors ContentReviewers

Resource permissions

Set the following resource permissions for each role type:

- The WCMAdmins group is assigned the administrator role for all resources.
- The SiteAdmins group is assigned the manager role to all resources except workflow and workflow elements as they do not require access to these resources.
- The other groups are assigned roles for each resource according to the following tables.

Authoring templates

The SiteDesigners group is assigned editor access to authoring templates as they are required to create new authoring templates.

Table 250. Authoring templates

Roles	Allow propagation	Allow inheritance	User or group
Administrator	Yes	Yes	WCMAdmins
Manager	Yes	Yes	SiteAdmins
Editor	Yes	Yes	SiteDesigners
User	Yes	Yes	
Contributor	Yes	Yes	

Components

SiteDesigners are assigned editor access to components as they are required to create components.

Table 251. Components

Roles	Allow propagation	Allow inheritance	User or group
Administrator	Yes	Yes	WCMAdmins
Manager	Yes	Yes	SiteAdmins
Editor	Yes	Yes	SiteDesigners
User	Yes	Yes	
Contributor	Yes	Yes	

Content

Both the SiteDesigners and ContentAuthors groups are assigned editor access to content as they are required to create content items.

The ContentReviewers group is assigned Reviewer access only, because they are not required to create new content items, but need Reviewer access to content items during a workflow. You must also assign the ContentReviewers group approve access in the properties section of any workflow stages that ContentReviewers use to approve content items.

Table 252. Content

Roles	Allow propagation	Allow inheritance	User or group
Administrator	Yes	Yes	WCMAdmins
Manager	Yes	Yes	SiteAdmins

Table 252. Content (continued)

Roles	Allow propagation	Allow inheritance	User or group
Editor	Yes	Yes	SiteDesigners ContentAuthors
User	Yes	Yes	
Contributor	Yes	Yes	
Reviewer	Yes	Yes	ContentReviewers

Presentation Templates

The SiteDesigners group is assigned editor access to presentation templates as they are required to create new presentation templates.

Table 253. Presentation templates

Roles	Allow propagation	Allow inheritance	User or group
Administrator	Yes	Yes	WCMAdmins
Manager	Yes	Yes	SiteAdmins
Editor	Yes	Yes	SiteDesigners
User	Yes	Yes	
Contributor	Yes	Yes	

Site areas and pages

Only the WCMAdmins and SiteAdmins groups require access to site areas and pages as these are the only groups that build site frameworks.

Table 254. Site areas and pages

Roles	Allow propagation	Allow inheritance	User or group
Administrator	Yes	Yes	WCMAdmins
Manager	Yes	Yes	SiteAdmins
Editor	Yes	Yes	
User	Yes	Yes	
Contributor	Yes	Yes	

Taxonomy

Only the WCMAdmins and SiteAdmins groups require access to taxonomies as these are the only groups that build taxonomies.

Table 255. Taxonomy

Roles	Allow propagation	Allow inheritance	User or group
Administrator	Yes	Yes	WCMAdmins
Manager	Yes	Yes	SiteAdmins
Editor	Yes	Yes	
User	Yes	Yes	
Contributor	Yes	Yes	

Workflow Items

Only the WCMAdmins group requires access to workflow items as this is the only group that creates workflows. The groups that use workflows do not require access to the workflow items resource permissions.

Table 256. Workflow Items

Roles	Allow propagation	Allow inheritance	User or group
Administrator	Yes	Yes	WCMAdmins
Manager	Yes	Yes	
Editor	Yes	Yes	
User	Yes	Yes	
Contributor	Yes	Yes	

Item-level security inheritance

By default, each role's access is automatically inherited down to each item in a library. To prevent a user or group from automatically having inherited access to an item, you need to turn off inheritance on that item.

The permissions set for item type do not automatically give you access to individual items. They give you access only to specific tasks and views within the authoring portlet.

You can also assign specific access to individual groups or users on each item.

Related concepts:

"User registry considerations" on page 128
A user registry or repository authenticates a user and retrieves information about users and groups to do security-related functions, including authentication and authorization.

"Users, Groups and Roles" on page 1557
Your content management system requires different types of users. You need to create a different group for each type of user and then assign those groups different roles within your system.

Related tasks:

"Setting up access" on page 1801
Access controls allow you to assign access to who has the ability to view rendered constant and pages, and who has the ability to edit or administer content, pages, or features.

Information architecture

The information architecture describes the information structure of the site and how users browse through the site.

The site map

The website structure that is designed by the information architect determines what pages and site areas need to be created to give your site a hierarchical structure. The site map describes the structure of the site and determines what pages and site areas are required for your site. For example:

- Home page
  - News Page
  - Products Page

    - Product Site Area 1
    - Product Site Area 2
    - Product Site Area 3
  – Downloads Page
  – Support Page

Content types

The content types that are identified by the information architect determine what authoring templates are required for your authoring system. For example, your site might require the following content types:

- Section home pages
- News articles
- Employee profiles
- Product information
- Photo galleries
- Legal disclaimers

Content profiling and taxonomies

The information architect is responsible for determining what taxonomies are required to allow users to profile content. This information determines what content is displayed within menu components.

This is an example of a taxonomy for a financial services company:

- MetaBank taxonomy:
  – Financial
    - Banking Solutions
    - Interest Rates
      - Personal
      - Business
      - Corporate
  – News

Related concepts:

Site framework
A site framework is a hierarchical set of pages and site areas that are used to define the navigational structure of your website. You store content items within the site framework by saving them under different pages or site areas.

"An overview of authoring templates" on page 1787
Authoring templates are like forms that content authors can use to create new content. It defines default settings for the items that are created by using the authoring template. There are two types in authoring templates, site area and content.

"Presentation templates" on page 1789
You use a presentation template define the layout of your web content. Use tags to determine which properties, elements, or components are displayed.

"Taxonomy" on page 1801
Before creating a taxonomy, you should analyze how the taxonomy will be used in your site to determine the best structure for your taxonomy.

Design architecture

The design architecture describes what your website looks like, and what components are needed to build your site. You need to define presentation templates, components, and themes.

Themes and style-sheets

You list the themes, style-sheets, and styles in the design architecture. These designs can be based on the styles that are developed for your prototype website.

Page wireframes

Page wireframes can be used to describe the basic structure of each page-type in your site and the elements that are used on each page.

Presentation templates

You list each presentation template that is required for your website plus its HTML in the design architecture. The HTML you developed for the prototype website can be used as the basis for the HTML in the design document. The presentation template design includes:

- The HTML design of the presentation template
- Access settings
- Whether a workflow is required and, if so, which one

Components and elements

Each component and element that is required for your website is listed in the design architecture, and additional information such as:

- HTML design for any required component fields
- Parameter selections for any component parameters
- Access settings
- Whether a workflow is required and, if so, which one

Personalization strategies

The website designer also determines what content needs to be personalized for specific users. For example, you might want to display a personalized home page for each individual user based on the user profile. The personalization strategies that are documented in the design architecture determines what rules, content spots, and campaigns need to be created when the Personalization application is used.

Related concepts:

"Presentation templates" on page 1789
You use a presentation template define the layout of your web content. Use tags to determine which properties, elements, or components are displayed.

"Components" on page 1795
You use components to store elements that are used in more than one area of your website. For example, a company logo or a copyright notice.

"Personalization" on page 2240
Portal Personalization provides automatic customization of website content for individual users and user groups.

Authoring architecture

The authoring architecture describes what types of content are required for the site and what change management strategies are applied when you update content and design. You need to define what authoring templates are required for your system, what workflows are used to manage changes and what folders are used to group items in the authoring portlet.

Library architecture

The library architecture describes where you store your web content items. For example, you might split your site between the items that control the design of your site, such as presentation templates and components, from the content of your site. You might also separate your libraries into different team libraries such as "human resources" and "support".

Authoring templates

You need a separate authoring template for each type of content item and site area that is used by your site. The list of authoring templates that are defined in your authoring architecture is based on the different page types that are identified in the project plan, analysis document, and prototype website.

The information that is specified for each authoring template includes:

- Which content type to use. For example:
  - A site area for a parent item
  - A content item for the children of the parent item
  - A site area for a subsection of a parent item
- For the authoring template itself:
  - Name
  - Display title
  - Description
  - Template type
  - Access settings
- Default form settings including:
  - Default presentation template, if required
  - Default style sheet, if required
  - Form layout
  - Whether editors can manage elements
  - Actions to hide
  - Where to save items
  - Help text
  - Versioning strategy
- Default content and fields for the item:
  - Name and field display options
  - Display title and field display options
  - Description and field display options
  - Required elements
    - HTML design for any default element content
    - Default parameter selections for any component parameters

    - Field display options
- Default properties for each item:
  – Any default profiling and field display options
  – Any default workflow parameters and field display options
  – Any default access settings and field display options
  – Field display options for the History section

Workflows

You use workflows to control the access to, verification of, and eventual approval of items. Only if an item is approved at all stages up to a published stage can it be viewed on your website. You can use a workflow to:

- Review the accuracy of content.
- Review content for any legal implications.
- Review content to ensure that it meets accessibility guidelines.
- Ensure that no malicious code such as cross scripting attacks is added to content.

Each workflow, workflow stage, and workflow action that is required for your web content system is listed in the authoring architecture.

Folders

Folders are used to group sets of item types into logical groupings. This is useful when you have large numbers of items in your library and want to distinguish between different groups of items within each item type view. Each folder that is required for your web content system is listed in the authoring architecture.

For example, you might use a set of folders to groups different sets of image components:

- Logos
- Diagrams
- Staff photos
- Product photos

Version management strategies

You can configure your system to either automatically save a version of an item each time it is published, or allow users to select when to save a version of an item. The version management strategy for different item types is documented in the authoring architecture.

Related concepts:

"Content libraries" on page 1798
Content libraries store the assets for your website, including but not limited to pages, content, images, authoring and presentation templates, and workflows.

"Users, Groups and Roles" on page 1557
Your content management system requires different types of users. You need to create a different group for each type of user and then assign those groups different roles within your system.

Web content folders
You use folders to group sets of web content item types into logical groupings.

"Workflow and change management" on page 1929
You can manage changes to web content items either by creating drafts, by using workflows, or adding items to projects.

How to manage versions of items
You can configure your system to either automatically save a version of an item each time it is published, or allow users to select when to save a version of an item. You can restore items individually, or choose to restore a set of items within a library that either have the same label or were versioned at, or before, a specified date and time.

Content acquisition architecture

The content acquisition architecture is used to define what existing content will be imported into your web content system, how it is imported, and what content needs to be created.

The content acquisition architecture is based on the content inventory that was collected during the analysis phase of the project. For each piece of content that is identified in the content inventory you define the following information:

- Will this content be imported into the web content system, or will it be created afresh.
- If the content is to be imported, what tool will be used. For example, WebDAV or the Web Content Integrator.

Additionally, the type of items being created need to be identified. For example:

- Reusable pieces of content are best created as components
- Web page-specific content is best created as a content item
- Content specific to a particular section of a website is best saved within a site area

Related concepts:

"The IBM Web Content Manager API" on page 3154
You can use the Web Content Manager API to extend functions of Web Content Manager.

"IBM Web Content Integrator" on page 1936
The Web Content Integrator is a solution for integrating externally managed web content with WebSphere Portal Express. By using standard content syndication feed technologies based on RSS 2.0, the Web Content Integrator provides a loosely coupled mechanism for transferring published content and metadata to the portal after they are approved in the source system. When the content and metadata are transferred to the portal, it is possible to use the built-in content management features of Web Content Manager to secure, personalize, and display the content to users.

Related tasks:

"WebDAV" on page 1986
With WebDAV for IBM WebSphere Portal Express, you can use standard operating system tools to create, modify, and delete web content rather than the standard authoring portlet.

Delivery architecture

Your technical architect and information architect need to define what delivery methods are most appropriate for the website you are delivering.

The delivery architecture describes which delivery methods are required for your web content system.

Pre-rendered delivery

You can pre-render a complete website into HTML and save it to disk. The pre-rendered site can then be used as your live site and displayed to users by using either Web Content Manager or a web server. You deploy a pre-rendered site when you are not using any WebSphere Portal features, such as portlets, and your content is static and is updated only periodically.

Servlet delivery

Users can access content that is displayed by the Web Content Manager servlet. A servlet-delivered website is used when you don't need to use any WebSphere Portal based features such as authoring tools.

Local web content viewer delivery

Web content viewers are portlets that display content from a web content library as part of a portal page. If your presentation is simple, a single web content viewer can be sufficient, but you can also use multiple web content viewers to aggregate content from different libraries and provide a richer experience for your users. A local web content view portlet is used to display content within your web content delivery environment.

Remote web content viewer delivery

A remote web content view portlet is used to display content on a remote WebSphere Portal server or cluster.

Related information:

"Delivering web content" on page 2016
The type of delivery method you use to deliver web content to your viewers depend on the type of content that is being delivered, and the type of viewers your website is intended for.

Maintenance architecture

You need to plan for the tasks that maintain the health and integrity of your web content system. Your technical architect and database architect need to define what maintenance procedures are required for your system, and when and how often they need to be run.

Backup and restore strategies

Document backup and restore strategies for all the software and repositories in your web content system. This approach includes documenting procedures to back up and restore:

- Data repositories
- Themes
- Portlets
- Page structures
- Configuration settings
- Custom application

Applying updates

The process and timing of software updates are defined in the maintenance architecture. This document not only applies to Web Content Manager and WebSphere Portal, but to all software products in your web content system.

For example, you would document procedures to install fix packs for Web Content Manager regularly.

Running maintenance tools

You document the process and timing of running each maintenance tool. This documentation not only applies to Web Content Manager and WebSphere Portal, but to all software products in your web content system.

For example, you would document when and how often to run maintenance tools, such as the member fixer task, and when to clear item and version histories to maintain server performance.

Related concepts:

"Web content administration tools" on page 1176
IBM Web Content Manager includes tasks and tools to help maintain your content management system. For example, use the member fixer task to resolve renamed or deleted users and user groups. Use the workflow checker and updater tools to modify workflow security settings, reschedule pending actions, and add workflow to items. Web Content Manager also includes tools to assist with library and item management, and item and version history management.

Roadmap to building a web content system

To build a web content system you need to deploy hardware, configure servers, design an authoring system, configure a delivery environment and enable syndication. Get an overview of the steps required to build your web content system. Keep in mind the analysis and design documents developed during the planning phase of a project as you review the roadmap.

1. "Deploying the authoring environment" on page 1782
   The first environment to deploy is your authoring environment. This environment can initially be used by a small team as a development environment, but can eventually be used as the authoring environment for all your users. The authoring environment is deployed based on the technical architecture that is defined in the project design documents.
2. "Building the content authoring system" on page 1783
   Before you can start adding content to your web content system, you need to create all the items that are used to manage and deliver your web content.
3. "Importing and creating content" on page 1784
   When the authoring system is ready, you then import and create the default content for your system. This task is often managed by a separate team from the team that builds the web content authoring system.
4. "Deploying the delivery environment" on page 1784
   The delivery environment is used to deliver your website to your website viewers. The delivery environment is deployed based on the requirements that are defined in the project design document.
5. "Going live with your website" on page 1785
   When your environments are installed, the authoring system is completed, your default content is created, and fully tested the system, you are ready to go live.

Related tasks:

Installing WebSphere Portal Express and Web Content Manager
Installing the digital experience software includes preparing your operating system and using IBM Installation Manager to install IBM WebSphere Portal Express and Web Content Manager. Then, use the Configuration Wizard to complete the deployment configuration.

Deploying the authoring environment

The first environment to deploy is your authoring environment. This environment can initially be used by a small team as a development environment, but can eventually be used as the authoring environment for all your users. The authoring environment is deployed based on the technical architecture that is defined in the project design documents.

About this task

Procedure

1. Based on the database architecture that is defined in the project design document, the database administrator deploys a database server for the authoring environment.
   - "Database" on page 119
2. Based on the security architecture that is defined in the project design document, the LDAP administrator:
   a. Creates an LDAP server if one does not exist.
   b. Creates all the groups and users required for the web content system.

   "Security architecture" on page 1771
3. Based on the server architecture that is defined in the project design document, the WebSphere Portal administrator does the following:
   a. Installs a WebSphere Portal server or cluster of servers.
   b. Configures the WebSphere Portal server or cluster to use the database server setup by the database administrator.
   c. Configures WebSphere Portal to use the LDAP server as its user repository.
   d. Configures various WebSphere Application Server, WebSphere Portal and Web Content Manager configuration properties to ensure that the system is correctly setup for web content authoring and is tuned for optimal performance.
   - Chapter 4, "Installing," on page 101
   - Chapter 5, "Configuring," on page 231
   - "Web Content Manager" on page 392
4. Based on the information architecture and security architecture that is defined in the project design document, the WebSphere Portal administrator:
   a. Creates all the libraries that are required for the web content system.
   b. Assigns users and groups to the roles and access permissions.
   c. Create all pages that are required by the web content system.
   d. Adds all required authoring portlets to the appropriate pages and configures the authoring portlets for use by different teams to create and manage web content items.
   e. Adds all required web content viewer portlets to the required pages for use as preview pages.
   - "Users, Groups and Roles" on page 1557

- "Security architecture" on page 1771
- "Displaying content with Web Content Viewers" on page 2034

5. Final testing and tuning of the authoring environment is undertaken by all administrators.

Results

The authoring environment is now ready to use.

Building the content authoring system

Before you can start adding content to your web content system, you need to create all the items that are used to manage and deliver your web content.

Procedure

1. Based on the design architecture that is defined in the project design document, the web content developer creates plug-ins and JSP code in preparation for the creation of the content authoring system.
   - "The web content developer" on page 1762
   - "Design architecture" on page 1776
   - Developing
2. Based on the information architecture that is defined in the project design document, the website creator does the following:
   a. Creates site areas.
   b. Creates taxonomies and categories.
   - "The website creator" on page 1762
   - "Information architecture" on page 1774
3. Based on the content authoring architecture that is defined in the project design document, the website creator does the following:
   a. Creates workflow actions, stages, and defines workflows.
   b. Creates authoring templates.
   - "The website creator" on page 1762
   - "Authoring architecture" on page 1777
4. Based on the design architecture that is defined in the project design document, the website creator does the following:
   a. Creates components.
   b. Creates presentation templates.
   - "The website creator" on page 1762
   - "Design architecture" on page 1776
   - Creating components
   - Creating a presentation template
5. Based on the information architecture that is defined in the project design document, the website creator edits site areas and defines template mappings.
   - "The website creator" on page 1762
   - "Information architecture" on page 1774
   - "Template mappings" on page 1792
6. If needed, the web content developer also creates a custom content authoring interface based on the content authoring architecture that is defined in the project design document.
   - "The web content developer" on page 1762

- "Design architecture" on page 1776
- Developing

7. The website creator then creates some test content to test the content authoring system, and previews the test content. Any defects that are found in the content authoring system are addressed and further changes that are made to improve performance and usability.

Importing and creating content

When the authoring system is ready, you then import and create the default content for your system. This task is often managed by a separate team from the team that builds the web content authoring system.

Procedure

1. Based on data that is gathered during the analysis phase of the project, the content acquisition team identifies and prepares any existing content for import into the web content authoring system.
   - "The content acquisition team" on page 1764
   - "Content acquisition architecture" on page 1779
2. The content acquisition team then imports existing content into the web content authoring system by using features such as the Web Content Integrator, WebDAV, or the web content API.
   - "IBM Web Content Integrator" on page 1936
   - Developing
3. The content acquisition team then create new content based on the information architecture that is defined in the project design document.
   - "Information architecture" on page 1774
4. The content acquisition team then test the authoring system, and preview the default content. Any defects that are found in the authoring system are addressed and further changes that are made to improve performance and usability.

Deploying the delivery environment

The delivery environment is used to deliver your website to your website viewers. The delivery environment is deployed based on the requirements that are defined in the project design document.

Procedure

1. Based on the database architecture that is defined in the project design document, the database administrator does the following:
   a. Deploys a database server for the delivery environment.
   b. Clones the data that is stored on the authoring database onto the delivery database.
   - "Database" on page 119
   - "How to clone a web content repository" on page 1214
2. Based on the server architecture that is defined in the project design document, the WebSphere Portal administrator does the following:
   a. Installs a WebSphere Portal server or cluster of servers.
   b. Configures the WebSphere Portal server or cluster to use the database server setup by the database administrator.

c. Configures various WebSphere Application Server, WebSphere Portal and Web Content Manager configuration properties to ensure that the system is correctly setup for web content delivery and is tuned for optimal performance.

- Chapter 4, "Installing," on page 101
- Chapter 5, "Configuring," on page 231
- "Web Content Manager" on page 392

3. Based on the information architecture and security architecture that is defined in the project design document, the WebSphere Portal administrator:
   a. Create all pages that are required by the web content system.
   b. Adds all required web content viewer portlets to the appropriate pages.

      "Displaying content with Web Content Viewers" on page 2034
4. The WebSphere Portal administrator configures and enables syndication.

   "Syndication" on page 448
5. Final testing and tuning of the delivery environment is undertaken by all administrators.

Results

The delivery environment is now ready to use.

Going live with your website

When your environments are installed, the authoring system is completed, your default content is created, and fully tested the system, you are ready to go live.

Procedure

1. Ensure that your stakeholders are ready to use your website and authoring system:
   a. Ensure that your content authors are trained on how to use the new web content authoring system.
   b. Ensure that any legacy authoring or delivery systems are redirected to the new system.
   c. Start marketing campaigns to bring visitors to the new website.
2. Based on the maintenance architecture that is defined in the project design document, you run regular monitoring and maintenance tasks to ensure that your system is operating efficiently.
   - "Maintenance architecture" on page 1780
   - "Web content administration tools" on page 1176

Creating reusable assets

Use reusable assets to store or generate content that is used in more than one place in your website.

"Page templates" on page 1786
Content authors use the page templates to quickly create pages that are consistent with your site design. They do not have to waste time to configure settings that are probably consistent across your site, such as theme selection, page layout, and more.

"An overview of authoring templates" on page 1787
Authoring templates are like forms that content authors can use to create new

content. It defines default settings for the items that are created by using the authoring template. There are two types in authoring templates, site area and content.

"Presentation templates" on page 1789
You use a presentation template define the layout of your web content. Use tags to determine which properties, elements, or components are displayed.

"Template mappings" on page 1792
Template mappings are used to determine which presentation templates are used to display each site area or content item.

"Content items" on page 1794
Content items are created from authoring templates. A single content item can be used one time in the website or it can be reused in different areas of the website.

"Components" on page 1795
You use components to store elements that are used in more than one area of your website. For example, a company logo or a copyright notice.

Page templates

Content authors use the page templates to quickly create pages that are consistent with your site design. They do not have to waste time to configure settings that are probably consistent across your site, such as theme selection, page layout, and more.

Available page templates

Three templates are provided for immediate use: Articles, Basic, and Label. The Basic template is an empty page that you can add content to.

The Articles template has two portlets that display content, Articles and List of Articles. There is also a content association between the two portlets. The List of Articles portlet displays a list of the articles that are stored in the page site area. You can select an article from the list and it displays in the Article portlet. Both of the portlets were developed from the content viewer portlet.

The Label template is a simple label. Labels are navigation elements. The label displays in the navigation, but label type page does not include content. The content of the first child page displays when you click the label in the navigation.

Custom page templates

The provided page templates are probably not comprehensive enough for your website. Based on your html prototype or wireframe for your website, you can determine common page layouts and configurations. After you identify a few common page layouts from your design, then you can create templates for the content authors. For example, if each landing page in your site includes breadcrumbs, a carousel, and a three-column content area, then create a landing page template with those elements.

When you create a template you can include portlets and portlet preferences, page layout and style selections, theme and skin settings, portlet wires for communication with other portlets, page parameters, and page descriptions. Use the Manage Pages Portlet in the administration area to create new templates.

Page templates are created in a specific area of the portal hierarchy, called Page Templates. Pages that you add to this area automatically appear as an available page template in the site toolbar in the **Create** > **Page** area.

Note: When you create a new page based on a page template, you might see the following error message: `The page template could not be instantiated underneath the page PARENT_PAGE since this page has no default content association to a site area outside of the Portal Site library or the current user does not have access to the mapped site area.` To prevent this from happening, decide where to store the content that is copied from the template page to your new page:

- If you want to store the content outside of the Portal Site library, create a default content association to a site area outside of the Portal Site library for the parent page of the new page.
- If you want to store the content in the Portal Site library, transfer the content for the template page to the Portal Site library first. For more information, go to *Transferring content associations to the Portal Site library*.

Related tasks:

"Transferring content associations to the Portal Site library" on page 380
When you enable manage pages, any web content pages that you have are converted to managed pages and added to the Portal Site library. However, the content that is associated with the web content pages remains in the original libraries. You can transfer this associated content to the Portal Site library with the `internalize-content-mappings` task.

Related information:

Creating page templates

An overview of authoring templates

Authoring templates are like forms that content authors can use to create new content. It defines default settings for the items that are created by using the authoring template. There are two types in authoring templates, site area and content.

Create authoring templates for your content authors to use as they develop new content for the website. The authoring template is mapped to a presentation template. You can use one presentation template for multiple authoring templates. This removes presentation considerations and treatment from the content. The website design is updated by modifying the presentation template instead of modifying multiple pieces of content.

Authoring template types

Site area templates

Site area authoring templates are used to define the default settings of site areas.

Default site area template:

A default site area is installed with IBM Web Content Manager. This can be used to maintain the behavior of site areas that are migrated from previous releases. You can disable the creation of new site areas by using the default site area template by changing the following configuration parameter to "false" in the **WCM WCMConfigService** service by using the WebSphere Integrated Solutions Console. If the setting does not exist, it can be added.

```
wcm.authoringui.defaultSiteAreaTemplateEnabled=false
```

Content templates

Content authoring templates are used to define the default settings for content items. You create render content in two different ways:

Content rendering.

If you configure a content authoring template to render as content, then the content items you create are standard content items. These content items are used to store elements that can be rendered within presentation templates.

Resource rendering.

If you configure a content authoring template to render as a resource, then the content items you create are based on a file that is stored in a file resource element. When a resource content item is rendered, the file that is stored in the selected file resource element is rendered on the web page. No presentation template is used when the file is rendered, only the content of the file itself. This strategy is useful when you want to store a file, such as a PDF file, and render it directly on a page but would also like to have the PDF file that is listed in navigational components such as menus and navigators.

Authoring template properties

Element selection

When you create an authoring template, you can add elements to the template to determine what types of content is stored within the item. When you construct the authoring template, you can select more than one element field of the same element type. For example, you might add three text element fields, two rich text element fields, and four image element fields to the same authoring template.

Default values

You can specify default values for each field and element in the authoring template to make it easier and more efficient for an author to create new items and streamline the item creation process.

Simplified form layout

The authoring template provides features that help you simplify the presentation of the authoring form.

Authoring form layout options.

You can control the general layout of the fields on the authoring form by specifying an authoring form layout option. Depending on the layout option, this selection can reduce the vertical space that is required to display the elements on the authoring form.

Hidden fields.

In addition to organizing an authoring form with a layout option for the fields, you can further simplify the form that is presented to the item author by using hidden fields. Except for those fields that are required for an authoring form, you can designate any other field in the authoring template to be hidden. A field marked to be hidden in the authoring template is not displayed on the authoring form, which streamlines the form's visual appearance. Note, however, that although a hidden field is not displayed on the authoring form, the information that is defined in the field is still associated with the authoring form and is processed with the form.

This is useful when used with a default value for a field because it means you can specify a setting for a field and then hide the field on the authoring form to ensure that the field's value cannot be changed by the item author. For example, you might want to set access control levels for item that is generated from the authoring template in the Access Control section of the template and then hide that section on the resulting authoring form. When an item is generated from the template, the access control levels for the item are derived from the default values in the template.

Custom help text

To further help tailor the content form for an item author, Web Content Manager provides the capability of adding customized help text to the authoring template.

- You can define help text for the entire authoring form that is generated from the authoring template. For example, this help text can be used to describe the purpose of the form. You must include whatever specific information you feel would be of use to the authors by using the form.
- In addition to the HTML text, you can also specify in-line help text that is displayed with each element on the form. This help text can provide targeted information for a particular field on the form, explaining possible values or noting special conditions that are related to the field.

Labeling elements

The names of element labels in different items must be the same if an element reference in a presentation template is to change depending on the current context. This is an important consideration if two authoring templates are using the same presentation template. The element types however, do not have to be consistent.

Table 257. Example: An element reference called Heading from Current Site Area

Site area	Element label	Element type
Business	Heading	Image
Personal	Heading	Rich Text
Features	Heading	Text
News	Heading	Text

Presentation templates

You use a presentation template define the layout of your web content. Use tags to determine which properties, elements, or components are displayed.

Defining presentation for Web Content Viewer portlets

When you display content with WebSphere Portal, the presentation template defines only the layout of content that is displayed in a Web Content Viewer portlet. The overall page design is determined by which page layout is selected, which portlets are added to the page, and the selected theme.

Defining presentation for servlet rendered content

When you display content in a servlet delivered site, the presentation template represents the entire page. The presentation template also defines the default properties of a web page such as the background and default font of a web page.

There is little difference between building a presentation template and using HTML to build a servlet delivered web page. It might even be helpful to build a "mock-up" of the page you are designing in HTML before creating a new presentation template. Then replace the different sections of your web page with references to elements and components by using web content tags.

"Page layout"
You use HTML to define the layout of a presentation template in the same way you use HTML to define the layout of a web page.

"Page style" on page 1792
You use HTML to define the default properties of a presentation template in the same way you use HTML to define the default properties of a web page.

Page layout

You use HTML to define the layout of a presentation template in the same way you use HTML to define the layout of a web page.

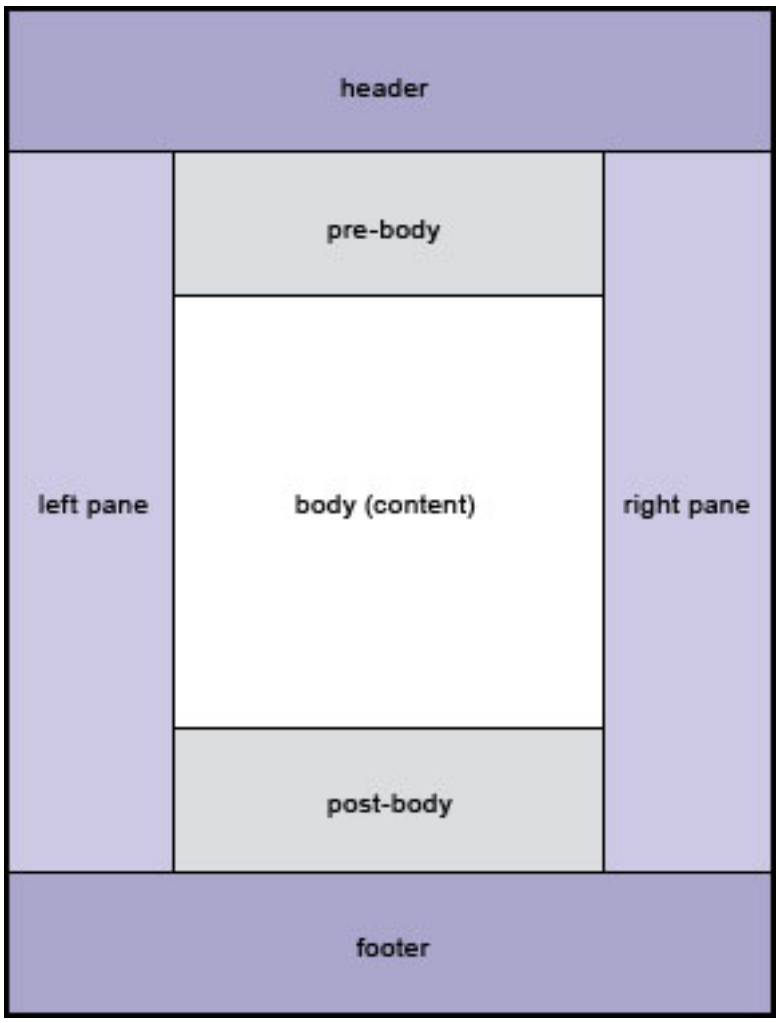
This is an example of a possible layout of a presentation template. Although it is recommended that HTML elements (such as tables) be used to specify the exact layout of a presentation template, you do not have to use them. You can lay out your page in any way you want.

When the layout of a page is defined, all you need to do is reference different components into the different sections of your HTML table. (You can reference more that one component within a single table cell.)

You can also enter text and HTML tags directly into a presentation template. This method is useful if you have an element that needs to appear on all pages that use a common presentation template. However, if that element is used on other presentation templates, it would be more efficient to save it as a component.

Example

This is an example of the HTML you might enter in a presentation template to set the layout of a presentation template.




```
<html>
<head></head>
<body>
<table width="100%" border="0"
cellspacing="0">
<tr><td colspan="3"></td></tr>
<tr><tdrowspan="3"></td><td></td><td
rowspan="3"></td></tr>
<tr><td></td></tr>
<tr><td></td></tr>
<tr><tdcolspan="3"></td></tr>
</table>
</body>
</html>
```

Text and IBM Web Content Manager tags are then added to the different table cells to create the finished web page.

Enabling Connect tags

Connect tags are advanced Web Content Manager tags that can be used to retrieve data from external sources and apply custom caching. **Process connect tags** must be selected in a presentation template form for connect tags to be processed.

Page style

You use HTML to define the default properties of a presentation template in the same way you use HTML to define the default properties of a web page.

Any valid HTML property can be set including:

- Margin sizes
- Text colors
- Background colors or images

Example

This is an example of the HTML you might enter in a presentation template to set default properties for a presentation template.

```
<html>
<head></head>
<body bgcolor="#CC0000" text="#000000" link="#6666FF"
vlink="#9999FF" alink="#FF33CC"leftmargin="5" topmargin="5">
</body>
</html>
```

Note: If the same page properties are used in more than one presentation template, they can be stored in a single text component that is itself referenced within the presentation template:

```
<body <component name="TextComponentName"/>>
```

This means that by editing a single text component, the page properties of multiple presentation templates can quickly be updated.

Cascading stylesheets

Default style properties cannot be set for components. The default page properties override any page properties set in a component.

Cascading stylesheets can be used to control the style of components. For example, You might make the links in a menu a different color to the links in a navigator by using cascading stylesheets to determine the style of different components.

Note: Where possible, use one cascading stylesheet for an entire site. A link to the stylesheet can be used, rather than embedding the stylesheet.

Template mappings

Template mappings are used to determine which presentation templates are used to display each site area or content item.

Template mapping strategies

The presentation template that is used by an item is determined by the relationship between the item's authoring template and a presentation template that is defined in the authoring template, or a template mapping that is defined in a site area in the path of the current item. Template mappings assigned in site areas override template mappings set in authoring templates.

This can result in the following relationships:

- A content item can be displayed by using two different presentation templates if linked to different site areas.

- Two items that use different authoring templates can be displayed by using the same presentation template if both authoring templates are mapped to the same presentation template.

Defined in authoring templates

If you select a default presentation template in an authoring template, it is used as the default presentation template for all items based on that authoring template. This ensures that all items based on that authoring template is rendered with the same presentation template, but it does not ensure design consistency between other items that are located in the same site area. If a different template mapping is specified in any site area in the item path, then the template mapping that is defined in the lowest part of the item path is used instead.

Defined in site areas

If you define a template mapping in a site area, this strategy ensures that all items based on the selected authoring template use the same presentation template in that site area. If a different template mapping is specified in any child site areas of the parent site area, then the template mapping that is defined in site area in the lowest part of the item path is used.

Template mapping examples

In these examples the following template mappings are used:

- *Authoring Template 1* uses *Presentation Template 1* as its default presentation template
- *Authoring Template 2* uses *Presentation Template 2* as its default presentation template
- *Authoring Template 3* also uses *Presentation Template 2* as its default presentation template
- *Authoring Template 4* has no default presentation template
- *Site Area 1* has no template mapping
- *Site Area 2* contains a mapping between *Authoring Template 1* and *Presentation Template 2*
- *Site Area 1* and *Site Area 2* are located under *Site Area A*.
- *Site Area A* contains a mapping between *Authoring Template 4* and *Presentation Template 3*

The presentation template that is used by each item is determined by the authoring template the item used, and the location of the item in the site framework.

Table 258. Template Map Results

Content and location	Result
Content 1 using *Authoring Template 1* located in *Site Area 1*	As *Site Area 1* contains no template mappings, *Content 1* is displayed by using *Presentation Template 1* that is the default presentation template of *Authoring Template 1*.
Content 1 using *Authoring Template 1* located in *Site Area 2*	As *Site Area 2* contains a mapping between *Authoring Template 1* and *Presentation Template 2*, *Content 1* is instead displayed by using *Presentation Template 2*.

Table 258. Template Map Results (continued)

Content and location	Result
Content 2 using *Authoring Template 2* located in *Site Area 1*	As *Site Area 1* contains no template mappings, *Content 2* is displayed by using *Presentation Template 2* that is the default presentation template of *Authoring Template 2*.
Content 3 using *Authoring Template 3* located in *Site Area 1*	As *Site Area 1* contains no template mappings, *Content 3* is also displayed by using *Presentation Template 2* that is the default presentation template of *Authoring Template 3*.
Content 4 using *Authoring Template 4* located in *Site Area 1*	As *Site Area 1* contains no template mappings, *Content 4* is displayed by using *Presentation Template 3* that is mapped to *Authoring Template 4* in *Site Area A*.

Element references

When referencing elements in a presentation, it is important to note the following:

- The elements a presentation template uses must be defined in the authoring template the content is based on.
- If the element being referenced does not exist in the current item, nothing is displayed in that section of the presentation template.

Although the template author can identify a number of elements that can be displayed on the item form, whether the elements are displayed depends on the presentation template that is used with the authoring template to render the content form. A presentation template might not include every element that is defined in an authoring template, but in order for an element or element type to be available to a presentation template, the element must be included in the authoring template that is used to create the content.

Content items

Content items are created from authoring templates. A single content item can be used one time in the website or it can be reused in different areas of the website.

If you have content that you want to appear on multi-pages, create a content item. Then link the content item to the different areas in the website where the content needs to appear. This approach ensures consistency and makes changing the content items easier.

The authoring template can determine what element types are included in a content item, whether a workflow is used or not and what workflow to use, and where the content item can be saved.

Each content item that you create is the equivalent to a web page in a traditional website. Unlike traditional websites, a single content item can be linked to different areas within your website, or linked to a different website altogether. The changes that you make to a single content item is visible in every place you link the content item to.

The look and feel of a content item when displayed in a website depends on what authoring template was used to create the content item, and what presentation

template is used to display the content. The presentation template that is used depends on the current context of the content item, and which template mapping applies to the current context of the content item.

Components

You use components to store elements that are used in more than one area of your website. For example, a company logo or a copyright notice.

Static components

Static components are used to store static content such as text, files, or images.

- component reference component
- date and time component
- file resource component
- HTML component
- image component
- JSP component
- link component
- number component
- rich text component
- short text component
- style sheet element
- text component
- user selection component

Dynamic components

Dynamic components are used to dynamically generate content based on the parameters set in the component properties.

- menu component
- navigator component
- Personalization component
- taxonomy component
- user name component

Tool components

Tool components are used to create tools that can be added to web pages for users to do tasks such as search, inline editing, and paging through pages of links.

- authoring tools component
- page navigation component
- search component

Building the website

Building a website includes creating libraries, pages, and web content. You also create links and navigation to your content, and assign access to different types of users, such as editors and viewers.

"Site Builder" on page 1797
Site Builder is an application that is used to create site templates and section

templates. You then use Site Builder to quickly roll out new websites, or add new sections to existing websites, using these site templates and section templates.

"Content Template Catalog" on page 1797
The Content Template Catalog (CTC) is a set of templates that accelerate the process of building a website. You should also install the Content Template Catalog with Site Builder so you can build sites based on Content Template Catalog components.

"Content libraries" on page 1798
Content libraries store the assets for your website, including but not limited to pages, content, images, authoring and presentation templates, and workflows.

"Setting up access" on page 1801
Access controls allow you to assign access to who has the ability to view rendered constant and pages, and who has the ability to edit or administer content, pages, or features.

"Taxonomy" on page 1801
Before creating a taxonomy, you should analyze how the taxonomy will be used in your site to determine the best structure for your taxonomy.

"Navigation" on page 1802
Navigation is defined by your page hierarchy and your site area hierarchy. Navigation elements include your page theme, menus, and navigators.

"Elements" on page 1804
Create a website by, building templates, and specifying other key components of the system. When these items are in place, you can begin adding elements to your templates and use them to create content items.

"Tags" on page 1833
Tags are used in your markup to reference content that is stored or generated by elements, or to display metadata from different items.

"Setting up search for site visitors" on page 1846
Learn about the initial considerations for setting up search for site visitors before you begin configuring and administering search on your website.

"Taxonomies, Categories, and keywords" on page 1847
The combination of taxonomy and categories enables control what displays in menus.

CF07 "How to store translated text in a content item or site area" on page 1848
Translated text can be stored in a content item or site area. The translated text can then be referenced in web content tags, or as localized text in web content authoring portlet forms.

"Hiding content" on page 1850
You can hide portlets and widgets on a page so they are present but not visible in the page layout.

"Static content" on page 1851
Static content is part of every website. In a portal site, static content can be rendered as static page or it can be added to specific content areas on a page.

"Dynamic content" on page 1880
Dynamic content is generated when a page is rendered and is based on a set of predefined criteria such as the current user, the metadata of the current page. It allows you to deliver content specifically customized for the current user, or to deliver content customized for a campaign or product in your organization.

CF05 "Content as a Service pages" on page 1881
Starting with version 8.5 CF05, IBM WebSphere Portal Express introduces the

concept of Content as a Service pages. The Content as a Service pages can be used to render content that is managed by your IBM Web Content Manager in different data formats.

"URLs" on page 1886
Portal URLs do not look like plain HTTP server URLs over a simple file system. Portal URLs have a complex structure and include a large compressed and encoded XML Navigation State document. The stream of random characters in a Portal URL is the Navigation State document. Full Portal function depends on correctly maintaining this Navigation State document during all the operations a user might do in Portal. Forcing Portal URLs to look like plain HTTP server URLs over a simple file system structure cripples the function of WebSphere Portal Express.

Site Builder

Site Builder is an application that is used to create site templates and section templates. You then use Site Builder to quickly roll out new websites, or add new sections to existing websites, using these site templates and section templates.

Installing Content Template Catalog and Site Builder together offers you the widest choice of website designs. After you install Site Builder and set up access for administrators and website creators, they can log in to IBM WebSphere Portal Express and go to the Site Builder page to access the features of Site Builder.

After you create a site with Site Builder, you can use IBM WebSphere Portal Express and IBM Web Content Manager to customize it further. For example, you can add pages and portlets, edit content, set up content targeting, and refine author and user access.

Related information:

Site Builder

Content Template Catalog

The Content Template Catalog (CTC) is a set of templates that accelerate the process of building a website. You should also install the Content Template Catalog with Site Builder so you can build sites based on Content Template Catalog components.

You can use Content Template Catalog to mock up a site quickly or to build the skeleton of a site as a basis for further development. By using the templating patterns that Content Template Catalog is built on, you can supply a customized system for your users that lets them build their own sites.

To get started, install Content Template Catalog on a development server and try customizing copies of the templates. Content Template Catalog templates are designed to allow your own templates, components, workflows, and other item types to slot into the framework easily so they can be used in conjunction with out-of-the-box Content Template Catalog assets. Experimenting with the templates will give you a good working knowledge of Content Template Catalog for your own site development. Building a prototype as you start using Content Template Catalog can drive the design process or be a starting point for development.

The Content Template Catalog templates cover many basic content types and design elements. Using these templates you can build a basic site very quickly

with little or no customization. For more complex sites, Content Template Catalog allows you to rapidly move past the initial build phase and proceed to delivering real value to your customer.

Related information:

Content Template Catalog

Content libraries

Content libraries store the assets for your website, including but not limited to pages, content, images, authoring and presentation templates, and workflows.

All pages are stored in a Portal Site library. The Portal Site library is immediately available and includes the welcome page, getting started page, and more. Create more libraries to store your website content and assets. You can create multiple libraries to organize your content or to reflect your companies structure. You might create a library for a specific department that requires different access control. In most cases you need to create a minimum of two libraries:

1. A design library where you store all the items that are required for the web content system itself
2. A content library that is used to store the content that is developed by your content creators

The access control and syndication configuration is unique for each library that you create.

Note: When you search through content libraries, you can search for items by their titles. However, if you sort items by title, the sorting option filters pages by the Unique name or Identifier of the page, and not by the title that you specified in the page settings.

New libraries and default items

When you create a new library, you can select to include default items. The default items are helpful if you are not familiar with standard library assets such as authoring templates, presentation templates, and workflows. Include the default items to see examples that you can customize and use.

To create a new library, Click the **Administration menu** icon. Then, click **Portal Content** > **Web Content Libraries**.

Important: When you create a web content library, ensure that the name of the library does not match the URL context of any virtual portals on the same server. If the name of a library and the URL context of a virtual portal have the same value, incorrect rendering behavior can result.

When to use the Portal Site library with web content

You can store web content in the IBM WebSphere Portal Express site library or in a separate content library:

- For dedicated page content that is created and displayed on one single page, it is more convenient to store the web content in the Portal Site library.
- For web content that is reused in multiple places or not tied to the Portal Site hierarchy, it is more convenient to store the web content in a separate dedicated content library.

Storing web content in the Portal Site library

- This scenario is easy to set up, but means that the content cannot be syndicated, exported or imported separately.
- Managed pages must be enabled.
- The web content is moved or deleted when a page is moved or deleted.

Storing web content in dedicated content libraries

- This scenario is easy to deploy if the content associations are set up correctly. The content can be syndicated, and exported or imported separately.
- The web content is associated with a page.
- The web content is not moved or deleted when a page is moved or deleted.
- Web content can be stored in separate libraries. This way, you can logically organize your content, and setup separate syndication scenarios.

Note: When you store web content in dedicated content libraries, maintain the site area hierarchy so that it exactly matches the page hierarchy. For example, when you create a new page, also create a site area at the same point in the hierarchy, and associate it with the new page that you created. Similarly, when you rename, move, or delete pages, update, and synchronize the associated site areas accordingly.

"Creating libraries"
Web content libraries are used to store your web content and managed pages. Syndication is used to keep libraries in synch between your different server environments.

"Web content library default items" on page 1800
When you create a web content library, you can choose to include a set of default web content items in the new library. These items can be used as a starting point for your Web Content Manager system and website.

Related concepts:

"IBM Web Content Manager Multilingual Solution" on page 2442
The Web Content Manager Multilingual Solution is a set of tools that are used to manage translated versions of localized and regionalized websites.

Related information:

Create a content library

Creating libraries

Web content libraries are used to store your web content and managed pages. Syndication is used to keep libraries in synch between your different server environments.

About this task

When setting up a website, you should only create multiple libraries when:

- You need to manage the access and security of different types of content for different users and groups in different ways. For example, by creating different libraries for design elements and content elements, different groups could be assigned library level access in different ways on each library,
- You need to manage syndication strategies differently for different types of content. For example, by creating different libraries for design elements and content elements, different syndication strategies are applied to each library. The

content library could be assigned automatic syndication, whereas the design library could be manually syndicated only when it was appropriate to do so.

Procedure

1. Click the **Administration menu** icon. Then, click **Portal Content** > **Web Content Libraries**.
2. Click **Create new library**.

Web content library default items

When you create a web content library, you can choose to include a set of default web content items in the new library. These items can be used as a starting point for your Web Content Manager system and website.

Default items

The following items are created when you select **Include default items in the new library** when you create a library.

Workflow items:

- A workflow that is named **Express Workflow** with a single workflow stage named **Publish Stage** by using the publish action named **Publish**.
- A workflow that is named **Three Stage Workflow** by using the following workflow stages:
  - **Draft Stage**
  - **Publish Stage** by using the publish action named **Publish**.
  - **Expire Stage** by using the expire action that is named **Expire**.

Authoring template:

The authoring template is named **Article** and contains a single rich text element that is named **Body** and uses the **Express Workflow** as the default workflow for content items that are created by using this authoring template.

Presentation template:

The presentation template is named **Article Presentation**.

Site area and content items:

- The content items that are named **Sample Article** and **Sample Article 2** are stored in the site area named **Articles**.
- The site area that is named **Articles** contains a template map between the authoring template that is named **Article** and the presentation template named **Article Presentation**.

Components:

- The authoring tool that is named **Article Toolbar** is used to add **New** and **Edit** functions to the rendered page. It is referenced in the presentation template named **Article Presentation**.
- The menu that is named **Articles List** is used to display a list of content items on the rendered page. It is referenced in the presentation template named **Article Presentation**.

Access controls

As the web content library default items are configured to inherit their access settings from the library they are stored in, users are not able to access these items until you configure the access settings of the library.

Default items

The default items are best displayed by using a web content viewer portlet:

1. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
2. Select **New Page From**.
3. Under **Web Content Mappings**, select the site area that is named **Articles** from your web content library.
4. Complete the rest of the form and click OK.
5. Edit the page layout of the new page and add a **Web Content Viewer** portlet to the page.
6. The content item that is named **Sample Article** is displayed on the page.

Setting up access

Access controls allow you to assign access to who has the ability to view rendered constant and pages, and who has the ability to edit or administer content, pages, or features.

About this task

Setting up access is covered in more detail in the Administering and Securing sections of the documentation, but there are some things to consider when setting up access for a website.

- Create groups for each type of user of your site. For example, if you have a set of users who need edit access to a site, create an Editors group and assign your editors to this group.
- Use library inheritance so that access is automatically inherited down to each item in a library. Inheriting permissions, instead of explicitly defining access rights on individual items, simplifies access rights maintenance and improves overall system performance. Define role assignments as high as possible in the library so that they apply to the largest set of items. Explicitly defined permissions are a powerful and flexible way to control access to an item, but when the same user or group is being granted the same role on all items within an area, or across an entire library, then inherited permission is the best option.
- Use Workflow security to assign access to groups or users at different stages in a workflow cycle.

Taxonomy

Before creating a taxonomy, you should analyze how the taxonomy will be used in your site to determine the best structure for your taxonomy.

Creating a taxonomy

To create a taxonomy:

1. Open the applications menu and click **Content** > **Web Content Authoring**.
2. Create a top-level taxonomy item.
3. Create child categories for the taxonomy item.

You cannot use taxonomies in menu searches. If you need a menu to return results based on content that is profiled with any category in a taxonomy, create a single top-level category. Then, and base the menu on the top-level category.

Additional classification options

Classifying users

You can add classification information to users. This can be used as search parameters in menu elements.

Classifying rendering portlets

You can add classification information a Web Content Viewer Portlet. This can be used as search parameters in menu elements.

Classifying content versus access controls

By classifying content with categories or keywords, you can personalize a website for different users. This is different from using item access controls to limit what items a user can access. In a personalized site, although a user may not be able to access all the pages via personalized menus, they may still be able to access other pages by using navigators, or by searching for content. Using access controls limits a user to only view items that they have been granted access to.

Related information:

Authoring portlet: Creating a taxonomy

Navigation

Navigation is defined by your page hierarchy and your site area hierarchy. Navigation elements include your page theme, menus, and navigators.

Hierarchical navigation

The hierarchical structure of your website is displayed to users in different ways.

Page navigation

The top levels of navigation are defined by your page hierarchy, and are automatically displayed by the default portal theme as links. The location of these links is determined by the page layout you select.

Site area navigation

If your website includes web content, extra navigational links are displayed with a navigator component. This component type displays a subsection of the hierarchical structure of the site areas and content items that are used by your website.

Criteria based navigation

Use menu components to display lists of links to your web content based on predefined or dynamic criteria. The parameters that are defined in a menu component can be based on set criteria, such as a category or authoring template, or they can be set with dynamic criteria that are based on the current content item, site area, or other variables.

"Creating links and navigation" on page 1803
Use navigators and menus to create links to content, and display content-level navigation.

"Adding page-level navigation" on page 1803
The default portal theme automatically adds page-level navigation to your site based on the page hierarchy of your site.

"Hiding and displaying pages in the navigation" on page 1803
By default, pages that you create are displayed in the navigation of the portal

site. If you do not want a page that you create to appear in the navigation, you can hide the page by setting the **com.ibm.portal.Hidden** page parameter to `true`. While this parameter does not affect your portal access control settings for the page, it is hidden from the navigation.

Creating links and navigation

Use navigators and menus to create links to content, and display content-level navigation.

About this task

Navigators

Navigators display links to different site areas in a website. They represent the logical arrangement of the content displayed within each page of your site.

A set of element designs is used to format the look of each branch of a navigator.

Menus

A menu displays metadata and content from content items that match the search criteria of the menu element. The search criteria of a menu can include matching site areas, authoring templates, categories, and keywords.

Adding page-level navigation

The default portal theme automatically adds page-level navigation to your site based on the page hierarchy of your site.

About this task

Page-level navigation is added to static pages using these portlets:

The Breadcrumb Trail portlet

The Breadcrumb Trail portlet allows you to add the navigation path to your static pages. The breadcrumb trail starts at the content root and goes down to the currently selected static page.

The Navigation portlet

The Navigation portlet allows you to add dynamic portal navigation to a static page. When writing a static portal page you can configure the starting point of the navigation and the number of navigation levels by using the initialization parameters.

Hiding and displaying pages in the navigation

By default, pages that you create are displayed in the navigation of the portal site. If you do not want a page that you create to appear in the navigation, you can hide the page by setting the **com.ibm.portal.Hidden** page parameter to `true`. While this parameter does not affect your portal access control settings for the page, it is hidden from the navigation.

About this task

As a page editor, you might want to access hidden pages by using the navigation. For example, to use the site toolbar to edit a page, you need to go to the page first. You can either use a direct URL to the hidden page or make hidden pages display as described in the following.

Procedure

- To hide hidden pages from the navigation, select **Menu** > **Hide Hidden Pages** from the site toolbar.
- To make hidden pages display in the navigation, select **Menu** > **Show Hidden Pages** from the site toolbar. Hidden pages are displayed in brackets in the navigation. For example, `[Hidden Page]`.

Related reference:

"Marking pages as hidden under the content root" on page 1107
By default, pages that you create under the content root display in the main menu. If you do not want a page that you create to appear in the main menu, you can hide the page.

Elements

Create a website by, building templates, and specifying other key components of the system. When these items are in place, you can begin adding elements to your templates and use them to create content items.

"Links and navigation" on page 1805
You use these elements to define or generate links between different pages in a website, or add navigational elements to a website.

"How to use a search element" on page 1807
A search element is used to display the results of a search query. A search element cannot be used in isolation, but must be used together with an HTML element that is used to define the search query form.

"How to store text and HTML" on page 1811
You use different types of elements to store text or HTML depending on the type of text or HTML being created.

"How to store files and images" on page 1814
You use these elements to store files or images.

"Selection elements" on page 1817
You use these elements to make selections from existing elements and data.

"Personalized content" on page 1819
You use these elements to create or reference personalized content.

"Page navigation element" on page 1826
A page navigation element provides navigation controls that are used to browse through a set of results that are generated by menus, navigators, personalization, and search elements.

"URL generation by using `PathCmpnt` and `URLCmpnt` tags" on page 1827
There are some special considerations to keep in mind when you use `URLCmpnt` and `PathCmpnt` tags to create URLs to other web content items from within your content.

"Personalizing federated documents" on page 1827
Portal Personalization provides the federated documents feature to retrieve metadata about documents that are stored in external content management systems or document repositories. Examples of these systems include IBM Content Manager, IBM FileNet Content Manager, and Microsoft Sharepoint. You can use a personalization component in IBM Web Content Manager to display metadata from federated documents and to create links to download or open the documents.

Related information:

How to work with elements

Element designs

Links and navigation

You use these elements to define or generate links between different pages in a website, or add navigational elements to a website.

"Link element"
A link element stores a link to a web content item, or to external content such as a web page.

"Menu element"
A menu element displays metadata and content from content items that match the search criteria of the menu element. The search criteria of a menu element can include matching site areas, authoring templates, categories, and keywords.

"Navigator elements" on page 1807
A navigator element displays metadata and content from a predefined section of a site framework, usually in the form of links.

Link element:

A link element stores a link to a web content item, or to external content such as a web page.

Most web pages contain links, either to other web pages, or to files. In a web content site, most links are generated by using navigator and menu elements. A link element stores a link that is not part of a site's navigation. For example, you can store a link to your "home" content item in a link element. You then add a reference to the link element in every presentation template that is used by your site to enable users to return to the "home" content item. If you want to change your "home" content, you change the selected item in the link element.

Creating a link element

To create a link element, you can either add a link element to an authoring template, site area or content item, or create a link component.

Menu element:

A menu element displays metadata and content from content items that match the search criteria of the menu element. The search criteria of a menu element can include matching site areas, authoring templates, categories, and keywords.

Creating a menu element

You can use a menu element only by creating a menu component. You cannot add a menu element to authoring templates, site areas, or content items.

Menu search options

Menu element search options are defined in the **Menu Component Query** section of the menu element form. These search options define which content items from your site are displayed in the menu element. Search options can include a combination of search parameters that include searches based on authoring templates, categories, and site areas.

Menus can search for the following content in a website:

- Content with matching authoring templates
- Content with matching site areas

- Content with matching categories
- Content with matching keywords

Between different criteria, menu searches are "and" searches, but within each search criteria, menu searches are "or" searches. For example, a menu element that searches for two different categories and an authoring template displays content items that are profiled with at least one of each profile type. Content that matches only one profile type is not displayed.

Menus do not display search results if you select a search criteria but do not enter any search parameters. For example, if the menu is configured to display results that are based on categories, but no categories are specified in the menu form, then no matches are displayed.

Menu sorting options

You can sort menu search results according to following criteria:

- Content document name
- Content document description
- Published date
- Expired date
- General date one
- General date two
- Last modified date

You can select a maximum of three sort options.

Menu paging options

IBM Web Content Manager provides flexible paging options that are used to display search results that are generated by the menu element.

- You can specify the number of results that are displayed in a menu page. For example, a menu that is defined to show five results per page would display only five records from the set of search results.
- You can indicate where in the results set you want to begin showing results by specifying which menu page to use as a starting point. As an example, if you are displaying five results per menu page and you want to show records 6–10, you would start showing search results with the second menu page instead of the first.
- To provide easier navigation of the search results in a menu, you can include a page navigation element in the header or the footer of the menu element. The page navigation element enables stepping forward or backward through multiple menu pages without the need for creating multiple menu elements to display the different pages.
- A large number of search results can cause a delay when the menu element is initially rendered. To prevent this delay, you can limit the maximum number of pages of results that are included in the menu. To further improve the efficiency of the menu, you can also specify how many pages of results must be read beyond the current page so that paging performance is not affected by rebuilding the menu.

While a page navigation element is a convenient way of displaying and browsing a menu's search results, you can use the menu's paging options to display search

results in other ways. For example, if you wanted to show the results in a 3-column table, you might create three menu elements with the same search criteria and then tailor the paging options of each menu to display different result sets:

Table 259. Example

Menu	Results per page	Start page	Records displayed
Menu element 1:	5	1	1 to 5
Menu element 2:	5	2	6 to 10
Menu element 3:	5	3	11 to 15

The three menus might then be referenced within three different cells of a table row in a presentation template.

Navigator elements:

A navigator element displays metadata and content from a predefined section of a site framework, usually in the form of links.

Navigators are not menus. Menus are a list of hyperlinks that take you to specific pages. Navigators can also be hyperlinks that can take you to specific pages but navigators are organized differently. Navigators present the logical arrangement of a website whereas menus are a list of related web pages in your website.

The navigator element is configured by selecting a start area and determining a child depth, a parent level, and a sibling value relative to the start area. Possible start areas are site areas, or content items.

There are also options to determine whether the start area is to be displayed, if content items are to be displayed, and if the hierarchy from the start area to the current site area is expanded.

A set of element designs is used to format the information for each branch of a navigator.

Navigators display links to different site areas in a website. As such, each site area in a website must have a default content item. Otherwise, some links in a navigator will not work.

Creating a navigator element

You can only use a navigator element by creating a navigator component. You cannot add a navigator element to authoring templates, site areas, or content items.

How to use a search element

A search element is used to display the results of a search query. A search element cannot be used in isolation, but must be used together with an HTML element that is used to define the search query form.

To create a search form, you must:

1. create a search query by using an HTML element
2. create a search results view by using a search component

You reference both the HTML element and search component in a single presentation template. The search component is only rendered after a search query is run by a user.

"Search query examples"
These are examples of search queries you can create by using an HTML element.

"Search result examples" on page 1810
These are examples of how to design your search results.

Search query examples:

These are examples of search queries you can create by using an HTML element.

The query parameters can be set in the request parameters. For instance, a search component that is displayed on a page whose URL contained a search query `?search_query=shoes` displays search results for shoes.

Simple search query

This is an example of a simple search query form:

Table 260. Simple search query

Code example	Description
`<form action='<PathCmpnt type="servlet" /> /library/sitearea/content' method="post">`	This is the form header where you specify the location of the content item containing the search element that is used to display the search result. This is typically the same content item that this HTML element is stored in.
`<table> <tr><td> <input type="text" name="search_query"/> </td></tr> <tr><td align="right"> <input type="submit" value="Search"/> </td></tr> </table>`	This is the body of the search form. Like any standard HTML form, it contains an input field and a submit button. In this example, a table has been used to format the search query form.
`</form>`	This closes the form.

Searching metadata

In this example, two more fields have been added allowing users to search both content title and author name:

Table 261. Searching metadata

Code example	Description
`<form action='<PathCmpnt type="servlet" /> /library/sitearea/content' method="post">`	This is the form header where you specify the location of the content item containing the search element that is used to display the search result. This is typically the same content item that this HTML element is stored in.
`<table> <tr> <td>Content Title</td> <td><input type="text" name="search_title"/></td> </tr> <tr> <td>Author's Name</td> <td><input type="text" name="search_authors"/></td> </tr> <tr> <td>Content Body</td> <td> <input type="text" name="search_query"/> </td></tr> <tr><td align="right"> <input type="submit" value="Search"/> </td></tr> </table>`	This is the body of the search form. Like any standard HTML form, it contains input fields and a submit button.
`</form>`	This closes the form.

Including hidden data

In this example, a hidden field has been added to restrict the search to content that use the authoring template called "Press Release":

Table 262. Including hidden data

Code examples	Description
`<form action='<PathCmpnt type="servlet" /> /library/sitearea/content' method="post">`	This is the form header where you specify the location of the content item containing the search element used to display the search result. This is typically the same content item that this HTML element is stored in.

Table 262. Including hidden data (continued)

Code examples	Description
`<input type="hidden"` `name="search_authoringtemplate"` `value="Press Release"/>`	Here a hidden input field has been added that searches for content that use the authoring template called "Press Release".
`<table>` `<tr>` `<td>Content Title</td>` `<td><input type="text" name="search_title"/></td>` `</tr>` `<tr>` `<td>Author's Name</td>` `<td><input type="text" name="search_authors"/></td>` `</tr>` `<tr>` `<td>Content Body</td>` `<td>` `<input type="text" name="search_query"/>` `</td></tr>` `<tr><td align="right">` `<input type="submit" value="Search"/>` `</td></tr>` `</table>`	This is the body of the search form. Like any standard HTML form, it contains input fields and a submit button.
`</form>`	This tag closes the form.

Setting the search query in the request attributes

The query parameters can also be set in the request attributes on the server. For instance, a search component that is displayed after the following tag is used: `[Plugin:RequestAttribute key="search_query" value="shoes" compute="once"]` displays search results for shoes.

Search result examples:

These are examples of how to design your search results.

Search element design example for use in a website

In this example, a table is used to lay out the search results.

Table 263. Search element design example for use in a website

Design field	Details	Code example
Header		`<table>`
Result	The attributes to display in each search result are defined here.	`<tr><td>` `<attributeResource attributeName="namelink"/> ` `<attributeResource attributeName="summary"/>` `</td></tr>`
Separator	A separator can be used to delineate each search result.	`<tr><td bgcolor="#FFFAA" colspan="2"/></tr>`

Table 263. Search element design example for use in a website (continued)

Design field	Details	Code example
Footer	A page navigation element that is stored in a component is referenced here to add page navigation to the search results.	`<tr><td>` `<component name="pagenavigationcomponent"/>` `</td></tr>` `</table>`
No results		There are no results for your query. Please refine your search and try again.

Search element design example for use in a rendering portlet

In this example, a table is used to lay out the search results.

Table 264. Search element design example for use in a rendering portlet

Design field	Details	Code example
Header		`<table>`
Result	The attributes to display in each search result are defined here. When displaying search results in a rendering portlet, you must specify the page that the linked content is displayed in when opened. A URL map to the portal page that contains the rendering portlet is required.	`<tr><td>` `<a href="/[PORTAL_CONTEXT_ROOT]/` `[PORTAL_PAGE_URL_MAPPING]/?WCM_GLOBAL_CONTEXT=` `<AttributeResource attributeName="url" />">` `<AttributeResource attributeName="title" /></a>` `<attributeResource attributeName="summary"/>` `</td></tr>`
Separator	A separator can be used to delineate each search result.	`<tr> <td bgcolor="#FFFAA" colspan="2"/> </tr>`
Footer	A page navigation element that is stored in a component is referenced here to add page navigation to the search results.	`<tr><td>` `[component name="pagenavigationcomponent"]` `</td></tr>` `</table>`
No results		There are no results for your query. Refine your search and try again.

How to store text and HTML

You use different types of elements to store text or HTML depending on the type of text or HTML being created.

Tip: Text that is common to all content that uses a single presentation template can be entered directly in the presentation template rather than separate elements.

Text, rich text and HTML elements:

You use the short text, text, rich text, and HTML elements to store blocks of text, but each has slightly different properties.

Creating an element

Short text, text, rich text, and HTML elements can be added to site areas, content items, and authoring templates or they can be created as individual components.

Short text element

A short text element is used to store small amounts of fixed-length text where the length is 250 bytes or less. Unlike the other text elements, short text elements can also be used as a search parameter in a Personalization rule.

Text element

You use a text element to store larger amounts of text than can be stored in a short text element. No special processing occurs for this element.

HTML element

An HTML element is used to store fragments of HTML that can be reused in presentation templates and other elements designs. You can enter HTML directly into the element or upload HTML from a previously created HTML file.

Rich text element

A rich text element is similar to the HTML element except that it includes a rich text editor. The rich text editor can be used to format text that is stored within a rich text element. The main purpose of the rich text element is to provide base-level content creators with an easy-to-use text editor. Advanced users who are required to produce more advanced code, including web content tags, or who need to store fragments of HTML must use an HTML element instead.

You must use rich text elements sparingly in authoring templates, site areas, and content items as adding multiple rich text elements to these items can reduce authoring performance.

The rich text editors that are used by Web Content Manager are supplied by other vendors. For information on using the rich text editor, see the user documentation that is supplied by the specific rich text editor vendor.

How to use web content tags in rich text and HTML elements

Table 265. How to use web content tags in rich text and HTML elements

Element type	Details
Short Text and Text elements	Web content tags cannot be used in short text and text elements.
HTML elements	Any combination of web content tags can be used in HTML elements with the following exceptions: 1. You cannot use single quotation marks around attribute values. • `[Component name='example']` • `[Component name='example' start='<a href="' end='">link</a>']` • `[Component name='example' start='<img src="' end='"/>']` 2. You cannot use double quotation marks inside attribute values. • `[Component name="example" start="<a href="" end="">link</a>"]` • `[Component name="example" start="<img src="" end=""/>"]`
Rich text elements	Basic Web Content Manager tags can be used in rich text elements. For example, the following tags can be used in Rich Text elements: • `[component name="test"]` • `[element type="content" context="current" key="body"]` The following tag formats are invalid: 1. The use of single quotation marks around attribute values. • `[Component name='example']` • `[Component name='example' start='<a href="' end='">link</a>']` • `[Component name='example' start='<img src="' end='"/>']` 2. The use of double quotation marks inside attribute values. • `[Component name="example" start="<a href="" end="">link</a>"]` • `[Component name="example" start="<img src="" end=""/>"]` 3. Embedding tags inside other HTML tags. • `<a href='[Component name="example"]'>link</a>` • `<img src='[Component name="example"]'/>`

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Number element:

You can store a numerical value in a number element.

Creating a number element

To create a number element, you can either add a number element to an authoring template, site area or content item, or create a number component.

How to store files and images

You use these elements to store files or images.

"File resource element"
You can store a file in a file resource element. You can then reference the file resource so that users can download the file by using a link or, for supported file types, convert the file directly to HTML and render it on the page.

"How to reference a file resource"
A file resource can be referenced within presentation templates and other element designs with either a component or element tag.

"Image element" on page 1815
You store an image in an image element.

"JSP elements" on page 1816
You use a JSP element to store a path to a JSP. When rendered within a presentation template or element design, a request to a JSP is generated and processed.

"How to use stylesheets in items" on page 1816
Stylesheets can be used to format the design of IBM Web Content Manager pages in the same way as normal web pages.

File resource element:

You can store a file in a file resource element. You can then reference the file resource so that users can download the file by using a link or, for supported file types, convert the file directly to HTML and render it on the page.

By storing a file in a file resource element:

- You can reference the file anywhere in your site regardless of the original file location or URL.
- The file is automatically transferred to other servers during syndication.

Creating a file resource element

To create a file resource element, you can either add a file resource element to an authoring template, site area or content item, or create a file resource component.

Using a resource content item versus using a file resource component:

If you configure an authoring template to be a resource template, the content items that are created by using that authoring template are known as resource content items. Like a file resource component, files that are stored within a resource content item can be directly rendered in a web page. The difference between using a resource content item instead of a file resource component is that a resource content item can be listed in navigational components such as menus and navigators.

How to reference a file resource:

A file resource can be referenced within presentation templates and other element designs with either a component or element tag.

Creating a link to a file resource

To create a link to a file resource in a presentation template or element design to enable users to download the file reference, use the following tag structures.

To create a link to a file resource component, you use a component tag:

```
<a href="[component name="FileResourceName"]">Link Text</a>
```

To create a link to a file resource element, you use an element tag. For example, to link to a file resource element in the current content item:

```
<a href="[element type="content" context="current" key="FileResourceName"]">Link Text</a>
```

Rendering a file resource on a page

If your file resource is a file type that can be converted to HTML, you can instead convert the file to HTML and render the converted HTML directly in your web content with the *format="HTML"* parameter in a component or element tag. For example:

```
[component name="FileResourceName" format="HTML"]
[element type="content" context="current" key="FileResourceName" format="HTML"]
```

Examples of supported file types include:

- word-processing documents (*.doc, *.odt)
- spreadsheets (*.xls) *
- HTML files (*.htm, *.html)
- Text files (*.txt)

Other file types might also work but you need to test them first.

Note: If you configure an authoring template to be a resource template, then the content items you create are resource content items. When a link to a resource content item is rendered, the file that is stored in the selected file resource element is rendered on the web page. In this case, you would not use an element tag. Instead, the file resource is rendered either with a placeholder tag in a navigator or menu design, or by writing a link directly to the resource content itself.

Image element:

You store an image in an image element.

When you store an image in an image element:

- You can reference the image anywhere in your site regardless of the original file location or URL.
- The image is automatically transferred to other servers during syndication.
- You can reference the image element within other items as required.

Creating an image element

To create an image element, you can either add an image element to an authoring template, site area or content item, or create an image component.

Note: Images with names that include non-ascii characters cannot be pre-rendered. If you are pre-rendering a site, you must rename the image before you add it to the image element.

Valid mime types

You can configure which mime types are valid for an image element by editing the `imageresourcecmpnt.allowedmimetypes` setting. See "Web content authoring

options" on page 396 for further information.

JSP elements:

You use a JSP element to store a path to a JSP. When rendered within a presentation template or element design, a request to a JSP is generated and processed.

Creating a JSP element

To create a JSP element, you can either add a JSP element to an authoring template, site area, or content item, or create a JSP component.

Syndication:

Syndication does not move the JSP page referred to in a JSP element. Only the item that contains the JSP element is moved. You need to store the JSP page in the same folders of both the subscribing and syndicating servers.

Note:

The JSP referenced within a JSP component must not include a reference, directly or indirectly, to the same JSP component. This restriction includes references within web content tags or the API. If it does, a loop is created and errors occur.

How to use stylesheets in items:

Stylesheets can be used to format the design of IBM Web Content Manager pages in the same way as normal web pages.

Creating and referencing stylesheet elements

Stylesheet elements can be stored only in stylesheet components.

To link a stylesheet component to a specific authoring template, you must select a stylesheet component as the default stylesheet in an authoring template.

To link a stylesheet component to a specific site area or content item, you must add a component reference element to a site area or content item and select a stylesheet component.

Referencing a stylesheet element in a presentation template

Stylesheet elements are referenced in the "header" section of a presentation template by using either a style element tag or component tag.

Table 266. Referencing a stylesheet element in a presentation template

Details	Code example
To use the stylesheet that is specified in the authoring template of the current content item, you must use a `<styleElement>` tag.	`<HTML>` `<HEAD>` `<styleElement source="template"/>` `</HEAD>` `<BODY></BODY>` `</HTML>`

Table 266. Referencing a stylesheet element in a presentation template (continued)

Details	Code example
To use the stylesheet that is selected in a component reference element that is stored in either the current site area or content item, you must use a <styleElement> tag.	<HTML> <HEAD> <styleElement source="path" name="component reference name"/> </HEAD> <BODY></BODY> </HTML>
To use a specific stylesheet, you must use a <component> tag.	<HTML> <HEAD> <component name="stylesheet component name"/> </HEAD> <BODY></BODY> </HTML>

When rendered in web content, references to stylesheet components are rendered as external stylesheet links:

```
<HTML>
<HEAD>
<link href="stylesheet" media="media-type" rel="styleheet-type" type="text">
</HEAD>
<BODY></BODY>
</HTML>
```

How to use styles in HTML tags

Styles that are used in HTML are stored in presentation templates and element designs in the same way as normal HTML. The style must exist in the stylesheet that is referenced in the presentation template that is used to render the HTML.

For example, to add a class that is called "wcm" to a heading tag:

```
<H1 class="wcm">Heading</H1>
```

How to use styles in web content tags

Stylesheets can be used to format the style of content that is retrieved by using Web Content Manager tags. The style must exist in the stylesheet that is referenced in the presentation template that is used to render the Web Content Manager tag.

For example, to format the links in a menu by using a stylesheet class called "wcm", the following placeholder tag would be used:

```
<a href="<placeholder tag="href" />" class="wcm"><placeholder tag="name" /></a>
```

Selection elements

You use these elements to make selections from existing elements and data.

"Component reference element" on page 1818
You use a component reference element to store a reference to a component.

"Date and time element" on page 1818
You use a date and time element to store a date or time to be displayed on a web page.

"Option selection element"
An option selection element is used to present a list of predefined options that your content creators can select from when you create a content item or site area. An option selection element can be added to an authoring template only.

"User selection element" on page 1819
A user selection element stores a selection of users or groups.

Component reference element:

You use a component reference element to store a reference to a component.

You can use component reference elements in two ways:

- A component-reference element can be added to an authoring template so that content creators can select an existing component from the component library rather than build the component themselves.
- By referencing a component-reference element in a presentation template or another element, the component that is being displayed can be changed without having to change the presentation template or element design.

  For example, if you referenced a component-reference element called "Featured Product" in more than one presentation template the element that is referenced within the component-reference element can be changed without having to edit the presentation templates.

Creating a component reference element

To create a component reference element, you can either add a component reference element to an authoring template, site area or content item, or create a component reference component.

Date and time element:

You use a date and time element to store a date or time to be displayed on a web page.

Creating a date and time element

To create a date and time element, you can either add a date and time element to an authoring template, site area or content item, or create a date and time component.

Option selection element:

An option selection element is used to present a list of predefined options that your content creators can select from when you create a content item or site area. An option selection element can be added to an authoring template only.

When you use an option selection element, you can either enter a custom set of options, or select a set of categories. Which option you use depends on how you want to use the option selection element.

Simplifying item profiling

If you select a set of categories in an option selection element, you can choose to have the categories that a user selects when you create an item added to the item's profile. You can use multiple option selection elements on an item form to simplify

the process of profiling an item. For example, if you used separate taxonomies for "product type", "team" and "campaign", you might use three separate option selection elements in an authoring template. This strategy makes it easier for your content creators to select categories from each taxonomy on the content form.

Displaying different text on a rendered page

A simple use-case for using option selection elements is when you want to enable content creators to select from a short list of text options that are rendered on a page. For example, if you wanted to indicate different document types such as a "procedure", "policy" or "news" on the rendered page.

How to add custom code

Option selection elements can also be used to allow content creators to select different options that can then be used as parameters in some custom code such as a custom workflow or a JSP component.

User selection element:

A user selection element stores a selection of users or groups.

Creating a user selection element

To create a user selection element, you can either add a user selection element to an authoring template, site area, or content item, or create a user selection component.

Using a user selection element

Click **Select Users** to select users and groups.

- Select either **Users** or **Groups**.
- Type text to search in the **Search** field and then click **Search**. Leave the **Search** field blank to display all users or groups.) Select the required users or groups and then click **OK**.
- To remove a user or group, select the users or groups you would like to remove and then click **Remove**.

Note: Virtual users cannot be selected in a user selection element.

Personalized content

You use these elements to create or reference personalized content.

"Personalization element" on page 1820
A personalization element stores a reference to a personalization rule or content spot that is generated by Portal Personalization. To use a personalization element, you must create a personalization component.

"Personalization element examples" on page 1821
The layout and design of a personalization element is created in a similar way to a menu element, with a header design, footer design, and a design to be repeated for each result.

"Taxonomy element" on page 1823
A taxonomy element defines the layout of a category selection form that enables users to select categories to display in a personalized menu.

"Creating category selection trees" on page 1823
You use category selection trees to allow users to personalize menus.
"Using a user name element" on page 1826
A user name element displays the name of the current user in a presentation template, component design, or element design. You can use a user name element only by creating a user name component. You cannot add a user name element to authoring templates, site areas, or content items.

Personalization element:

A personalization element stores a reference to a personalization rule or content spot that is generated by Portal Personalization. To use a personalization element, you must create a personalization component.

A personalization element can:

- Display personalization content within a presentation template or element design.
- Display a link to personalization content within a presentation template or element design.
- Display attributes of personalization content within a presentation template or element design.

Creating a personalization element

You can use a personalization element only by creating a personalization component. You cannot add a personalization element to authoring templates, site areas, or content items.

Note: A maximum of 100 items can be displayed in a single Personalization element.

Access controls

When you create a personalization element, a user can select only those personalization rules and content spots that they have access to in Portal Personalization.

The personalization rule or content spot that is selected in the personalization element is rendered only if the user that views the web content has access to the personalization rule or content spot in Portal Personalization.

Re-creating Personalization Rules and content spots

If you delete a Personalization Rule or content Spot that is referenced in a Personalization element and then create a new Personalization Rule or content spot with the same name, it is no longer displayed in the Personalization element. You need to edit the Personalization element and reselect the Personalization Rule or content spot.

Caching Personalization components

Web content caching can sometimes be used with Personalization components but depends on the conditions set in the personalization rule, or the resources that are used to determine the rule results. Cache testing is required to determine whether

the content returned by your personalization component can be cached by using web content caching.

Personalization element examples:

The layout and design of a personalization element is created in a similar way to a menu element, with a header design, footer design, and a design to be repeated for each result.

Creating a Personalized menu

1. Create a content spot or personalization rule in Portal Personalization based on some Web Content Manager content.
2. Create a personalization element in Web Content Manager.
   - Click **Search** and select the content spot or personalization rule you created before. Click **OK**.
   - Create an element design to display the results of the content spot or personalization rule. This method is similar to designing a menu element or navigator. For example, enter this text in the "**Design for each menu search result**" section:

     ```
     [placeholder tag="namelink"]
     <br>
     ```
3. Save the personalization element.
4. Reference the personalization element in a presentation template.

Displaying personalized content

To display a single piece of personalized Web Content Manager content for different users:

1. Create an authoring template that includes an element. For example, a text element called "body".
2. Create a set of content items that are based on this authoring template.
3. Create a content spot or personalization rule in Portal Personalization based on the authoring template and content. The content spot or personalization rule must return only a single piece of Web Content Manager Content for each user.
4. Create a personalization element in Web Content Manager.
   - Click **Search** and select the content spot or personalization rule you created before. Click **OK**.
   - Create an element design to display the results of the content spot or personalization rule. For example, enter this text in the **Design for each menu search result** section:

     ```
     [element type="Content" context="autofill" key="Body"]
     ```

     This displays the content of the text element called "Body" from the content item that is returned by the content spot or personalization rule.
5. Save the personalization element.
6. Reference the personalization element in a presentation template.

Displaying personalized web content components

A set of personalized web content components can be displayed by using a personalization element:

1. Create a content spot or personalization rule in Portal Personalization that searches for web content components.
2. Create a personalization element in Web Content Manager.
   - Click **Search** and select the content spot or personalization rule you created before. Click **OK**.
   - Create an element design to display the results of the content spot or personalization rule. For example, Enter this text in the **element design** section:

     Header:

     ```
     <div>
     ```

     Design for each menu search result:

     You must use a Component tag with a context of "autofill".

     ```
     <span>
     [Component context="autofill"]
     </span><br>
     ```

     Footer:

     ```
     </div>
     ```
3. Save the personalization element.
4. Reference the personalization element in a presentation template.

Displaying attributes of Personalized content

The attributes of personalized content can also be displayed by using a personalization element:

1. Create a content spot or personalization rule in Portal Personalization .
2. Create a personalization element in Web Content Manager.
   - Click **Search** and select the content spot or personalization rule you created before. Click **OK**.
   - Create an element design to display the results of the content spot or personalization rule. For example, Enter this text in the **element design** section:

     Header:

     ```
     <div>
     ```

     Design for each menu search result:

     You must use a "AttributeResource" tag for each attribute you want to display. For example:

     ```
     <span>
     [AttributeResource attributeName="ibmcm:title"]
     [AttributeResource attributeName="ibmcm:effectiveDate"]
     </span><br>
     ```

     Footer:

     ```
     </div>
     ```
3. Save the personalization element.
4. Reference the personalization element in a presentation template.

Notes:

Displaying keywords and categories:

To retrieve a list of categories or keywords, Use the Property tag.

Displaying authors and owners:

To retrieve a list of authors or owners, Use the Property tag.

Displaying the Site Path:

To display the site path to a personalized Web Content Manager element, use a placeholder tag.

Taxonomy element:

A taxonomy element defines the layout of a category selection form that enables users to select categories to display in a personalized menu.

You configure the element by selecting either a taxonomy or a category to be the start area of the category selection tree. You then select a child depth relative to the start area. Select "Include Start" if you would like the start area to display in the category selection tree. This option has no effect if the start area is a taxonomy.

There are two element design options available: one is rendered when the logged in user selects the category that is to be displayed, and the other is rendered if the user does not select the category. These element designs are rich text elements, and are used in a similar fashion to the navigator and menu elements.

Note: To use this feature, you must configure a property extension database to store user-specific data. See “Configuring a property extension database” on page 596 for further information.

Creating a taxonomy element

You can use a taxonomy element only by creating a taxonomy component. You cannot add a taxonomy element to authoring templates, site areas, or content items.

Creating category selection trees:

You use category selection trees to allow users to personalize menus.

Note:

- Ensure the `connect.businesslogic.module.ajpecatselect.class` property is defined in the `WCM WCMConfigService` service, using the IBM WebSphere Application Server administration console, with a value of `com.aptrix.pluto.CategoryProfileUpdaterModule`.
- You cannot use category selection trees in a local rendering portlet. Instead, you must render the tree directly. For example:

  `http://host:port/wps/wcm/myconnect/library/`*`sitearea`*`/`*`content`*

Taxonomy element form

The main function of the taxonomy element is to display a category selection tree that is used to allow a user to select categories for menu personalization.

- You configure the element by selecting either a taxonomy or a category as a start area.
- Select a child depth from the start area and a parent level relative to the start area.

- Select "Include Start" to display the start area. This option has no effect if the start area is a taxonomy.
- There are two element design options available
  - One is rendered when the logged in user has selected the category that is to be displayed.
  - The other is rendered if the user has not selected the category.

  These element designs are rich text elements, and are used in a similar fashion to the navigator and menu elements.

The taxonomy element form example creates a check box input form:

- The category identity number is assigned to the "value" attribute in the input fields.
- Check box input fields are created, assigning the "selectedCategories" value to the "name" attribute.
- Hidden input fields are created, assigning the "visibleCategories" value to the "name" attribute.

Element designs

The following code examples are used to develop a basic category selection tree:

Table 267. Header

Code	Details
`[PathCmpnt end="/[Library]/[SiteArea]/[Content]" type="servlet"]` `method=post>` `"start="` `<FORM action='" type="servlet"]`	This code calls the Category Profile Updater Module.
`<input type="hidden"` `value='[PathCmpnt type="servlet"]/[Library]/[SiteArea]/[Content]'` `name="redirectURL">`	This code points to the page to go to after the user request has been processed. The element does not render correctly if the path, "[Site area]/[Content]" is not valid.
`<input type="hidden"` `name="updateSourceProfile" value="true">`	This line determines how a user's category profile is updated. The "value" parameter is optional. **True** Permanently updates the user category profile. **False** Updates only the user's session profile. **Updating the user's session profile:** The user's selected categories are calculated by combining the categories that are in the user's session profile and the categories that are in the permanent user category profile. Therefore, if a category is in the user category profile and is removed from the user's session profile only, it will still be shown as selected.

Table 268. Unselected element design

Code	Details
<input type="checkbox" name="selectedCategories" value="[Placeholder tag="idnum"]"/> <IndentCmpnt offset="0" repeat=".."> [Placeholder tag="name"] <input type="hidden" name="visibleCategories" value="[Placeholder tag="idnum"]"/> 	This is used to display unselected items in the rendered category selection tree.

Table 269. Selected element design

Code	Details
<input type="checkbox" checked name="selectedCategories" value="[Placeholder tag="idnum"]"/> [IndentCmpnt offset="0" repeat=".."] [Placeholder tag="name"] <input type="hidden" name="visibleCategories" value="[Placeholder tag="idnum"]"/> 	This is used to display selected items in the rendered category selection tree.

Table 270. Footer

Code	Details
<input type="submit" value="Set User Categories"> </form>	This footer contains the submit button.

Indent element:

This example uses the indent element tag. This can be used in the navigator and taxonomy elements. This tag represents an HTML/text string that should be repeated depending on the depth of a tree node being rendered in these elements.

In the taxonomy element example, the indent element is used to render and repeat the "." string dependent on the depth of the node the element design is being applied to. It is possible to offset the repeat value by assigning an integer value to the "offset" attribute of the tag. For example, A current node depth of 5 and an offset value of -2 would render the repeat string three times. If the sum of the offset and the node depth is negative or 0, the repeat string is not rendered.

element designs:

The only difference between the unselected element design and the selected element design is that the check box input field in the selected element design has the "checked" attribute set.

User access:

If using a taxonomy element, users must be given "Edit" access to their own user item to enable them to update their selected categories.

Using a URL to update user categories:

You can use a URL as an alternative to using a category selection tree to update a user's selected categories:

```
http://host:port/wcm/connect/SiteArea/SelectPage?MOD=AJPECatSelect
&redirectURL=/wcm/connect/SiteArea/Content&updateSourceProfile=false
&selectedCategories=categoryID1,categoryID3
&visibleCategories=categoryID1,categoryID2,categoryID3,categoryID4
```

The "selectedCategories" and "visibleCategories" parameters have multiple values which are comma delimited. The categories specified in "selectCategories" should be a subset of "visibleCategories".

This URL could be used on a page in the form of a button to allow users to update their user categories. For example, You could create a button that would add the category "News" to a user's selected categories list.

Related tasks:

“Setting service configuration properties” on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Using a user name element:

A user name element displays the name of the current user in a presentation template, component design, or element design. You can use a user name element only by creating a user name component. You cannot add a user name element to authoring templates, site areas, or content items.

About this task

To create a user name component, open the applications menu and go to **Content** > **Web Content Management** and then click **New** > **Component** > **User Name**.

Page navigation element

A page navigation element provides navigation controls that are used to browse through a set of results that are generated by menus, navigators, personalization, and search elements.

A page navigation element can generate two kinds of page navigation controls:

- Shuttle controls provide navigation relative to the current page. This includes sequential linking to the previous or next page of results and quick linking to the first and last pages in the set.
- Paging controls provide navigation according to the page number of the result set. A list of page number links is displayed, along with a continuation link for access to the previous or next set of page numbers, if all page numbers are not displayed.

A page navigation element can combine both shuttle and paging controls, as in the following Navigation example.

Table 271. Page navigation layout example

Attribute	Example
First	<<
Previous	<
Continuation	...
Page Numbers	4,5,6

Table 271. Page navigation layout example (continued)

Attribute	Example
Continuation	...
Jump to page	Go to page:
Next	>
Last	>>
Page size	Items to display: 10 | 50 | All

Creating a page navigation element

You can only use a page navigation element by creating a page navigation component. You cannot add a page navigation element to authoring templates, site areas, or content items.

URL generation by using `PathCmpnt` and `URLCmpnt` tags

There are some special considerations to keep in mind when you use `URLCmpnt` and `PathCmpnt` tags to create URLs to other web content items from within your content.

The `URLCmpnt` tag

The URL component tag `URLCmpnt` can be used to create URLs to content items in a presentation template. If the attribute `mode="portal"` is set, the content item is displayed through a content viewer portlet.

The `PathCmpnt` tag

The path component is used to create the base part of a URL in web content. Typically, the base part is extended by some string that identifies the content to be displayed. If the path component is used inside the context of a web content viewer, the generated URL has the updated URL format and thus cannot be displayed with the traditional web content viewer.

Adding URL parameters

Adding URL parameters to a regular portal URL do not affect the web content viewer. For this purpose, you must use the `URLCmpnt` or `PathCmpnt` tag. Extra URL query parameters, like those parameters that are used with the traditional web content viewer, can be appended to the URLs generated by these tags.

Personalizing federated documents

Portal Personalization provides the federated documents feature to retrieve metadata about documents that are stored in external content management systems or document repositories. Examples of these systems include IBM Content Manager, IBM FileNet Content Manager, and Microsoft Sharepoint. You can use a personalization component in IBM Web Content Manager to display metadata from federated documents and to create links to download or open the documents.

About this task

The federated documents feature can acquire metadata from remote systems that support the following methods:

- Document Services remote interfaces, as supported by IBM Content Manager and FileNet Content Manager
- Content Management Interoperability Services 1.0 (CMIS 1.0), as supported by IBM Connections, IBM Content Manager, IBM FileNet Content Manager, Microsoft Sharepoint, and others.
- Atom feeds

"Creating a federated documents selection rule"
Create a selection rule that selects metadata of documents that are contained in a specific folder of a remote content management system or document repository.

"How to use a federated documents rule in a personalization component" on page 1829
To access the information from a federated documents rule in your web content system, create a personalization component, associate it with the rule, and specify the design for displaying document information that is retrieved by the rule.

Creating a federated documents selection rule:

Create a selection rule that selects metadata of documents that are contained in a specific folder of a remote content management system or document repository.

Before you begin

Before you create a rule to access federated documents information, ensure that you configure the feature as described in "Setting up support for federated documents" on page 405.

About this task

Procedure

1. Open the Personalization Navigator.
2. Browse to the folder where you want to create the rule, or create a folder for the new rule.
3. Click **New** > **Rule**.
4. Enter a name for the rule.
5. Optional: Enter a description for the rule to identify the data that the rule selects.
6. Click the link for the **Select** action in the rule editor (for example, **Web Content**), and select **Federated Documents**.
7. Specify the folder on the remote system either by entering the URL directly or by browsing the remote system. In addition to a folder on a remote system, you can enter the URL of any Atom feed available on the network. The Atom data that is created by those feeds is mapped to corresponding AttributeResource tags in IBM Web Content Manager.
   - To enter the URL for a folder directly, complete these steps:
     a. Click **value** *.
     b. Enter the URL of the folder.
     c. Click **Submit**.
   - To browse a Document Services server or CMIS server, click the > symbol on the **Feed URL** condition, and click **Select Document Folder** to start the wizard to select a folder.

To connect to a remote server, you identify the server and the authentication method that is used to access the server. You can either select a predefined server or enter the server URL directly. By default, no predefined servers are configured, but an administrator can add servers to the list. To authenticate with a server, there are several available methods:

- If single sign-on is configured between the remote server and the portal, you can connect with the current user.
- You can enter a user ID and password for the remote server.
- You can select a credential vault slot that is associated with the server. Credential vault slots are set up by an administrator and enable users to log in without credentials.

If you use a user ID and password or credential vault slot, the remote server must accept authentication requests that use HTTP basic authentication.

For details on how administrators can add servers and configure authentication, see *Configuring the federated documents feature.*

Note: To browse remote servers and select a folder, the page that contains the Personalization Editor requires a theme and module profile that support the wp_federated_documents_picker theme module. For example, you can use the Portal 8.0 theme and the full or deferred profile. If the page does not use this theme module, the wizard to select a folder is not available when you click **Select Document Folder**. You can still manually enter a feed URL to a remote folder by clicking **value** *. To enable the folder selection wizard, ensure that the wp_federated_documents_picker theme module is available to the page that contains the Personalization Editor.

8. Optional: Click **show all items**, and specify the maximum number of entries to be retrieved.
9. Click **Save**.

Results

You can now use this rule in a personalization component to render the selection result of this rule in web content.

Related concepts:

"Reserved authoring portlet" on page 430
When you use the web content viewer or web content pages, some scenarios involve web content authoring tasks that are accomplished with authoring tools components. Authoring tasks are run through a special instance of the authoring portlet that is reserved specifically for these tasks and is installed on page that is hidden from the page navigation available to typical users.

"The module framework" on page 2526
The module framework allows extensions to contribute to different areas of a page to provide flexibility, enhance the user experience, and maximize performance.

Related tasks:

"Configuring the federated documents feature" on page 407
Configure the federated documents feature to specify information about the source servers for the documents that are available to users.

How to use a federated documents rule in a personalization component:

To access the information from a federated documents rule in your web content system, create a personalization component, associate it with the rule, and specify the design for displaying document information that is retrieved by the rule.

About this task

Procedure

1. Open the authoring portlet, and click **New** > **Component** > **Personalization**.
2. Enter a name and description for the new component.
3. Click **Search** in the **Personalization Element** section, and select the federated documents selection rule.
4. Define the design for the headers, footers, and individual documents that are selected by the configured selection rule. Use the **AttributeResource** element to reference individual metadata fields from the selected documents.
5. Save the component.

Results

You can now preview your component and reference it from within presentation templates, other element designs, or a web content viewer.

"Sample designs for a federated documents selection rule"
When you render document metadata information retrieved with a federated documents selection rule, you can tailor the header, footer, and menu search result designs for simple or more elaborate presentations.

"`AttributeResource` values for federated documents" on page 1831
The `AttributeResource` tag is used as a placeholder to display attributes from a federated documents selection rule within a personalization element design. It cannot be used in a presentation template or other element types.

Sample designs for a federated documents selection rule:

When you render document metadata information retrieved with a federated documents selection rule, you can tailor the header, footer, and menu search result designs for simple or more elaborate presentations.

Bulleted list design

To render a simple bulleted list of links to the documents, you can use a design like that described here.

Header

```
<ul>
```

Design for each menu search result

```
<li>
  <a target="_blank" href="[AttributeResource attributeName="contentLink"]">
    [AttributeResource attributeName="title"]
  </a>
</li>
```

Footer

```
</ul>
```

Table list design

To render a complete list of information with a table, you can use a design like that described here.

Header

```
<table>
<tr>
<th>Title</th>
<th>Authors</th>
<th>Load</th>
<th>Load via Portal</th>
<th>Load via Portal with Authentication</th><th>Load</th>
<th>Edit</th>
<th>Content Type</th>
<th>Size</th>
<th>Updated</th>
<th>Published</th>
</tr>
```

Design for each menu search result

```
<tr>
<td><b>[AttributeResource attributeName="title"]</b></td>
<td>[AttributeResource attributeName="authors" separator=","]</td>
<td><a target="_blank" href=" [AttributeResource attributeName="rawContentLink"]">download</a></td>
<td><a target="_blank" href=" [AttributeResource attributeName="contentLink"]">download</a></td>
<td><a target="_blank" href=" [AttributeResource attributeName="contentLinkAuthenticated"]">download</a></td>
<td><a target="_blank" href="[AttributeResource attributeName="viewLink"]">open</a></td>
<td>[AttributeResource attributeName="contentType"]</td>
<td>[AttributeResource attributeName="size"]</td>
<td>[AttributeResource attributeName="updated"]</td>
<td>[AttributeResource attributeName="published"]</td>
</tr>
```

Footer

```
</table>
```

`AttributeResource` values for federated documents:

The `AttributeResource` tag is used as a placeholder to display attributes from a federated documents selection rule within a personalization element design. It cannot be used in a presentation template or other element types.

When used with a federated documents selection rule, the following values for the `attributeName` attribute of the `AttributeResource` tag are supported for each document in the result set:

`authors`
: This attribute displays the names of the authors of the document. The actual result depends on the corresponding attribute mapping that needs to exist at the remote content management system. If no such mapping exists, an empty value is displayed.

`contentLink`
: This attribute displays the absolute URL that can be used to download the document. Unlike the `rawContentLink` attribute, the `contentLink` attribute contains the URL that addresses the Ajax proxy with parameters used to download the document through the proxy. You can disable proxied URLs by editing the `wcm.pzn.ecm.enable.proxy.content.links` property in the `WCM` `WCMConfigService` configuration service and setting the value to `false`.

`contentLinkAuthenticated`
: This attribute displays the absolute URL that can be used to download the document, including authentication information when needed. Similar to the `contentLink` attribute, the `contentLinkAuthenticated` attribute contains the URL that addresses the Ajax proxy with parameters used to download the document through the proxy. However, depending on the personalization selection rule, the `contentLinkAuthenticated` attribute

might also include information about a shared credential vault that is used to authenticate the user. The credential vault authentication information is available only if the selection rule was created by using a credential vault.

contentType
This attribute displays the MIME type of the document. If this information is not served by the remote content management system, an empty value is displayed.

contributors
This attribute displays the names of the contributors of the document. The actual result depends on the corresponding attribute mapping that needs to exist at the remote content management system. If no such mapping exists, an empty value is displayed.

id This attribute displays the unique ID of the document.

published
This attribute indicates the point in time of the first availability of the document. The actual result depends on the corresponding attribute mapping that exists at the remote content management system. If no such mapping exists, an empty value is displayed.

rawContentLink
This attribute displays the raw absolute URL that can be used to download the document. Although similar to the `contentLink` attribute, the `rawContentLink` attribute contains the URL as it appears in the federated documents feed used by the federated document selection rule. This value does not include any additional proxy that is addressed in the URL.

size This attribute displays the size of the document in bytes. If this information is not served by the remote content management system, an empty value is displayed.

summary
This attribute displays the summary of the document. If this information is not returned by the remote content management system, an empty value is displayed.

Note: Because the value of the summary attribute can contain large character data, the amount of data that is returned by this attribute is limited. You can increase or decrease the amount of data that is returned by setting the `wcm.pzn.ecm.max.field.length` property in the **WCM WCMConfigService** configuration service.

title This attribute displays the title of the document. The actual result depends on the corresponding attribute mapping that needs to exist at the remote content management system. If no such mapping exists, the file name is displayed.

updated
This attribute indicates the point in time of the last update to the document. The actual result depends on the corresponding attribute mapping that exists at the remote content management system. If no such mapping exists, an empty value is displayed.

viewLink
This attribute displays the absolute URL that can be used to open the document in context of the remote content management user interface. If no such URL is returned by the remote content management system, an empty value is displayed.

Note: The `viewLink` attribute is not supported if you are connected to a IBM Content Manager repository or are using a CMIS server.

Here is an example of how to use these attribute values in a design:

```
<li>
  <a target="_blank" href="[AttributeResource attributeName="contentLink"]">
    [AttributeResource attributeName="title"]
  </a>
</li>
```

Tags

Tags are used in your markup to reference content that is stored or generated by elements, or to display metadata from different items.

"Web content tags"
You use tags to reference elements within presentation templates and element designs.

CF07 "Web content tag behavior and enhancements"
These features and enhancements can be used to simplify the process of creating and maintaining web content tags.

"Indenting element designs" on page 1835
You use an indent tag to format element designs that require results to be indented.

"Writing links to web content" on page 1836
Links to content items can be written as URLs.

"Contextual linking" on page 1840
Contextual linking is used in systems where content from one site is shared across multiple sites. When content is linked into a site, embedded links (link elements and links in HTML) reference the site the original content item is in. Contextual linking is used so that when content is linked from another site, the link is rendered relative to the current site if possible.

"Custom caching" on page 1840
You can overrule the default caching parameters of a site by using "cache" and "expire" parameters in URLs and IBM Web Content Manager tags.

Web content tags

You use tags to reference elements within presentation templates and element designs.

To create a web content tag, click **Insert a Tag** from a presentation template or element design field. The Tag Helper dialog opens.

Web content tag behavior and enhancements

These features and enhancements can be used to simplify the process of creating and maintaining web content tags.

CF07

Tag delimiter

Web content tags can be written by using square or angle brackets. For example:

- `[Property field="title"]`
- `<Property field="title">`

CF07

Parameter delimiter

A choice of attribute delimiters can be used when you write web content tags. For example, the field parameter in a property tag can be written with double quotation marks, single quotation marks, or no quotation marks:

- `[Property field="title"]`
- `[Property field='title']`
- `[Property field=title]`

Note: The anchor and image HTML tags are the only HTML tags that are processed within web content tags. An attribute delimiter tag must always be used with these HTML tags, and are added automatically if omitted.

CF07

Primary parameters

Tags can be shortened by using their primary parameter.

For example, the tag `[Property field="title"]` can be shortened to `[Property:title]`.

These primary parameters can be used to shorten a tag:

Field This parameter is the primary parameter of the Property and EditableProperty tags.

Key This parameter is the primary parameter of the Element, EditableElement, IfDefined, and IfNotDefined tags.

Restriction: A shortened tag cannot be used with elements that have a space in their name. For example, if you add a text element to an item named "First Text Element" you cannot shorten its tag to `[Element:First Text Element]`. You must instead use the full tag syntax: `[Element key="First Text Element"]`

Tag This parameter is the primary parameter of the Placeholder tag.

Type This parameter is the primary parameter of the Path Component tag.

CF07

Dynamic Parameter Tag

The dynamic parameter tag is used as a simple representation of a more complex web content tag. It is represented by using a $ symbol.

For example, the property tag `[Property field="title"]` can be written as `[$title]`.

When a dynamic parameter tag is used, the context of the tag is determined in this order: Property, then element, then parameter resource. So, if you used this tag, `[$yellow]`, a property named "yellow" would be used first. If this property does not exist, an element named "yellow" would be used. If neither of these tags exist, a tag that contains a resource parameter named "yellow" would set the context.

The parameters of a dynamic parameter tag always default to `context="autofill"`, and `type="auto"`.

CAUTION: If you use the dynamic parameter tag, errors will occur when using cross-version syndication to servers that are not upgraded to CF07 or higher. All data will be lost from any element on the older server that uses a dynamic parameter tag.

CAUTION: If you roll back to CF06 or earlier after using the dynamic parameter tag, you will need to change any tags using the dynamic parameter tag back to the pre-CF07 standards.

CF07

Simplified plug-in tags

This tag is an example of a plug-in tag: `[Plugin:Page paramKey1='id']`

You can simplify the plug-in tag to this: `[Page paramKey1='id']`

The simplified plug-in tag cannot replace an existing web content tag. For example, if you created a plug-in named "Property", you must use the full tag: `[Plugin:Property]`. The tag `[Property]` is treated as a property tag, not a plug-in tag.

CF07

Default undefined tag attributes

If not specified in a tag, the `context` and `type` tag parameters default to `context="autofill"` and `type="auto"`.

CF07

HTML image tag

The border tag that is used in HTML image tags that are generated by Web Content Manager are not rendered in the tag if the border specified in an image element is not specified, or is zero.

Indenting element designs

You use an indent tag to format element designs that require results to be indented.

About this task

This is an example of the format of an IndentCmpnt tag:

```
[indentcmpnt repeat=" " offset=" " start=" " end=" " ]
```

To create a IndentCmpnt tag:

Procedure

1. Click **Insert a Tag** from a presentation template, component, or element design field. The **Tag Helper** dialog opens.
2. Select **Indent** as the tag type.
3. Select an offset level. The offset is used to determine how many times the repeat string is used for each indent. The offsets that are used are based on the number of nodes of the hierarchical content that is displayed. For example, A current node depth of 5 and an offset value of -2 would render the repeat string three times. If the sum of the offset and the node depth is negative or 0, the repeat string is not rendered.

4. Click **OK** to add the tag to your navigator design.

What to do next

When you add the tag to your design, you can also add the following parameters to the tag:

Table 272. indent tag parameters

Tag parameter	Description
`repeat=" "`	Enter the text to repeat at the beginning of the indent.
`start=" "` `end=" "`	The start and end attributes are used to wrap the data that is returned by a tag within other tags, such as HTML. These attributes are not mandatory.

Double-byte character sets:

Not all double-byte character sets support extended ASCII. To use tags, such as "` `", you need to replace "`&`" with "`&`".

For example:

```
[indentcmpnt offset="0" repeat="&amp;nbsp;&amp;nbsp;"]
```

Writing links to web content

Links to content items can be written as URLs.

Linking to web content from other web content

The following examples show how to write links to web content that are to be used with the web content viewer or the Web Content Manager servlet.

To create a link from a piece of web content to another piece of web content, use the following URL format:

```
[URLCmpnt mode="current" context="Selected" type="Content"
name="library/site_area_path/content"]
```

library
The name of the web content library.

site_area_path
The path to the site area where the content is located.

content
The name of the content item.

Linking to web content from an external portlet or website

To create a link from an external portlet or website that displays web content, use the following URL format:

```
http://hostname/context_root/library/site_area_path/content
```

hostname
The name of the Web Content Manager host.

context_root
The Web Content Manager context root. For example: `wps/wcm/connect`

library
The name of the web content library.

site_area_path
The path to the site area where the content is located.

content
The name of the content item.

Linking to content displayed in a web content viewer from an external portlet or website

To create a link from an external portlet or website to content displayed in a web content viewer, use one of the following piece of content (POC) URIs:

wcm:path:*content_path*
Use this POC URI to link to the content with the specified path. For example: `/Web Content/Articles/Sample Article`.

wcm:oid:*content_id*
Use this POC URI to link to the content with the specified ID. For example: `fa2bfd32-7b2f-4394-a5ab-2e150c5ed8aa`.

Use one of the POC URIs to create a URL with the following format:

`http://`*`hostname`*`/`*`context_root`*`/mypoc/`*`virtual_portal_context?urile=poc_uri`*

Note: The POC URI needs to be URL encoded. For example:

```
http://myportal/wps/mypoc/myvp?urile=wcm%3Apath%3A/Web+Content/Articles/
Sample+Article
```

You can add query parameters to the URL to specify how the portal resolves the POC URI.

To address a specific portal page, use one of the following parameters. The parameters cannot be combined:

page To specify the unique name or the object ID of the target page, use this parameter. For example: `&page=my.content.page`

mapping
To specify the target page using a URL mapping, use this parameter. For example: `&mapping=myContentPage`

current
To specify that the current page is the target page, use this parameter. For example: `&current=true`

To use a specific presentation template to render the requested content, use the following parameter:

pagedesign
Specify the path to the presentation template in your web content library including the names of folders. For example: `&pagedesign=/Web+Content/My+Templates+Folder/My+Presentation+Template`

CF05 To specify a mime type for the Content as a Service feature described in topic *Content as a Service,* use the following parameter:

mime-type
Specify the MIME type that is used as response content type of a **Content as a Service** request. If you enable your content for different data formats based on the MIME type, this parameter specifies the content type produced by the design component that renders the requested content. You can either use an element from the content item or a presentation template to produce the output. For more information about enabling different data formats based on the mime-type parameter, go to section **Selecting data format based on MIME** type. For example: `&mime-type=application/json`

Dynamic page lookup: The `page` parameter is optional. You can use the link broadcasting feature of the web content viewer to dynamically look up pages by omitting the `page` parameter. For example, if you have a content item **News1**, stored in the library **Web Content** under the site area **News**, you can create a link to that content item with the following URL:

`http://`*`hostname`*`/`*`context_root_poc`*`?urile=wcm%3Apath%3A/Web+Content/News/News1`

hostname
The name of the Web Content Manager host.

context_root_poc
The portal context root. For example, **wps/poc** or **wps/mypoc**.

Alternatively you can also add a specific portal page by using a URL mapping by using the following format:

`http://`*`hostname`*`/`*`context_root`*`/portal_page_url_mapping/?current=true&urile=wcm%3Apath%3A`*`library/site_a`*

hostname
The name of the Web Content Manager host.

context_root
The portal context root. For example, anonymous sites can use **wps/portal**. Otherwise use **wps/myportal**.

portal_page_url_mapping
The compound name of the portal URL mapping to the portal page that contains the Web Content Manager portlet (URL mappings can be set up using the portal administration portlets).

library
The name of the web content library.

site_area_path
The path to the site area where the content is located.

content
The name of the content item.

Note: The web content viewer on the target page must be configured to receive links from **Other portlets and this portlet**.

Adding cache parameters to a URL

You can add web content "Cache parameters" on page 1841 and "Cache expire parameters" on page 1842 to a URL to custom caching strategies to individual items. For example:

`http://`*`hostname`*`/`*`context_root`*`/`*`library`*`/`*`site_area_path`*`/`*`content`*`?cache=site`
`&contentcache=session`

hostname
The name of the Web Content Manager host.

context_root
The Web Content Manager context root. For example: `wps/wcm/connect`

library
The name of the web content library.

site_area_path
The path to the site area where the content is located.

content
The name of the content item.

Adding a last modified parameter to a URL

You can add the last modified date of the current content item to the header of the rendered page. For example:

`http://`*`hostname`*`/`*`context_root`*`/`*`library`*`/`*`site_area_path`*`/`
`content``?returnLastModified=true`

hostname
The name of the Web Content Manager host.

context_root
The Web Content Manager context root. For example: `wps/wcm/connect`

library
The name of the web content library.

site_area_path
The path to the site area where the content is located.

content
The name of the content item.

Overriding the context mode

The **`urlModeOverride`** parameter is used to override the default URL context when linking to URL, Link, and Placeholder components. These are the accepted values:

`current`
Chooses the URL style based on the request.

`standalone`
Generates URLs that render outside a portal site, such as a servlet site.

`poc` Generates a stateful Portal URL that renders content against the mapped wcm rendering portlet, or a servlet URL if no mapping exists. Functions as 'static' when rendering outside a portal site.

`static` Generates a short stateless Portal URL, or a servlet URL if no mapping exists.

Other URL Parameters

`WCMRenderAbsoluteURLs`
Add this to a URL to generate an absolute URL of an item instead of the relative URL.

Related concepts:

"Content as a Service pages" on page 1881
Starting with version 8.5 CF05, IBM WebSphere Portal Express introduces the concept of Content as a Service pages. The Content as a Service pages can be used to render content that is managed by your IBM Web Content Manager in different data formats.

"Select data format based on MIME type" on page 1882
With Content as a Service pages, you can specify different representation of your web content for different MIME types. This way when you request Content as a Service pages, you can specify the preferred representation of your web content. There are different options to manage the presentation components that produce the output for the MIME types you like to support.

Contextual linking

Contextual linking is used in systems where content from one site is shared across multiple sites. When content is linked into a site, embedded links (link elements and links in HTML) reference the site the original content item is in. Contextual linking is used so that when content is linked from another site, the link is rendered relative to the current site if possible.

Contextual path links

Contextual path links attempt to resolve a link by using a relative path technique. Contextual path linking assumes that each site framework that the linked content is stored in has the same site structure.

Contextual path links can be applied to elements referenced by using the element tag. For example:

- `[Element type="content" context="current" key="body" link="path" ]`

It can be used only if *context=current* or *context=autofill*.

When contextual path linking is used, a compatible link is searched for using the same relative path. If no link is found, the original link is used.

Custom caching

You can overrule the default caching parameters of a site by using "cache" and "expire" parameters in URLs and IBM Web Content Manager tags.

Note: Custom caching can be used only when a server's default web content cache is set to none or advanced caching. If basic caching is used as your default web Content cache, Custom caching cannot be used.

There are two basic methods in which custom caching can be used with your default server caching settings:

Default Server Caching Enabled

In this scenario, some form of default server caching is enabled. Caching parameters within connect tags and URLs can be used to either:

- Disable caching for the data that is being requested.
- Apply different caching parameters to the data that is being requested.

This method is used with sites that are mostly static, but that contain a few dynamic elements that require a different caching strategy from the server's default caching strategy.

Default Server Caching Disabled

In this scenario, default server caching is disabled. Caching parameters within connect tags and URLs can be used to enable caching for the data that is being requested.

This scenario is used with sites that contain many elements that require different caching strategies.

"Cache parameters"
Use the **cache** parameters in IBM Web Content Manager tags and URLs to specify whether the retrieved data is cached or not. If it is cached, how it is cached. The **cache** parameter is not mandatory.

"Cache expire parameters" on page 1842
You use the "expires" parameter in IBM Web Content Manager tags and URLs to specify how long to maintain data in the cache before it is expired. When data expires from a cache, the next request for the data will be retrieved from the original server. The **expires** parameter is not mandatory.

"Caching IBM Web Content Manager elements" on page 1845
You can apply caching to elements by using "connect" tags to reference elements within presentation templates instead of the component or element tag.

Cache parameters:

Use the **cache** parameters in IBM Web Content Manager tags and URLs to specify whether the retrieved data is cached or not. If it is cached, how it is cached. The **cache** parameter is not mandatory.

Custom caching parameters can be used only when a server's default web content cache is set to none or advanced caching. If basic caching is used as your default web content cache, custom caching cannot be used. Custom caching can be used to set cache parameters for basic, advanced, and data caches. When custom caching is used in a connect tag, the caching applies to the data that is being retrieved with the connect tag. When custom caching is used in a URL request, the caching applies to the entire page that is being requested.

Table 273. Values for the CACHE parameter

Basic caching	Advanced Caching	Data caching
CACHE=SITE CACHE=SESSION CACHE=NONE	CONTENTCACHE=SITE CONTENTCACHE=SESSION CONTENTCACHE=USER CONTENTCACHE=SECURED CONTENTCACHE=PERSONALIZED CONTENTCACHE=NONE	CONNECTORCACHE=SITE CONNECTORCACHE=SESSION CONNECTORCACHE=NONE

Examples:

```
<CONNECT MOD=Web SRV=HTML ACTION=http://www.ibm.com CACHE=SITE >
```

```
http://host:port/wps/wcm/connect/library/sitearea/content?cache=site&contentcache=session
```

Custom caching strategies

- When you apply custom caching to static content, you would mostly use `CACHE=SITE`, `CACHE=SESSION` or `CONTENTCACHE=USER`.
- When User Groups are used in implementing site security, you can use the `SECURED` custom caching strategy: `CONTENTCACHE=SECURED`.

- When Categories and/or Keywords, along with User Groups, are used for customization of your site, you can use the `PERSONALIZED` custom caching strategy: `CONTENTCACHE=PERSONALIZED`.
- If your Server's default web Content Cache is set to Advanced, you must use `CONTENTCACHE=NONE` to disable caching.
- If you retrieve external data, you must use `CONNECTORCACHE=NONE` to disable caching.

CacheKey parameter

The *CacheKey* parameter is used when caching content with the basic cache. A CacheKey is used as a key instead of a URL. This strategy is useful if you have multiple URLs for the same page but want it cached only once. This reduces the amount of memory that is used by the cache.

Example:

The following URLs use the same web page called news.html.

```
<CONNECT MOD=Web SRV=HTML ACTION=http://www.ibm.com/news.html
CACHE=SITE CACHEKEY=news >

<CONNECT MOD=Web SRV=HTML ACTION=http://www.ibm.com.au/news.html
CACHE=SITE CACHEKEY=news >

<CONNECT MOD=Web SRV=HTML ACTION=http://www.lotus.com/news.html
CACHE=SITE CACHEKEY=news >
```

In this example, "news" is used as the CacheKey to store the value of the response from these connect tags. This means that news.html is cached only once instead of being cached three separate times.

Caching, content updates, and syndication

When an item is updated, either directly or as a result of syndication, no cache is updated. The rendered items are not updated until each configured cache is expired. It is important to choose cache timeout parameters that complement your syndication strategy.

Cache expire parameters:

You use the "expires" parameter in IBM Web Content Manager tags and URLs to specify how long to maintain data in the cache before it is expired. When data expires from a cache, the next request for the data will be retrieved from the original server. The **`expires`** parameter is not mandatory.

Custom expiring parameters can be used only when a server's default web content cache is set to none or advanced caching. If basic caching is used as your default web content cache, custom expiration cannot be used. Even though you cannot use custom expiration with basic caching enabled, you can still use custom expiration (when you use the advanced cache) to expire data in the basic cache.

Values for the **`expires`** parameter can represent either a relative period or an absolute date and time:

Basic cache

- `EXPIRES="ABS {date and time}"`
- `EXPIRES="REL {integer}{units}"`

Advanced Caching

- CONTENTCACHEEXPIRES="ABS {date and time}"
- CONTENTCACHEEXPIRES="REL {integer}{units}"

Data caching

- CONNECTORCACHEEXPIRES="ABS {date and time}"
- CONNECTORCACHEEXPIRES="REL {integer}{units}"

Examples:

```
<CONNECT MOD=Web SRV=HTML ACTION=http://www.ibm.com CACHE=SITE
EXPIRES="REL 9000s">
```

```
http://host:port/wps/wcm/connect/library/sitearea/content?cache=site&expires="REL 9000s"
```

Custom expiring strategies

- CONNECTORCACHEEXPIRY= must be used when you set custom expiry parameters to retrieving external data by using a connect tag or URL request.
- If your default cache is basic, or if you specify CACHE= in a connect tag or URL request, you must use EXPIRES=.
- If your default cache is advanced, or if you specify CONTENTCACHE= in a connect tag or URL request, you must use CONTENTCACHEEXPIRES=.
- If your default cache is none, and only CACHE=, or CONTENTCACHE= is specified in a connect tag or URL request, the connect.connector.httpconnector.defaultcacheexpires property in the WCM WCMConfigService service is used to expire the data.

Specifying an absolute time

An absolute date specifies the date and time the document expires.

To indicate a time, use the following format:

- ABS {date and time}

For example:

- ABS Mon, 29 May 2000 03:04:18 GMT

A request for this document after this exact time causes the document to be flushed from the cache and a new copy retrieved.

When you specify an absolute expiry date, the date must be prefixed with ABS, and the date must be in one of the following formats:

- Mon, 06 Nov 2000 09:00:00 GMT
- Monday, 06-Nov-00 09:00:00 GMT
- Mon Nov 6 09:00:00 2000.
- 6 Nov 2000 9:00 AM.

The first three date formats use the standard HTTP specification, while the last is a simple, short date format for convenience.

When you use absolute times and dates to expire data, cached items remain in the cache until they expire. When expired, the original item is retrieved on the next request and a copy is placed in the cache. Because the absolute time or date has already expired, the item will immediately be expired. Whn expired, an item is not

permanently cached again when you use absolute times and dates. All absolute time values are in GMT.

Specifying a relative period

Rather than specifying an absolute time, a relative time can be used to specify that the document will expire some time after the document is placed in the cache, for example a number of hours or days. The actual time the document expires is then calculated from the time the document is retrieved and added to the cache.

Rather than specifying a fixed time for the expiry of cached data, the expiry can be specified relative to the time that the data was added to the cache, for example, a number of hours or days.

To indicate a relative time use the following format:

- REL {integer}{units}

Note: The space after REL is required.

The integer specifies a whole number of time units. Decimal numbers are not supported. The units are specified by using a single case-insensitive character:

- S: Seconds
- H: Hours
- D: Days
- M: Months

Table 274. Formatting examples

In a connect tag	In a URL Request
• `EXPIRES="REL 2M"` • `EXPIRES="REL 9000s"`	• `EXPIRES=REL 2M` • `EXPIRES=REL 9000s`

The first example indicates an expiry of two months. The second indicates 9000 seconds (2.5 hours).

By design only seconds, hours, days, or months can be specified. Minutes are not supported to simplify the interface (M is used for months). Instead, a multiple of seconds can be used (for example, 300 seconds for 5 minutes).

Caching, content updates, and syndication

When an item is updated, either directly or as a result of syndication, no cache is updated. The rendered item is not updated until each configured cache is expired. It is important to choose cache timeout parameters that compliment your syndication strategy.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Caching IBM Web Content Manager elements:

You can apply caching to elements by using "connect" tags to reference elements within presentation templates instead of the component or element tag.

Example: Applying custom caching

An example of the type of tag that can be used to cache individual elements within a presentation template:

```
<connect
 SRV="cmpnt" PATH="/Library/SiteArea/Content"
 SOURCE="library" CMPNTNAME="TestNav" CONTENTCACHE="site" CONTENTCACHEEXPIRES="REL 9000s">
</connect>
```

Table 275. Connect tag details

Parameter	Details
`SRV="cmpnt"`	The Service for this Module is "cmpnt".
`PATH="/libraryname/SiteArea/Content"`	This parameter sets the context for the element. The "sitepath" and "name" placeholders can be used instead of "PATH=" when caching menus or navigators: `[placeholder tag="sitepath"]/[placeholder tag="name"]`
`SOURCE="library"`	Source is either "content", "sitearea" or "library". In this example it is "library" because the element being cached comes from a component.
`CMPNTNAME="TestNav"`	This parameter is the name of the element to be cached.
`CONTENTCACHE="site"`	This parameter is either "site or "session".
`CONTENTCACHEEXPIRES="REL 9000s"`	The time the component expires from the cache is set here.

The first time the presentation template is rendered, the element is added to the cache. The next time the presentation template is rendered, the element is displayed from the cache instead of being rendered afresh by the Web Content Manager application. The element is not rendered again by the Web Content Manager application until it is expired from the cache. For this reason, only elements that do not require to be freshly rendered every time that a page is accessed should be cached.

If you are caching a component that is used in more than one presentation template, save the connect tag as a HTML component and then reference that component in each presentation template. If you then need to change the cached component tags, you only need to change it in the HTML component rather than in multiple presentation templates.

If you have a set of cached components that use the same `ContentCacheExpires` parameter, save the `ContentCacheExpires` parameter as an HTML component and then reference that component in each connect tag that is used to cache components. If you then need to change the `ContentCacheExpires` parameter, you change it in the HTML component only, rather than in multiple connect tags. This also applies to any common cache tags.

Example: Disabling caching

You can also use this method to disable caching. In this example the property *CONTENTCACHE=NONE* is used to disable caching of this element.

```
<connect
 SRV="cmpnt" PATH="/SiteArea/Content"
 SOURCE="library" CMPNTNAME="TestNav" CONTENTCACHE="none" >
</connect>
```

Setting up search for site visitors

Learn about the initial considerations for setting up search for site visitors before you begin configuring and administering search on your website.

What is your portal environment?
If you are operating in a stand-alone environment, you can use the Default Search collection to start working immediately with Portal Search. If you are operating in a cluster environment, remote search service must be installed and configured before Portal Search can be used. Remote search service enables all nodes of the cluster to equally access the same search service.

What keywords, categories, and metadata do you want to specify for searching?
Predicting specific search terms and keywords to index is an important first step when you are setting up your site for search by site visitors. Oftentimes, the prediction of search terms, metadata, and keywords to be indexed depends upon current industry and marketing trends for your specific business. Content authors can set keywords and metadata terms for a specific page. You can also enable searching of the words that are used in the body of your content instead of keywords or metadata that is associated with content items.

How do you want to return search results when a site visitor runs a search?
The results of a search on WebSphere Portal collections are displayed in the Search Center by default. The Search Center is wired together with the search form in standard themes included with WebSphere Portal. However, you can use the Search Results page template to create a search landing page that displays a list of search results when a site visitor runs a search query. The search landing page that you create can be placed anywhere in your site. Search results are automatically displayed on the new search landing page instead of in the Search Center.

Do you need to index documents of various formats, such as PDFs?
If so, you must install and configure Document Conversion Services (DCS). This third-party software is included with WebSphere Portal. DCS converts various types of documents into a format that can be consumed and processed by the search engine. DCS is needed in order for attachments to be indexed and appear in search results.

Which search service suits your business and website needs?
Portal Search is available for immediate use, though you can also provide advanced search features on your website by using IBM Content Analytics with Enterprise Search. Additionally, you can integrate search on your website with third-party search engines if you have a license for that third-party search software. Your website's specific environment and search needs determine which search service your website requires.

Taxonomies, Categories, and keywords

The combination of taxonomy and categories enables control what displays in menus.

Taxonomies and categories

A Category refers to the subject matter of your content item. For example, your content item might be of the category *New Products* or *Latest News*. Sets of categories are grouped within different top-level taxonomies. List of links to content items that use the same category are displayed by using a menu component.

Keywords

Use Keywords to profile content. Unlike categories, which are chosen from a predefined list, you can enter any keywords that you like when you create content items. List of links to content items that use the same keyword are displayed by using a menu component.

Taxonomy example

In the example:

- "Financial" is the ancestor of "Interest Rates", "Personal", "Business", "Corporate" and "Banking Solutions".
- "Personal", "Business" and "Corporate" are the descendants of "Interest Rates" and "Financial".
- Financial
  - Banking Solutions
  - Interest Rates
    - Personal
    - Business
    - Corporate
- News

Category scenarios

When building a hierarchy of taxonomies and categories, it is important to consider how a menu uses your categories in a search, because menus search both upwards and downwards within groups of categories.

Table 276. Examples

Scenario	Example
If you base a menu on a top-level category, all content that is profiled with categories that belong to that top-level category and their descendants appear in the menu.	In the previous example, a menu based on Financial displays content that is profiled with any of the following: • Financial – Banking Solutions – Interest Rates - Personal - Business - Corporate

Table 276. Examples (continued)

Scenario	Example
If you base a menu on a mid-level category, all content that is profiled with the mid-level category or its descendants or its ancestors appear in the menu.	A menu based on Interest Rates displays content that is profiled with any of the following: • Financial – Interest Rates - Personal. - Business. - Corporate.
If you base a menu on a bottom-level category, all content that is profiled with the bottom-level category or its ancestors are returned by the menu.	A menu based on Business displays content that is profiled with any of the following: • Financial – Interest Rates - Business.

How to store translated text in a content item or site area

CF07

Translated text can be stored in a content item or site area. The translated text can then be referenced in web content tags, or as localized text in web content authoring portlet forms.

How to create the content item or site area to store the translated text

- Create a content item or site area.
- Add `ibm.wcm.TextProvider` as a keyword in the content item or site area's profile settings.
- Create a set of text elements for each language that is used by your site. Each text element must be named entering a language code for the language in the name field. See "Supported languages" on page 3622 for a list of codes. At render time, if the requested language does not exist in the list of elements, the first element is used as the default language.
- Type the translated text in each field by using this format: `keyname=translated-text`. You can add as many key names as you need into each text element. For example:

```
keyname1=translated text
keyname2=translated text
keyname3=translated text
```

- The key names must be consistent for each language.
- The list of key names and translated text uses the Java Properties File Format.

How to select the translated text in a web content authoring portlet form

Fields that support localized text have a **Localizations** link next to the field title. Click Localizations to select the content item or site area where your translated text is stored, and then select a key name.

How to select the translated text in a web content tag

You can render translated text by using a plug-in tag:

```
[Plugin:TextProvider key="" provider=""]
```

You specify the *name* path to the content item or site area in the `provider` parameter, and the name of the key to display in the `key` parameter. For example:

```
[Plugin:TextProvider key="keyname2" provider="library/sitearea/contentitem"]
```

Note: If you move or rename a content item or site area that is used to store text provider text, or its parent, you will break the text provider reference that is specified in a tag. To avoid this issue, you can instead specify the item ID instead of the name path.

For example:

```
[Plugin:TextProvider provider="[Element key='Text Provider Name' format='id']" key="keyname2"]
```

See Content plug-ins for further information about text provider plug-in tag.

Setting a default text provider in a web content tag

If you intend to use the TextProvider plug-in tag more than once in a component design or presentation template design using the same text provider, you can specify the default text provider item name path in one tag:

```
[Plugin:TextProvider defaultProvider="library/sitearea/contentitem"]
```

You can then leave out the provider parameter in the tags that follow the default provider tag:

```
[Plugin:TextProvider defaultProvider="library/sitearea/contentitem"]
[Plugin:TextProvider key="key1"]
[Plugin:TextProvider key="key2"]
[Plugin:TextProvider key="key3"]
```

The default provider is applied in all presentation template designs and component designs, including component designs that are included by using the tags, such as a component tag.

To remove the default provider, use:

```
[Plugin:TextProvider defaultProvider=""]
```

How to use replacement characters

You can use replacement characters with the text provider plug-in tag.

For example, to use `keyname_with_replacement=Welcome {0} to the website of {1}` in the content item that is located at `library/sitearea/contentitem`, use:

```
[Plugin:TextProvider key="keyname_with_replacement" provider="library/sitearea/contentitem" param.
```

This is rendered as:

```
Welcome Fred to the web site of Mary
```

Hiding a text provider item from the web content authoring interface

If you don't want a text provider item to appear in the list of text providers in the web content authoring interface, add a text element with the name of `isShownInAuthoringUI`. Then, type the word `false` in the text element field. You can still reference this text provider item in a plugin-in tag even though it is now hidden from the list of text providers in the web content authoring interface.

Hiding content

You can hide portlets and widgets on a page so they are present but not visible in the page layout.

About this task

Why Hide Content?

Sometimes you might want a portlet or widget on the page, but it does not have a user interface. In that case, it should be hidden. An example is if you have a data processing widget and a data viewer widget that are wired together. The data viewer widget is visible and the data processing widget would stay hidden.

In the portal theme, hide content by putting it into a special container in the layout that is hidden with CSS. All of the out of box layout templates have this hidden container.

Procedure

1. Viewing hidden content
   a. Click **Edit Mode**.
   b. Click **Menu** > **Show Hidden Content**.
2. How to hide content. There are two ways to hide a piece of content:
   Use the **Portlet Actions** menu:
   a. View the hidden content as described in step 1.
   b. Click the drop-down menu icon in the portlet or widget.
   c. Click **Hide**.
   d. Click **Save Draft** to save the page.

   Use drag and drop option:
   a. View the hidden content as described in step 1.
   b. Drag the portlet or widget into the hidden container that appeared in step a.
   c. Click **Save Draft** to save the page.
3. How to unhide content There are two ways to display a piece of content:
   Use the **Portlet Actions** menu:
   a. View the hidden content as described in step 1.
   b. Click the drop-down menu icon in the hidden portlet or widget.
   c. Click **Move Down**.
   d. Click **Save Draft** to save the page.

   Use drag and drop option:
   a. View the hidden content as described in step 1.

b. Drag the portlet or widget out of the hidden container.
c. Click **Save Draft** to save the page.

Static content

Static content is part of every website. In a portal site, static content can be rendered as static page or it can be added to specific content areas on a page.

To include, update, and administer your static content pages, you can use any of the portal administration tools, Manage Pages portlet, XML configuration interface, or the Portal Scripting Interface.

Working with static content pages has the following advantages:

- You or your web designers can create a portal page by using standard web authoring tools. This tool can be HTML editors or even simple text editors. Creating an HTML portal page requires no knowledge of JSP.
- You have more control over the layout of the page than by using the portal layout model.
- You can include portlets as dynamic elements and containers as placeholders for portlets in your pages. You can display these portlets by using server-side aggregation, Ajax, or iFrame techniques.
- You can update an existing static page by uploading a modified HTML file while you preserve the portlet customization on that page.
- Static pages can be rendered in the portal by the following two ways:
  - As stand-alone web pages that control the complete browser area.
  - As part of the portal content area. In this case, the portal still controls the banner and navigation area.
- You can deploy and manage your static content pages by using any of the portal administration tools.
- You can use portlet communication with static pages, for example by wires.

Compared to dynamic content pages based on the portal container model, static HTML pages also have the following characteristics:

- You create and administer static pages similarly as other portal pages. Some steps and options differ. For details, refer to the user interface and the portlet help of the Manage Pages portlet and its subportlets.
- To update a static page, you make the required changes in the HTML file. Then, you replace the portal page with the updated page by using the Manage Pages and Properties portlets or other portal administration tools. You can use the portal Page Customizer to update the static page layout if the static page contains portlet containers that are defined by the portlet container microformat.
- You provide national language support by bundling localized markup files into the compressed file, together with the HTML file that defines the static content. At rendering time, the portal globalization algorithms decide which locale is rendered, based on the request, on settings, and on the locals that are available.
- The portal defines a set of microformats for skins and portlet actions, such as configuring the portlet settings, portlet communication, and navigation for rendering. These microformats are styled by CSS that the static page author provides.
- Static pages can include drag actions. These actions are defined through a microformat. Users with the appropriate access rights can drag UI elements, such as portlets or pages.

- You provide JS, CSS, or image files for static pages by bundling them into a compressed file, together with the HTML file that defines the static content. These resources can then easily be referenced through relative links from the static page template.
- You can have static pages that are rendered by server-side aggregation or by client-side aggregation.
- Using skins and other graphic features with static content: When you write a static page and include it in your portal, the portal can render the page itself. But not with the visual features that you might configure for your portal. For example, you cannot encode skins within static pages. When the portal renders such a page, portlets on the page are rendered without a skin. To have portlets on static pages that are rendered with a skin, use CSS style sheets or JavaScript that use the microformats at rendering time.

The following topics describe how you create static pages in HTML, and what features you can use.

When you write the static page, you can include it in your portal by using the Manage Pages portlet.

Decision point: After you include a page in the portal, you cannot change the page from static content to standard portal layout or from standard portal layout to static content. If you want to change the page type after you create it, you need to delete the existing page and create a new page of the required type.

Note: When a static page uses the default Portal 8.5 theme, Portal 8.0 theme, or Portal 7.0.0.2 theme, users can change the style of the page, but they cannot change the layout of the page or add content to it.

"Creating a static page"
You can create a new portal page by starting with a static HTML file or an HTML fragment. If you revise the HTML, you can refresh the page to render the changes in the portal.

"Static resources" on page 1869
In addition to the HTML file that describes the page, static pages can contain resources such as images, scripts, and styles. Learn about the static resources that are available when you are using static pages.

"Dynamic page metadata" on page 1870
HTML defines some elements that refer to information that is managed as page metadata on portal pages. You can use the dynamic page metadata rewriting feature to place this information into the static HTML code automatically.

"Including static content pages in your portal" on page 1871
You can create a new portal page by starting with a static HTML file or an HTML fragment. If you revise the HTML, you can refresh the page to render the changes in the portal. To include, update, and administer your static content pages, you can use any of the portal administration tools, Manage Pages portlet, XML configuration interface, or the Portal Scripting Interface.

"Using WebDAV to manage pages and static content" on page 1877
WebDAV for IBM WebSphere Portal Express provides a simple and easy way to administer portal resources. Both administrators and users can use it.

Creating a static page

You can create a new portal page by starting with a static HTML file or an HTML fragment. If you revise the HTML, you can refresh the page to render the changes in the portal.

About this task

After you included a page in the portal, you cannot change the page from static content to standard portal layout or from standard portal layout to static content. If you want to change the page type after you created it, you must delete the existing page and create a new page of the required type.

When you use the Manage Pages and Page Properties portlets to create your static page in the portal, you select **Static Content** for the **Type of Page**. Then, select and complete the other options accordingly. For more information, see the portlet help.

Procedure

- The static HTML file can contain references to portlets, containers, and navigation. It defines the places in the portal page that host portlets or portlet containers. When the page is rendered, these places are filled by the server with the - possibly dynamic - content of portlet and with a microformat that defines metadata for these portlets. For example, portlet actions and the portlet title. For this purpose, the portal provides the following microformats:
  - The **portlet microformat** defines portlet windows and portlet actions, such as Edit default settings, Configure, Maximize, Minimize, Personalize, and Delete.
  - The **container microformat** defines portlet containers as placeholders for portlets.
  - For drag-and-drop actions, the portlet microformat can provide the drag source, and the container microformat provides the drop target.
  - The **navigation microformat** defines the navigation if your static page is rendered as a web page.

  The portlet window and portlet container microformats can contain object IDs. The server can handle these object IDs dynamically.

  When you write the static page, you can use CSS or JavaScript techniques that use the microformats to produce and render a user friendly user interface.
- You can define whether the static page is rendered as a web page or as part of a portal page:
  - To render the page as a stand-alone static page, include the `<html>` element as a root element in the markup file.
  - To render the page as part of a portal page, omit the `<html>` element.
- You can also include other resources as part of the page, such as cascading style sheets or graphic images. You must bundle all the files into a compressed file. This single compressed file is then used to create or update the static page.
- You can use portal frameworks such as Live Text with your static pages. To achieve this, include your static page as part of a dynamic portal page when you add the static page to your portal in a later step.
- To enable globalization, that is to represent your static page in different languages or locales, you bundle localized static markup files into a compressed file. For example, these can be HTML files, graphic files, such as JPGs, style sheets such as CSS or JS files. Observe the following naming convention for your localized files: For a base file *base_file_name.file_name_extension*, you must name the localized version of the file `base_file_name_`*locale*`.file_name_extension`. Example: For a base file named `my_page.html`, the English version of the file is `my_page_en.html`, and the US English version of the file is `my_page_en_us.html`. Although these files have different file names, they logically represent the same resource and are referenced by references to their base name. The portal serves the localized version of the resource when appropriate.

- To enable device support, which represents your static page for different device classes, bundle the static markup files that are device-class-specific into a compressed file. Observe the following naming conventions for your device-specific files:

  For a base file `base_file_name.file_name_extension`
  Name the device-specific version of the file `base_file_name_device.file_name_extension`. For example, for a base file named `my_page.html`, the smartphone-specific version of the file is `my_page_smartphone.html`, and the tablet-specific version of the file is `my_page_tablet.html`. Although these files have different file names, they logically represent the same resource and are referenced by references to their base name. The portal serves the device-specific version of the resource when appropriate.

  For localized, device-specific files
  For a base file named `my_page.html`, the smartphone-specific English version of the file is `my_page_smartphone_en.html`, and the smartphone-specific US English version of the file is `my_page_tablet_en_us.html`.

  These naming conventions also apply for the definition of static portal pages in HTML.

What to do next

To update a static page, you make the required changes in the HTML file, then you replace the portal page with the updated page by using the Manage Pages and Properties portlets or other portal administration tools. If the static page contains portlet containers that are defined by the portlet container microformat, you can use the portal Page Customizer to update static page layout .

Note: When you use only characters that can be encoded in ASCII in the names of the compressed file and the directories and files within the compressed file, you can use a compressed tool of your choice to create the file package. If you use characters that are not ASCII encoded, for example special characters or DBCS, in the names of the compressed file and the directories and files within the compressed file, you must create the compressed file by using the JRE tool `jar.exe`.

"Example HTML markup for defining a portal page" on page 1855
Use these code samples as examples of HTML markup to create a portal page.

"Class attributes for portlets on static pages" on page 1857
When you place a portlet on a static HTML page to be rendered by the portal, use a suitable CSS file to format the portlet. The CSS file makes use of the portlet microformat. You can make use of this microformat if you want to render portlets on your static HTML page with a skin of your choice. When you write the static page, you can use CSS or JavaScript techniques to convert the microformat into a friendly user interface. The portlet references in the static HTML page are replaced by the content of the portlet and the portlet microformat.

"Class attributes for a portlet container on static pages" on page 1859
To render a portlet container on a static page, you use a CSS file that makes use of the container microformat. One of the benefits is that users with the required access rights can later move the portlets by drag and drop.

"Class attributes for components on static pages" on page 1860
When you place a component on a static HTML page to be rendered by the

portal, use a suitable CSS file to format the portlet. The CSS file uses the component microformat. You can use this microformat if you want to render components on your static HTML page with a skin of your choice. When you write the static page, you can use CSS or JavaScript techniques to convert the microformat into a friendly user interface. The component references in the static HTML page are replaced by the content of the component and the component microformat.

"Navigation options for static pages" on page 1863
You can provide navigation for your static pages by using either the portal theme or by making use of the navigation microformat.

"Portlets for adding dynamic elements to static pages" on page 1866
If you want to add dynamic elements such as portal navigation to your static pages, you can use portlets that the portal provides.

Example HTML markup for defining a portal page:

Use these code samples as examples of HTML markup to create a portal page.

The HTML markup that you use to create a portal page has a direct effect on how the page is rendered in the portal. This topic provides examples of the HTML markup that you can use in a source file to produce various types of content in a portal page.

You can include portlets in a static HTML file by using the class attribute `portlet-window` on a `<div>` CF03 or an `<object>` element:

- Use a `<div>` element if the portlet is part of the `<body>` of the page.
- CF03 Use the `<object>` element if the portlet is part of the `<head>` element of the page.

The marked up element is replaced by the dynamic content of the portlet when the page is rendered. Unless the portlet is embedded in a container, it cannot be moved or deleted by the page customizer or other portal mechanisms. To delete such a portlet, you need to replace the static page with an updated HTML file that does not contain the portlet. In addition to the class attribute `portlet-window`, a portlet `<div>` CF03 or `<object>` element needs to contain the following information:

The parser decodes HTML documents and treats the following tags specifically:

Name attribute
This is the name of the portlet instance that is unique across the page. It is used to distinguish between different instances of the same type of portlet on the page. When a static page is updated, this instance name is used to determine if portlets need to be updated or deleted. This name is not the unique name in the portal.

Style attribute
The style attribute identifies the portlet definition, that is the type of the portlet. The attribute needs to contain the portlet definition style. The value of the style contains the object ID or the unique name of the portlet definition.

Example:

```
<div class="portlet-window" name="instancename"
     style="portlet-definition:definitionname"></div>
```

You can also parametrize portlet windows directly in the HTML document. These parameters are passed on as edit default preferences to the portlet instances at page creation or page update time. For a portlet included with a element, the parameters consist of name-value pairs that are formatted by using an HTML definition list. Example:

```
<div class="portlet-window" name="<instancename>"
     style="portlet-definition:<definitionname>">
   <dl>
      <dt>key1</dt>
      <dd>value1</dd>
      <dt>key2</dt>
      <dd>value2</dd>
   </dl>
</div>
```

CF03 For a portlet included by an <object> element, the parameters consist of name-value pairs that are formatted by using the <param> elements of the <object> element. Example:

CF03 <object class="portlet-window" name="<instancename>" style="portlet-definition:<definitionname>"> <param name="key1" value="value1"> <param name="key2" value="value2"> </object>

Portlet containers contain portlet windows that can be rearranged or deleted by a page editor, for example, the page customizer. In addition you can add new portlet windows after the page has been deployed. The portlet windows that you define in the static page as the content of the container are the portlets that are initially part of the container. Containers cannot be nested.

Similar to portlet windows, containers are marked up by using the class attribute portlet-container on a tag. In addition you need to specify the following:

Name attribute
This is the name of the container that is unique across the page. This is not the unique name in the portal.

Example:

```
<div class="portlet-container" name="holdername">
   <div class="portlet-window" name="instancename"
        style="portlet-definition:definitionname"></div>
</div>
```

Rendering a page from full HTML markup

When you include the beginning <html> and ending </html> markup in the source file, the resulting portal page is rendered without the portal theme, or surrounding navigation frame. The user sees only the layout that you code inside the HTML file. An example of the HTML markup looks like this:

```
<html>
   <head>
      <title>Sample Static Page</title>
   </head>
   <body>
      <p>This is a static page example.</p>
      <p>Welcome portlet</p>
      <div class="portlet-container" name="portletContainer1">
         <div class="portlet-window" name="portletWindow1"
              style="portlet-definition:wps.p.Welcome To WebSphere Portal">
```

```
        </div>
      </div>
   </body>
</html>
```

The values given for the name attributes must be unique in the scope of the page.

With this example, all information that the portal requires to render the page is well known at the time when you create and edit the static page.

Rendering a page from an HTML fragment

When the HTML source file is a fragment of HTML markup, and does not include the opening or closing `<html>` markup, then the page is rendered inside the portal navigation frame. An example of this HTML coding:

```
<p>This page has one Welcome Portlet.</p>
<p>Welcome portlet</p>
<div class="portlet-container" name="portletContainer1">
   <div class="portlet-window" name="portletWindow1"
        style="portlet-definition:wps.p.Welcome To WebSphere Portal">
   </div>
</div>
```

The values given for the name attributes must be unique in the scope of the page.

Class attributes for portlets on static pages:

When you place a portlet on a static HTML page to be rendered by the portal, use a suitable CSS file to format the portlet. The CSS file makes use of the portlet microformat. You can make use of this microformat if you want to render portlets on your static HTML page with a skin of your choice. When you write the static page, you can use CSS or JavaScript techniques to convert the microformat into a friendly user interface. The portlet references in the static HTML page are replaced by the content of the portlet and the portlet microformat.

Refer to the following list of class attributes for more information. An example of a portlet rendered from HTML is also provided.

portlet-window

This value of the `class` attribute identifies the containing `<div>` tag for the portlet window as it was defined by the Web designer when the static page was written. The style attribute and the name attribute contain the same information that was provided by the author of the page on the `<div>` element that referenced the portlet. The ID attribute contains the object ID of the window. Refer to the following example for rendering a page from full HTML markup:

```
<html>
   <head>
      <title>Sample Static Page</title>
   </head>
   <body>
      <p>This is a static page example.</p>
      <p>Welcome portlet</p>
      <div class="portlet-container" name="portletContainer1">
         <div class="portlet-window" name="portletWindow1"
              style="portlet-definition:wps.p.Welcome To WebSphere Portal">
         </div>
      </div>
   </body>
</html>
```

portlet-info
This value of the `class` attribute introduces the metadata section of the portlet. All metadata is encapsulated in this container so that it can be hidden it by CSS. The `xoxo` class denotes that this contains a list of items.

portlet-title
This value of the `class` attribute represents the localized title of the portlet. The `lang` attribute identifies the actual locale.

portlet-actions
This value of the `class` attribute lists the different portlet actions or operations that can be invoked in the portlet window.

Note: Such operations are not necessarily actions in the sense of HTTP. Instead, some actions are safe interactions, whereas others can be unsafe, as they modify the server. In the list of actions each interaction is either represented as a link to denote a safe interaction or as a `form` to denote an unsafe interaction.

portlet-action
This value of the `class` attribute describes the actual actions or operations. Those actions that would result in an idempotent operation are denoted by the selected attribute. In addition each action is classified by an action specific attribute to allow CSS to style these actions, for example, by adding action specific icons. The list of possible actions is computed on the server by making use of the Operations Feed. This feed is extensible, therefore new operations can be added over time.

portlet-window-body
This value of the `class` attribute denotes the body of the portlet.

iw-iWidiget
This value of the `class` attribute identifies the beginning of the microformat for placing an iWidget instance on a static page. Additional information about the iWidget microformat for including iWidgets into a broader set of markup, such as a portal page, provided by the iWidget specification and on the topic about Class attributes for iWidgets on static pages.

selected
This value of the `class` attribute identifies a selected item, either the portlet itself or an action. If it appears on the `<div>` tag for the portlet window, this means that the portlet was the target of the interaction that produced the page.

portal-drag-source
This is an optional class on the portlet window. It denotes that this portlet window can be dragged around on the screen by users. This attribute is only valid if the portlet window is part of a modifiable portlet container, and if the user has the rights to modify the page and the container is not locked. Refer to the Container Microformat to see how a drop target is represented and how the actual drag and drop operation can be performed.

Note: A drag and drop action is triggered from the client side, but is executed on the server.

xoxo This attribute denotes that the following is a list of items.

Example of rendering a portlet from HTML

The following is an example of a microformat representation for a portlet window on a page:

```
<div class='portlet-window' id='7_CGAH47L00OGRB02DA9LR6H1024' name='window2' >
   <ul class='xoxo portlet-info' >
      <li class='portlet-title' lang='en'>
         PetStorePortlet
      </li>
      <li class='portlet-actions' >
         Actions
         <ul class='xoxo portlet-action' >
            <li class='portletoperation-view selected' >
               <a href='?uri=wp.operations:onPortletModeView
                  (7_CGAH47L00OGRB02DA9LR6H1024)'
                  rel='view' >Back</a>
            </li>
            <li class='portletoperation-normal selected' >
               <a href='?uri=wp.operations:onWindowStateNormal
                  (7_CGAH47L00OGRB02DA9LR6H1024)'
                  rel='normal' >Restore</a>
            </li>
            <li class='portletoperation-minimized' >
               <a href='?uri=wp.operations:onWindowStateMinimized
                  (7_CGAH47L00OGRB02DA9LR6H1024)'
                  rel='minimized' >Minimize</a>
            </li>
            <li class='portletoperation-maximized' >
               <a href='?uri=wp.operations:onWindowStateMaximized
                  (7_CGAH47L00OGRB02DA9LR6H1024)'
                  rel='maximized' >Maximize</a>
            </li>
            <li class='portletoperation-delete_portlet' >
               <form method='POST' action='?uri=wp.operations:onDeletePortlet
                     (7_CGAH47L00OGRB02DA9LR6H1024)'
                     rel='delete_portlet' >Delete</form>
            </li>
         </ul>
      </li>
   </ul>
   <div class='portlet-window-body' >
   </div>
</div>
```

Class attributes for a portlet container on static pages:

To render a portlet container on a static page, you use a CSS file that makes use of the container microformat. One of the benefits is that users with the required access rights can later move the portlets by drag and drop.

To place a portlet container on a static page, use a style file that makes use of the container microformat. Refer to the following list of class attributes and the example of rendering a portlet container from HTML.

portlet-container

This value of the `class` attribute identifies the containing `<div>` tag for a portlet container as it was defined by the Web designer when the static page was written. The `ID` attribute contains the Object ID of the container, which is globally unique in the portal. The `name` attribute contains a name for the container that is unique only in the scope of the hosting page. In static page aggregation the HTML page designer determines this name.

portal-drop-target
This value of the `class` attribute identifies the container as a drop target of a drag and drop operation. Only modifiable containers use this class attribute.

drop-handler
This value of the `class` attribute identifies the `form` that represents the callback handler for drag and drop. The action contains a URL that uniquely identifies the drop target. You can add additional hints and the drag sources dynamically to the form by using Javascript.

For the drag source, use the attribute from the portlet microformat when defining the portlet:

portal-drag-source
This is an optional class on the portlet window. It denotes that this portlet window can be dragged around on the screen by users. This attribute is only valid if the portlet window is part of a modifiable portlet container, and if the user has the rights to modify the page and the container is not locked. Refer to the Container Microformat to see how a drop target is represented and how the actual drag and drop operation can be performed.

Note: A drag and drop action is triggered from the client side, but is executed on the server.

Example of rendering a portlet container from HTML

The following is an example of a microformat representation for a portlet container on a page:

```
<div id="content-area">
   <div class='portlet-container portal-drop-target'
        id='7_CGAH47L0008K402D2V3F7I2005' name='c1' >
      <form class='drop-handler' enctype='multipart/form-data' method='POST'
            action='/wps/mycontenthandler/!ut/p/dnd/lm:
                 oid:7_CGAH47L0008K402D2V3F7I2005@
                 oid:6_CGAH47L0008K402D2V3F7I2000?uri=dnd%3alm%3a
                 oid%3a7_CGAH47L0008K402D2V3F7I2005%40
                 oid%3a6_CGAH47L0008K402D2V3F7I2000'>
         <input type='hidden' name='_charset_'>
      </form>
   </div>

   ... the actual portlets come here

</div>
```

Class attributes for components on static pages:

When you place a component on a static HTML page to be rendered by the portal, use a suitable CSS file to format the portlet. The CSS file uses the component microformat. You can use this microformat if you want to render components on your static HTML page with a skin of your choice. When you write the static page, you can use CSS or JavaScript techniques to convert the microformat into a friendly user interface. The component references in the static HTML page are replaced by the content of the component and the component microformat.

For more information, see the following list of class attributes for portlet components. An example of a component that is rendered from HTML is also provided.

portlet-window / component-control
This value of the `class` attribute identifies the containing `<div>` tag for the component window as it was defined by the web designer when the static page was written. The style attribute and the name attribute contain the same information that was provided by the author of the page on the `<div>` element that referenced the component. The attribute `id-`*`ObjectID`* is the prefixed object ID of the component window, which is globally unique in the portal. Refer to the following example for rendering a page from full HTML markup:

```
<html>
   <head>
      <title>Sample Static Page</title>
   </head>
   <body>
      <p>This is a static page example.</p>
      <p>Welcome component</p>
      <div class="component-container" name="componentContainer1">
         <div class="portlet-window component-control" name="componentWindow1"
              id="Z7_2QC68B1A08A0B0IAUQ7VBO20G3"
              style="portlet-definition:wps.p.Welcome To WebSphere Portal">
         </div>
      </div>
   </body>
</html>
```

portlet-info
This value of the `class` attribute introduces the metadata section of the portlet. All metadata is encapsulated in this container so that it can be hidden by CSS. The `xoxo` class denotes that this contains a list of items.

portlet-title
This value of the `class` attribute represents the localized title of the portlet. The `lang` attribute identifies the actual locale.

portlet-actions
This value of the `class` attribute lists the different portlet actions or operations that can be started in the portlet window.

Note: Such operations are not necessarily actions in the sense of HTTP. Instead, some actions are safe interactions, whereas others can be unsafe, as they modify the server. In the list of actions each interaction is either represented as a link to denote a safe interaction or as a `form` to denote an unsafe interaction.

portlet-action
This value of the `class` attribute describes the actual actions or operations. Those actions that would result in an idempotent operation are denoted by the selected attribute. In addition, each action is classified by an action-specific attribute to allow CSS to style these actions, for example, by adding action-specific icons. The list of possible actions is computed on the server by using the Operations Feed. This feed is extensible, and therefore new operations can be added over time.

portlet-window-body
This value of the `class` attribute denotes the body of the portlet.

iw-iWidget
This value of the `class` attribute identifies the beginning of the

microformat for placing an iWidget instance on a static page. Additional information about the iWidget microformat for including iWidgets into a broader set of markup, such as a portal page, provided by the iWidget specification and on the topic about Including iWidgets in a static page.

selected
This value of the `class` attribute identifies a selected item, either the portlet itself or an action. If it appears on the `<div>` tag for the portlet window, this means that the portlet was the target of the interaction that produced the page.

portal-drag-source
This value is an optional class on the portlet window. It denotes that this portlet window can be dragged around on the screen by users. This attribute is valid only if the portlet window is part of a modifiable portlet container, and if the user has the rights to modify the page and the container is not locked. Refer to the Container Microformat to see how a drop target is represented and how the actual drag and drop operation can be done.

Note: A drag and drop action is triggered from the client side, but is run on the server.

xoxo This attribute denotes that the following is a list of items.

Example of rendering a component from HTML

The following is an example of a microformat representation for a portlet window on a page:

```
<div class='portlet-window' id='7_CGAH47L00OGRB02DA9LR6H1024' name='window2' >
   <ul class='xoxo portlet-info' >
      <li class='portlet-title' lang='en'>
         PetStorePortlet
      </li>
      <li class='portlet-actions' >
         Actions
         <ul class='xoxo portlet-action' >
            <li class='portletoperation-view selected' >
               <a href='?uri=wp.operations:onPortletModeView
                  (7_CGAH47L00OGRB02DA9LR6H1024)'
                  rel='view' >Back</a>
            </li>
            <li class='portletoperation-normal selected' >
               <a href='?uri=wp.operations:onWindowStateNormal
                  (7_CGAH47L00OGRB02DA9LR6H1024)'
                  rel='normal' >Restore</a>
            </li>
            <li class='portletoperation-minimized' >
               <a href='?uri=wp.operations:onWindowStateMinimized
                  (7_CGAH47L00OGRB02DA9LR6H1024)'
                  rel='minimized' >Minimize</a>
            </li>
            <li class='portletoperation-maximized' >
               <a href='?uri=wp.operations:onWindowStateMaximized
                  (7_CGAH47L00OGRB02DA9LR6H1024)'
                  rel='maximized' >Maximize</a>
            </li>
            <li class='portletoperation-delete_portlet' >
               <form method='POST' action='?uri=wp.operations:onDeletePortlet
                     (7_CGAH47L00OGRB02DA9LR6H1024)'
                     rel='delete_portlet' >Delete</form>
            </li>
         </ul>
```

```
      </li>
   </ul>
   <div class='portlet-window-body' >
   </div>
</div>
```

Navigation options for static pages:

You can provide navigation for your static pages by using either the portal theme or by making use of the navigation microformat.

Portal theme to provide navigation

To provide navigation for static pages by using the portal theme, remove the `<html>` and `<head>` tags from your static page content. If you do that, the portal renders the content as navigation.

Navigation microformat to provide navigation

If you have full static HTML pages with beginning and ending `<html>` tags in your portal, you might want to include links to other pages in the portal. In this case you write your HTML code so that it includes a navigation portlet in the page and makes use of the navigation microformat. Refer to the following list of class attributes and the example of the microformat navigation option.

first This attribute denotes the first child in a list. This is required for CSS styling.

last This attribute denotes the last child in a list. This is required for CSS styling.

expanded
This attribute denotes an expanded node. A node can only be expanded if it has children and its navigational state is set to **expanded**.

collapsed
This attribute denotes a collapsed node. A node can only be collapsed if it has children and its navigational state is set to **collapsed**.

page-actions
This attribute lists the actions that are available on the page. Typically, this list contains the actions used to expand or collapse the navigation nodes. For a page that is currently selected in the navigation, this list also contains the actions that are available for that page.

selected
This attribute denotes the page that is currently selected in the navigation.

pageoperation-expand
This attribute expands a collapsed node.

pageoperation-collapse
This attribute collapses an expanded node.

Example of the microformat navigation option

The following is an example of a microformat representation for a navigation for a page:

```
<div class="xoxo portal-navigation">
  <ul>
    <li class="first expanded">
      <a href='/wps/myportal/!ut/p/c5/ . . . . . '>Home</a>
```

```
  <ul>
    <li class="first">
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Web 2.0 Introduction</a>
    </li>
    <li>
      <a href='/wps/myportal/!ut/p/c5/e/'>Web 2.0 Portlets</a>
    </li>
    <li class="collapsed">
      <ul class="xoxo page-actions">
        <li class="pageoperation-expand">
          <a href='/wps/myportal/!ut/p/c5/e/'>Expand</a>
        </li>
      </ul>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Static Page Aggregation</a>
    </li>
    <li>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Navigation Page</a>
    </li>
    <li>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Nav1</a>
    </li>
    <li>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>IBM</a>
    </li>
    <li>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>dnd</a>
    </li>
    <li>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>derivable</a>
    </li>
    <li>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>deriv1</a>
    </li>
    <li class="selected last">
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Container Test</a>
    </li>
  </ul>
</li>
<li class="collapsed">
  <ul class="xoxo page-actions">
    <li class="pageoperation-expand">
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Expand</a>
    </li>
  </ul>
  <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Administration</a>
</li>
<li>
  <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Resource Policy Editor</a>
</li>
<li>
  <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Resource Policy Editor CA</a>
</li>
<li>
  <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Domino Integration</a>
</li>
<li class="collapsed">
  <ul class="xoxo page-actions">
    <li class="pageoperation-expand">
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Expand</a>
    </li>
  </ul>
  <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Templates</a>
</li>
<li class="collapsed">
  <ul class="xoxo page-actions">
    <li class="pageoperation-expand">
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Expand</a>
    </li>
  </ul>
  <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Site
  Map</a>
</li>
<li class="collapsed">
  <ul class="xoxo page-actions">
    <li class="pageoperation-expand">
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Expand</a>
    </li>
  </ul>
  <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>About</a>
</li>
```

```
    <li>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Login</a>
    </li>
    <li>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Edit My Profile</a>
    </li>
    <li class="collapsed">
      <ul class="xoxo page-actions">
        <li class="pageoperation-expand">
          <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Expand</a>
        </li>
      </ul>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Page Customizer</a>
    </li>
    <li>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Page Properties</a>
    </li>
    <li>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Template Parameters</a>
    </li>
    <li>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Template and Application Properties</a>
    </li>
    <li>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Template and Application Layout</a>
    </li>
    <li>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Application Roles</a>
    </li>
    <li>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Application Membership</a>
    </li>
    <li>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Policy Status</a>
    </li>
    <li>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Organize Favorites</a>
    </li>
    <li>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Search Seedlist</a>
    </li>
    <li>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>People Palette</a>
    </li>
    <li>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Content Palette</a>
    </li>
    <li class="collapsed">
      <ul class="xoxo page-actions">
        <li class="pageoperation-expand">
          <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Expand</a>
        </li>
      </ul>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Quick Links</a>
    </li>
    <li class="collapsed">
      <ul class="xoxo page-actions">
        <li class="pageoperation-expand">
          <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Expand</a>
        </li>
      </ul>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Theme
      Links</a>
    </li>
    <li>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Open Tasks</a>
    </li>
    <li class="last collapsed">
      <ul class="xoxo page-actions">
        <li class="pageoperation-expand">
          <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Expand</a>
        </li>
      </ul>
      <a href='/wps/myportal/!ut/p/c5/ . . . . . /'>Application Root</a>
    </li>
  </ul>
</div>
```

Inline navigation

The inline navigation feature can be enabled by setting the page metadata **spa.ex.anchor.enabled** to `true`. The default is `false`. With this feature, the markup of the static page can contain `<a>` links that point to different portal pages that use relative URLs. The **href** attribute is rewritten at rendering time and replaced by the portal URL that points to this page. The relative URLs in the source markup use the friendly path of the current page as the base URL. The relative reference is applied to this base URL and the resulting friendly path is used to address the target page. This feature works only if both the current page and the targeted page have a friendly path.

For example, if the current page has the friendly path `/home/mypage` and contains the following markup; `<a href="child">child_page</a>`, then this markup is rewritten to a portal URL that points to the page with the friendly name `/home/mypage/child`.

Notes:

- If the **href** attribute of an anchor link contains an absolute URL, no rewriting occurs and the link continues to point to the absolute URL.
- If the **href** attribute of an anchor link contains a URL that starts with `/`, then the rewriting logic resolves the URL as a friendly path. If no page for this path exists, the system assumes a link to another application on the same server and leaves the value of the **href** attribute untouched.
- If the **href** attribute of an anchor link starts with `?`, then the remainder is considered query parameters and no rewriting occurs.

Related tasks:

"Using friendly URLs" on page 1280
You can associate friendly URLs with portal pages and labels. You and your users can use these friendly URLs to access specific portal pages or labels by using a human readable path, which is easy to remember.

Portlets for adding dynamic elements to static pages:

If you want to add dynamic elements such as portal navigation to your static pages, you can use portlets that the portal provides.

WebSphere Portal Express provides the following portlets for static pages. These portlets are included in a default portal installation. You can address them by using their unique names.

Navigation portlet
: The Navigation portlet adds portal navigation to a page. When writing a static portal page you can configure the starting point of the navigation and the number of navigation levels by using the initialization parameters.

Breadcrumb Trail portlet
: The Breadcrumb Trail portlet adds the path from the content root down to the currently selected static page.

Page List portlet
: The Page List portlet adds an arbitrary list of links to pages to a static page.

For more details about these portlets and how you can use them refer to the following topics.

"The Navigation portlet"
The Navigation portlet allows you to add dynamic portal navigation to a static page. When writing a static portal page you can configure the starting point of the navigation and the number of navigation levels by using the initialization parameters.

"The Breadcrumb Trail portlet" on page 1868
The Breadcrumb Trail portlet allows you to add the navigation path to your static pages. The breadcrumb trail starts at the content root and goes down to the currently selected static page.

"The SPA Resource Addressability portlet" on page 1869
The SPA Resource Addressability portlet allows you to add a dynamically computed list of links to portal resources to a static page. This list is produced dynamically via a POCURI (Piece Of Content URI).

The Navigation portlet:

The Navigation portlet allows you to add dynamic portal navigation to a static page. When writing a static portal page you can configure the starting point of the navigation and the number of navigation levels by using the initialization parameters.

Unique name

You can address the Navigation Portlet by its unique name: `wps.p.SpaNavigation` .

Usage

You can embed the navigation portlet into a static page by using the semantic tag `portlet-window` described in the topic about class attributes for portlets on static pages. When rendering the page, the server replaces the semantic tag with the portlet microformat, and the portlet renders the navigation by using the navigation microformat. The portlet accepts the following configuration parameters that you can embed by coding a definition list (`<dl>`) in the static page:

root This identifies the node where the navigation starts that the portlet renders. This parameter accepts a path-like expression that is relative to the current selection. Path segments can include the OID or unique name of a node. The period character (**.**) or the double period (**..**) have the same meaning as in relative URLs. If you omit this parameter, the navigation starts at the current node.

levels The number of child levels that start at the node that is identified by the `root` parameter. A value of `-1` means all levels. This is also the default if you omit this parameter.

Example

The following example produces a single level navigation of the siblings of the current page:

```
<div class="portlet-window" name="main-navigation"
     style="portlet-definition:wps.p.SpaNavigation">
     <dl>
     <dt>root</dt>
     <dd>..</dd>
     <dt>levels</dt>
     <dd>1</dd>
     </dl>
</div>
```

The following example shows the quick link navigation:

```
<div class="portlet-window" name="quick-navigation"
     style="portlet-definition:wps.p.SpaNavigation">
     <dl>
     <dt>root</dt>
     <dd>ibm.portal.Quick Links</dd>
     </dl>
</div>
```

Related reference:

"Class attributes for portlets on static pages" on page 1857
When you place a portlet on a static HTML page to be rendered by the portal, use a suitable CSS file to format the portlet. The CSS file makes use of the portlet microformat. You can make use of this microformat if you want to render portlets on your static HTML page with a skin of your choice. When you write the static page, you can use CSS or JavaScript techniques to convert the microformat into a friendly user interface. The portlet references in the static HTML page are replaced by the content of the portlet and the portlet microformat.

The Breadcrumb Trail portlet:

The Breadcrumb Trail portlet allows you to add the navigation path to your static pages. The breadcrumb trail starts at the content root and goes down to the currently selected static page.

Unique name

You can address the Breadcrumb Trail portlet by its unique name: `wps.p.SpaBreadCrumbTrail` .

Usage

You can embed the Breadcrumb Trail portlet into a static page by using the semantic tag `portlet-window` described in the topic about class attributes for portlets on static pages. When rendering the page, the server replaces the tag with the portlet microformat, and the portlet renders the breadcrumb trail by using the navigation microformat.

Example

The following example produces a breadcrumb trail navigation for the current page:

```
<div class="portlet-window" name="breadcrumb"
     style="portlet-definition:wps.p.SpaBreadCrumbTrail">
</div>
```

Related reference:

"Class attributes for portlets on static pages" on page 1857
When you place a portlet on a static HTML page to be rendered by the portal, use a suitable CSS file to format the portlet. The CSS file makes use of the portlet microformat. You can make use of this microformat if you want to render portlets on your static HTML page with a skin of your choice. When you write the static page, you can use CSS or JavaScript techniques to convert the microformat into a friendly user interface. The portlet references in the static HTML page are replaced by the content of the portlet and the portlet microformat.

The SPA Resource Addressability portlet:

The SPA Resource Addressability portlet allows you to add a dynamically computed list of links to portal resources to a static page. This list is produced dynamically via a POCURI (Piece Of Content URI).

Unique name

You can address the SPA Resource Addressability portlet by its unique name: `wps.p.SpaResourceList` .

Usage

You can embed the SPA Resource Addressability portlet into a static page by using the semantic tag `portlet-window` described in the topic about class attributes for portlets on static pages. When rendering the page, the server replaces the tag with the portlet microformat, and the portlet renders the list of page links by using the navigation microformat. The portlet accepts the following configuration parameters that you can embed by coding a definition list (`<dl>`) in the static page:

uri This is the POCURI to an ATOM feed. The list of resource links is the list of alternate links in this ATOM feed.

<any parameters>
All other parameters are taken as parameters for the POC (Piece Of Content) DataSource that produces the ATOM feed.

Related reference:

"Class attributes for portlets on static pages" on page 1857
When you place a portlet on a static HTML page to be rendered by the portal, use a suitable CSS file to format the portlet. The CSS file makes use of the portlet microformat. You can make use of this microformat if you want to render portlets on your static HTML page with a skin of your choice. When you write the static page, you can use CSS or JavaScript techniques to convert the microformat into a friendly user interface. The portlet references in the static HTML page are replaced by the content of the portlet and the portlet microformat.

Static resources

In addition to the HTML file that describes the page, static pages can contain resources such as images, scripts, and styles. Learn about the static resources that are available when you are using static pages.

Linked resources

If the static page contains the `<html>` tag, a full page is represented. The `<head>` element of the page might contain `<link>` tags that point to static resources such as CSS style sheets and `<script>` tags that point to script resources. To efficiently point to resources embedded in the static page, enable the rewriting feature for these tags by modifying the following page metadata:

- Set **`spa.ex.link.enabled`** to `true`. The default is `false`.
- Set **`spa.ex.script.enabled`** to `true`. The default is `false`.

If the previous metadata is set to `true`, the `href` and `src` attributes are rewritten in the following ways:

- If the attribute contains an absolute URL, the URL is rewritten to point to the Ajax proxy server.

- If the attribute contains a relative URL, the URL is rewritten to point to an access point that serves the resource out of the static page container for the page.

In the following example, assume that the page contains the following resources:

- `index.html`
- `images/logo.gif`
- `css/styles.css`
- `script/script.js`

The following source code would enable the use of these resources:

```
<html>
<head>
 <link rel="shortcut icon" href="images/logo.gif">
 <link rel="stylesheet" href="css/style.css">
 <script src="script/script.js"></script>
</head>

...
</html>
```

The rewriting of the link occurs at rendering time. Therefore, the resulting URLs that point to the static resources provide an efficient way to serve them.

Images

To efficiently reference images from the static HTML source, enable the rewriting feature for image tags by modifying the following page metadata:

- Set **`spa.ex.image.enabled`** to `true`. The default is `false`.

If the previous metadata is set to `true`, the `src` attribute for `<img>` tags is rewritten in the following ways:

- If the attribute contains an absolute URL, the URL is rewritten to point to the Ajax proxy server.
- If the attribute contains a relative URL, the URL is rewritten to point to an access point that serves the resource out of the static page container for the page.

Resource sharing

Often, multiple static pages belong together in the form of an application. If so, multiple resources can be shared across the application. The mechanism that is used to rewrite resources searches across the parent page hierarchy, beginning with the current page. If a resource, for example, an image, cannot be found on one level, the fallback mechanism searches the next level. This mechanism enables the sharing of resources in a parent page that can be reused by child pages.

Note: Child pages do not have to have a special syntax to enable this sharing mechanism. The child pages contain a relative URL as if the resource were contained in the page itself.

Dynamic page metadata

HTML defines some elements that refer to information that is managed as page metadata on portal pages. You can use the dynamic page metadata rewriting feature to place this information into the static HTML code automatically.

The dynamic page metadata rewriting feature affects the HTML tags <base>, <title>, and <meta>. Enable the rewriting for each of these tags by modifying the following page metadata:

- Set **spa.ex.base.enabled** to true. The default is false.
- Set **spa.ex.title.enabled** to true. The default is false.
- Set **spa.ex.meta.enabled** to true. The default is false.

If you enable the previous settings, the following rewrites occur:

spa.ex.base.enabled
The href attribute of the <base> tag is replaced by the base URL to the current page.

spa.ex.title.enabled
The text value of the <title> tag is replaced by the page title that is using the currently requested location.

spa.ex.meta.enabled
If the <meta> tag contains an http-equiv attribute, the content attribute specifies the requested HTTP response header.

If the <meta> tag contains a name attribute, the rewriting depends on the value of the attribute. Depending upon the value of the name attribute, the content attribute is replaced by one of the following options:

description
The description of the portal page in the currently requested location.

csrf_param
The name of the **FORM** parameter for CSRF protection.

csrf_token
The CSRF token for the current request.

generator
The name or version of the IBM portal server that is hosting the page.

Including static content pages in your portal

You can create a new portal page by starting with a static HTML file or an HTML fragment. If you revise the HTML, you can refresh the page to render the changes in the portal. To include, update, and administer your static content pages, you can use any of the portal administration tools, Manage Pages portlet, XML configuration interface, or the Portal Scripting Interface.

About this task

After you have included a page in the portal, you cannot change the page from static content to standard portal layout or from standard portal layout to static content. If you want to change the page type after you have created it, you need to delete the existing page and create a new page of the required type.

When you use the Manage Pages and Page Properties portlets to create your static page in the portal, you select **Static Content** for the **Type of Page**. Then select and complete the other options accordingly. For more details refer to the portlet help.

To update a static page, you make the required changes in the HTML file, then you replace the portal page with the updated page by using the Manage Pages and Properties portlets or other portal administration tools. You can use the portal Page

Customizer to update static page layout if the static page contains portlet containers defined by the portlet container microformat.

"Exporting and importing static pages"
You can work with static portal pages by using the portal XML configuration interface. Learn about the tasks that you can perform and the XML elements for working with static pages.

"Scripting for static pages" on page 1875
You can work with static portal pages by using the Portal Scripting Interface, which enables you to use administration function through the Jacl scripting language. Get familiar with the scripting commands for working with static pages.

Exporting and importing static pages:

You can work with static portal pages by using the portal XML configuration interface. Learn about the tasks that you can perform and the XML elements for working with static pages.

Exporting a static content page

Exporting a static page by using the XML configuration interface works like exporting any other portal page. For details about how to do this refer to the topics about the XML configuration interface.

Importing a static content page

You can import a static content page into your portal by using one of the following three methods:

- Specify the static content page in binary content format
- Have the static content page in a archive or compressed file, and reference that file from the XML import script
- Create the content page from a layout template that is installed in the portal.

"Importing a static page in binary format" on page 1873
You can import a static content page in binary encoded content format by using the XML configuration interface.

"Importing a static page from an archive or compressed file" on page 1874
You can import a static content page from an archive or compressed file by using the XML configuration interface.

"Importing a static page from an installed template" on page 1874
You can import a static content page from an installed layout template by using the XML configuration interface.

Related tasks:

"Exporting pages or page hierarchies by using the Manage Pages portlet" on page 1063
You can do an XML export of a page or an entire page hierarchy by using the Manage Pages portlet.

"Importing pages or page hierarchies by using the XML Import portlet" on page 1063
You can import an XML configuration file by using the XML Import portlet.

Related reference:

"Importing static page content from archive or compressed files" on page 1107
You can import the content of static pages from an external archive or compressed file by using the XML configuration interface.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Importing a static page in binary format:

You can import a static content page in binary encoded content format by using the XML configuration interface.

You can adapt the following example and use it to import a static page. This example shows the following additional possibilities:

- It assumes that the portlet is already installed. Therefore it uses only the `locate` action for the Web module, and not the `update` action.
- It shows how you can specify the content of static page in binary format. To obtain the binary format content, export the page by using the XML configuration interface.
- You can export the result to generate a template for your XML scripts.

Sample file CreateStaticPage.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<request xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" update="true"
    xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd" create-oids="true">
   <portal action="locate">
        <web-app action="locate" uid="ilwwcm-localrenderingportlet-jsr.war.webmod">
           <portlet-app action="locate" uid="ilwwcm-localrenderingportlet-jsr.war">
                <portlet action="locate" objectid="theExamplePortlet" name="Web Content Viewer (JSR 286)">
                </portlet>
            </portlet-app>
        </web-app>

        <content-node action="locate" objectid="parentPage" uniquename="ibm.portal.Home"/>
        <content-node action="update" active="true" allportletsallowed="true" content-parentref="parentPage"
                      create-type="explicit" domain="rel" ordinal="last" type="staticpage"
                      uniquename="ibm.portal.ssa.SamplePage.2">
            <localedata locale="en" prefix="page.sample">
               <title>Sample Page 1</title>
            </localedata>

            <pagecontents markup="html" display-option="inline">
                <content>UEsDBBQACAAIALKbTDcAAAAAAAAAAAAAAAALAAAAaW5kZXgxLmh0bWxtUE1rwzAMvQf6H0TuTeh1
                         ZL7stFsghZ5VW2sEjmxir1n//YTdbDAGOoj3IelpmAmdOTRD5uzJTLhETzBlzGxhxBsNfWVU0u/a
                         a3APc9AmmvPMCbQQUvVE9QB9lTkdFLpAMyYIQhDDquPyu6heLMGFvA0LwVjxDurcHX7Khz6WhY7v
                         YD2m9No+maMNkpGF1hYEF/oh3nb81Jr/nRuLC9sf26WAp1bzPPwvfHT0wcKZg7xsMXWx2y88B81w
                         neJMa02BvizsdWO5uTZNiTV6wkRw58QZPvVtNz2xhFNdfWvzDVBLBwjDeEQR4AAAAJMBAABQSwEC
                         FAAUAAgACACym0w3w3hEEeAAAACTAQAACwAAAAAAAAAAAAAAAAAAAAAAAaW5kZXgxLmh0bWxQSwUG
                         AAAAAAEAAQA5AAAAGQEAAAAA
                </content>
            </pagecontents>

            <parameter name="com.ibm.portal.bookmarkable" type="string"
                       update="set"><![CDATA[Yes]]&gt;</parameter>
            <parameter name="com.ibm.portal.friendly.name" type="string"
                       update="set"><![CDATA[staticpage2]]&gt;</parameter>
            <parameter name="com.ibm.portal.static.page.file.name.html" type="string"
                       update="set"><![CDATA[index1.html]]&gt;</parameter>
            <access-control externalized="false" owner="uid=wpsadmin,o=defaultwimfilebasedrealm" private="false"/>
            <component action="update" active="true" domain="rel" ordinal="100" orientation="V" type="container">
                <component action="update" active="true" domain="rel" ordinal="100" orientation="V" type="container"/>
                <component action="update" active="true" domain="rel" ordinal="100" orientation="H" type="container">
                    <parameter name="com.ibm.portal.layoutnode.localname" type="string"
                               update="set"><![CDATA[portletContainer1]]&gt;</parameter>
                    <component action="update" active="true" domain="rel" ordinal="100" type="control">
                        <parameter name="com.ibm.portal.layoutnode.localname" type="string"
                                   update="set"><![CDATA[portletWindow1]]&gt;</parameter>
                        <portletinstance action="update" domain="rel" portletref="theExamplePortlet" />
                    </component>
```

```
            </component>
          </component>
        </content-node>
    </portal>
</request>
```

Importing a static page from an archive or compressed file:

You can import a static content page from an archive or compressed file by using the XML configuration interface.

You can adapt the following example and use it to import a static page. This example shows the following additional possibilities:

- It assumes that the portlet is already installed. Therefore it uses only the `locate` action for the Web module, and not the `update` action.
- It shows how you can specify the content of static page from a compressed file of file type `.zip` .

Sample file CreatePageFromZip.xml

```
<request xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" type="update"
    xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd" create-oids="true">
    <portal action="locate">
        <web-app action="locate" uid="ilwwcm-localrenderingportlet-jsr.war.webmod">
           <portlet-app action="locate" uid="ilwwcm-localrenderingportlet-jsr.war">
                <portlet action="locate" objectid="theExamplePortlet" name="Web Content Viewer (JSR 286)">
                </portlet>
            </portlet-app>
        </web-app>

        <content-node action="locate" objectid="parentPage" uniquename="ibm.portal.Home"/>
        <content-node action="update" active="true" allportletsallowed="true" content-parentref="parentPage"
                      create-type="explicit" domain="rel" ordinal="last" type="staticpage"
                      uniquename="ibm.portal.ssa.SamplePage.2">
            <localedata locale="en" prefix="page.sample">
               <title>Sample Static Page 1</title>
            </localedata>

            <pagecontents markup="html" display-option="inline">
                <url>file:///$server_root$/base/wp.xml/doc/xml-samples/index1.zip</url>
            </pagecontents>

            <parameter name="com.ibm.portal.bookmarkable" type="string"
                       update="set"><![CDATA[Yes]]&gt;</parameter>
            <parameter name="com.ibm.portal.friendly.name" type="string"
                       update="set"><![CDATA[staticpage2]]&gt;</parameter>
            <parameter name="com.ibm.portal.static.page.file.name.html"
                       type="string" update="set"><![CDATA[index1.html]]&gt;</parameter>
            <access-control externalized="false" owner="uid=wpsadmin,o=defaultwimfilebasedrealm" private="false"/>
            <component action="update" active="true" domain="rel" ordinal="100" orientation="V" type="container">
                <component action="update" active="true" domain="rel" ordinal="100" orientation="V" type="container"/>
                <component action="update" active="true" domain="rel" ordinal="100" orientation="H" type="container">
                    <parameter name="com.ibm.portal.layoutnode.localname" type="string"
                               update="set"><![CDATA[portletContainer1]]&gt;</parameter>
                    <component action="update" active="true" domain="rel" ordinal="100" type="control">
                        <parameter name="com.ibm.portal.layoutnode.localname" type="string"
                                   update="set"><![CDATA[portletWindow1]]&gt;</parameter>
                        <portletinstance action="update" domain="rel" portletref="theExamplePortlet" />
                    </component>
                </component>
            </component>
        </content-node>
    </portal>
</request>
```

Importing a static page from an installed template:

You can import a static content page from an installed layout template by using the XML configuration interface.

You can adapt the following example and use it to import a static page. It assumes that the portlet is already installed. Therefore it uses only the `locate` action for the Web module, and not the `update` action. For more information about layout templates refer to the topic about Theme layout templates.

Sample file CreatePageFromTemplate.xml

```
<request xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" type="update"
    xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd" create-oids="true">
    <portal action="locate">
        <web-app action="locate" uid="ilwwcm-localrenderingportlet-jsr.war.webmod">
           <portlet-app action="locate" uid="ilwwcm-localrenderingportlet-jsr.war">
                <portlet action="locate" objectid="theExamplePortlet" name="Web Content Viewer (JSR 286)">
                </portlet>
            </portlet-app>
        </web-app>

        <content-node action="locate" objectid="parentPage" uniquename="ibm.portal.Home"/>
        <content-node action="update" active="true" allportletsallowed="true" content-parentref="parentPage"
                      create-type="explicit" domain="rel" ordinal="last" type="staticpage"
                      uniquename="ibm.portal.ssa.SamplePage.2">
            <localedata locale="en" prefix="page.sample">
                <title>Sample Static Page 1</title>
            </localedata>

     <pagecontents markup="html" display-option="inline"/>

            <parameter name="com.ibm.portal.bookmarkable" type="string"
                       update="set"><![CDATA[Yes]]&gt;</parameter>
            <parameter name="com.ibm.portal.friendly.name" type="string"
                       update="set"><![CDATA[staticpage2]]&gt;</parameter>
            <parameter name="com.ibm.portal.layout.template.ref" type="string"
                       update="set"><![CDATA[dav:fs-type1/layout-templates/2ColumnEqual]]&gt;</parameter>
     <access-control externalized="false" owner="uid=wpsadmin,o=defaultwimfilebasedrealm" private="false"/>
            <component action="update" active="true" domain="rel" ordinal="100" orientation="V" type="container">
                <component action="update" active="true" domain="rel" ordinal="100" orientation="V" type="container"/>
                <component action="update" active="true" domain="rel" ordinal="100" orientation="H" type="container">
                    <parameter name="com.ibm.portal.layoutnode.localname" type="string"
                               update="set"><![CDATA[portletContainer1]]&gt;</parameter>
                    <component action="update" active="true" domain="rel" ordinal="100" type="control">
                        <parameter name="com.ibm.portal.layoutnode.localname" type="string"
                                   update="set"><![CDATA[portletWindow1]]&gt;</parameter>
                        <portletinstance action="update" domain="rel" portletref="theExamplePortlet" />
                    </component>
                </component>
            </component>
        </content-node>
    </portal>
</request>
```

Related concepts:

"Understanding the Portal Version 8.5 modularized theme" on page 2521
Modern websites and browsers enable incredible new capabilities that can greatly enhance your user's web experiences. However, these capabilities are not without cost in terms of large page sizes and more processing in the browser when each page is rendered. These capabilities are worth it when you need them, but removing them for an entire site or including them only on pages that take advantage of these capabilities provides for more flexibility.

Scripting for static pages:

You can work with static portal pages by using the Portal Scripting Interface, which enables you to use administration function through the Jacl scripting language. Get familiar with the scripting commands for working with static pages.

Including a static page in the portal

To include a static page in the portal, use the following command:

`$Content create staticpage title markup compressedfilename filename [displayoption] [select]`

Example: `$Content create staticpage MyStaticPageTitle html c:/tmp/StaticContentPage.zip index.html inline select`

This creates a static page underneath the currently selected content node for the html markup with the page title MyStaticPageTitle. The content of the page is read from the file `c:/tmp/StaticContentPage.zip`, and the entry point for the page display is read from the file `index.html`, which needs to be contained in the compressed file. To specify the display method, you can use the optional parameter `displayoption`. It takes one of the values `inline`, `iframe`, or `ajax`. The default value is `inline`. To make the newly created static page the currently selected content node, use the optional parameter `select`

Getting the static page content as a compressed file

To get a static page content in the format of a compressed file, use the following command:

`$Content pageget oid markup compressedfilename`

Example: `$Content pageget 6_CGAH47L00G2N802TJFV58Q3000 html c:/tmp/MyStaticContentPage.zip`

This writes the content of the specified static page to the compressed file `c:/tmp/MyStaticContentPage.zip`.

Setting the static page content by specifying a compressed filename

To set the static page content by specifying a compressed filename, use the following command:

`$Content pageset oid markup compressedfilename filename`

Example: `$Content pageset 6_CGAH47L00G2N802TJFV58Q3000 html c:/tmp/NewStaticContentPage.zip index.html`

This updates the specified static page content with the content of the compressed file `c:/tmp/NewStaticContentPage.zip`. The entry point for the page display is read from the file `index.html`. This file needs to be contained in the compressed file.

Getting attributes of a static page

To get attributes of a static page, use the following command:

`$Content get oid attribute markup`

Valid attributes are as follows:

filename
Use this attribute for getting the file name of the static page layout file in the compressed file.

displayoption
Use this attribute for the markup, for example HTML.

Example: `$Content get 6_CGAH47L00G2N802TJFV58Q3000 filename html`

This returns the filename of the entry point for the page display, that is `index.html` for the specified markup.

Setting attributes of a static page

To set attributes for a static page, use the following command:

`$Content set` *`oid attribute value markup`*

Valid attributes are as follows:

- `filename`. Use this attribute for getting the file name of the static page layout file in the compressed file.
- `displayoption`. Use this attribute for the markup, for example HTML.

Example 1: `$Content set 6_CGAH47L00G2N802TJFV58Q3000 filename anotherindex.html html`

This sets the entry point for the page display to `anotherindex.html` for the specified markup.

Example 2: `$Content set 6_CGAH47L00G2N802TJFV58Q3000 displayoption iframe html`

This sets the display option to `iframe` for the specified markup. Valid display option settings are `inline`, `iframe`, and `ajax`.

Deleting a static page

The command for deleting a static page is the same as for deleting a standard portal page. You can use the command: `$Content delete` *`oid`*

To delete a static page, use the following command:

Example: `$Content delete 6_CGAH47L00G2N802TJFV58Q3000`

Related concepts:

"Working with the Portal Scripting Interface" on page 1120
Learn more about the different modes that you can use with the Portal Scripting Interface.

Using WebDAV to manage pages and static content

WebDAV for IBM WebSphere Portal Express provides a simple and easy way to administer portal resources. Both administrators and users can use it.

About this task

WebDAV for WebSphere Portal Express allows users to administer portal pages and content of static pages of a portal by using standard operating system tools. This way client side administrators and users can browse, read, and write these resources by using file explorers or editors.

Portal pages are represented as folders. They can contain subfolders that represent child pages. Static pages are a special case: the content of a static page is located in a separate folder named `staticcontent` under the main folder of that page.

Users can work in the folders as usual, for example by performing drag-and-drop operations. The folders also hold property files that contain metadata for portal resources, such as the title and description. Users can edit the property files to update portal resources. When the user saves the updated file, the updates are transferred and applied to the portal directly.

Note: All the files shown in WebDAV are virtual files created from the page data in the portal database

You can use WebDAV to perform the following tasks:

- Browse through the page hierarchy. Each portal page is represented as a folder. The name of the folder is the unique name or the object ID of the page. Children pages in the topology are represented as subfolders.
- Change globalization information of pages. To do this, users edit and save properties files that contain the globalization information.
- Change metadata of pages. To do this, users edit and save properties files that contain the metadata information.
- Delete pages.
- For static pages only, you can browse, read, create, update, save, copy, move, and delete static content, such as HTML or image files. Users can access the content of static pages via the subfolder `staticcontent` of the page.

You cannot use WebDAV to perform any of the following tasks:

- Create new pages.
- Update portal content.
- Modify the unique name or `objectID` of pages.
- Move or copy pages.

WebSphere Portal Express contains the WebDAV service and enablement layer. Before using WebDAV for WebSphere Portal Express, users must set up their WebDAV client.

After setting up the WebDAV client, they can connect to WebDAV for WebSphere Portal Express and work with portal pages and content. To connect to WebDAV for WebSphere Portal Express, they enter the WebDAV entry URL.

Security: The WebDAV entry point requires user authentication via HTTP basic authentication. SSL access is not supported at this time. To use WebDAV, users log in to the portal with their portal user ID. Users can then access and work with portal pages according to their access permissions as set by Portal Access Control.

The WebDAV entry URL is as follows:

- For default portal installations:

  `http://`*`server_name`*`:`*`port_number`*`/wps/mycontenthandler/dav/contentmodel/wps.content.root/`
- For virtual portals:
  - If a host name was specified when the virtual portal was created, the WebDAV URL looks like this:

    `http://`*`virtual_portal_host_name`*`:`*`port_number`*`/wps/mycontenthandler/dav/contentmodel/wps.content.root/`
  - If the virtual portal was created with a URL context only and no host name was specified, the WebDAV URL looks like this:

`http://`*`server_name`*`:`*`port_number`*`/wps/mycontenthandler/`*`URL_context_of_the_virtual_portal`*`/!ut/p/dav/contentmodel/wps.content.root/`

For details about virtual portals and how to create them by host name or URL context refer to the topic about Multiple virtual portals and its subtopics.

"Connecting to WebDAV to work with portal pages and static content"
To connect to WebDav to work with portal pages and static content, you enter the WebDAV entry URL.

Connecting to WebDAV to work with portal pages and static content:

To connect to WebDav to work with portal pages and static content, you enter the WebDAV entry URL.

About this task

The WebDAV entry URL is as follows:

- For default portal installations:

  http://*server*:*port*/wps/mycontenthandler/dav/contentmodel/*page_unique_name*/

  or

  http://*server*:*port*/wps/mycontenthandler/dav/contentmodel/*page_object_id*/
- For virtual portals:
  - If a host name was specified when the virtual portal was created, the WebDAV URL is as follows:

    http://*virtual_portal_host_name*:*port*/wps/mycontenthandler/dav/contentmodel/*page_unique_name*/

    or

    http://*virtual_portal_host_name*:*port*/wps/mycontenthandler/dav/contentmodel/*page_object_id*/
  - If the virtual portal was created with a URL context only and no host name was specified, the WebDAV URL is as follows:

    http://*server*:*port*/wps/mycontenthandler/*URL_context_of_the_virtual_portal*/!ut/p/dav/contentmodel/*page_unique_name*/

    or

    http://*server*:*port*/wps/mycontenthandler/*URL_context_of_the_virtual_portal*/!ut/p/dav/contentmodel/*page_object_id*/

Where:

server Host name of the portal server.

port Port number of WebSphere Portal Express.

URL_context_of_the_virtual_portal
URL context of the target virtual portal, if the virtual portal URL context is configured to be encoded into the URL.

virtual_portal_host_name
Host name of the target virtual portal, if the virtual portal host name is configured to be encoded into the URL.

page_unique_name
Unique name for the portal page.

page_object_id
`objectID` for the portal page as it appears in the Manage Pages portlet.

Examples of entry URLs to all portal pages are as follows:

```
http://www.my_company.com:10039/wps/mycontenthandler/dav/contentmodel/wps.content.root/
```

For virtual portals examples of entry URLs to all portal pages are as follows:

- For a virtual portal created with the host name `vp.mycompany.com`:
  `http://vp.mycompany.com:10039/wps/mycontenthandler/dav/contentmodel/wps.content.root/`
- For a virtual portal created with the URL context `vp1` and without a host name:
  `http://localhost:10039/wps/mycontenthandler/vp1/!ut/p/dav/contentmodel/wps.content.root/`

For details about virtual portals and how to create them by host name or URL context refer to the topic about Multiple virtual portals and its subtopics.

Related concepts:

"Virtual portals" on page 1361
View information on how you can scope your WebSphere Portal Express to have multiple virtual portals.

Dynamic content

Dynamic content is generated when a page is rendered and is based on a set of predefined criteria such as the current user, the metadata of the current page. It allows you to deliver content specifically customized for the current user, or to deliver content customized for a campaign or product in your organization.

Dynamic web content

Web content can be generated at render-time using dynamic element types:

Menus
: A menu displays metadata and content from content items that match the search criteria of the menu element. The search criteria of a menu can include matching site areas, authoring templates, categories, and keywords.

Personalization element
: A personalization element stores a reference to a personalization rule or content spot generated by Portal Personalization. To use a personalization element you must create a personalization component.

User name element
: This element will display different content for authenticated or anonymous users.

Targeted content and Portal Personalization

Personalization can recognize a specific user based on a profile or can determine characteristics of a user based on previous purchases, products or pages viewed, and so forth. Personalization then selects content that is appropriate for that profile. If a person has a high salary range, Personalization can be configured to retrieve information about a commercial website premium product. If an individual belongs to a particular geographic region, content specific to that region may be targeted to the individual. The page is assembled with the proper personalized information, and the user sees her personalized page.

Targeted content provides you with a way to deliver multiple pieces of content to different audiences. Targeted content matches the best content with the most appropriate group by using segments. Segments help you split your audience into meaningful groups with different interests or characteristics.

A targeted spot displays different content to different segments. You can create a target spot by defining content that is targeted to specific segments:

- Add content items to your content spot in a web content viewer.

- Add segments to each content item to display your content to the correct audience. Segments help you define your target audience. For example, you can define the audience by users, device class, or other attributes.

Content as a Service pages

Starting with version 8.5 CF05, IBM WebSphere Portal Express introduces the concept of Content as a Service pages. The Content as a Service pages can be used to render content that is managed by your IBM Web Content Manager in different data formats.

Individual representations of your web content can be generated by using the corresponding IBM Web Content Manager presentation components. Instead of generating HTML, the IBM Web Content Manager presentation components can be used to generate representations of your web content in formats of your choice for example, JSON, or XML. Content as a Service pages allow the content that is centrally authored and maintained on your website, to be accessed by the other data clients in raw data formats. Those data clients can generate their own client-specific visualization of the data. This approach of generating data is especially useful in mobile application that might want to present the content consistent with the other parts of the application.

CF05 "Technical concepts"
Before you use the Content as a Service pages in IBM WebSphere Portal Express, familiarize yourself with its building blocks.

CF05 "Select data format based on MIME type" on page 1882
With Content as a Service pages, you can specify different representation of your web content for different MIME types. This way when you request Content as a Service pages, you can specify the preferred representation of your web content. There are different options to manage the presentation components that produce the output for the MIME types you like to support.

CF05 "Setting up Content as a Service" on page 1884
To be able to work with Content as a Service pages in WebSphere Portal Express, you must enable it by using an IBM WebSphere Portal Express configuration task.

CF05 "Removing Content as a Service" on page 1884
To remove Content as a Service feature in WebSphere Portal Express, you must disable it by using an IBM WebSphere Portal Express configuration task.

CF05 "Access Content as a Service" on page 1885
To access your Content as a Service pages, you can write links to your content that specifies the CaaS page as target.

Technical concepts

Before you use the Content as a Service pages in IBM WebSphere Portal Express, familiarize yourself with its building blocks.

CaaS page

The CaaS page is the default Content as a Service page with the unique name `ibm.portal.caas.page`.

The CaaS page can be created by running the `install-caas-vp` task. The page uses the CaaS theme and a single Web Content Viewer portlet instance to generate the individual data representations for your content. The page can be addressed through its unique name or through the friendly name *caas*.

Important: Portal pages and web content libraries are virtual portal scoped resources. To use Content as a Service with content from a virtual portal, you must install the CaaS page to the virtual portal first.

CaaS theme
This theme with the unique name `wp.theme.caas` is referenced by the CaaS page. The theme ensures that no portal page aggregation-related markup gets added to the data representation strings generated by your Content as a Service presentation component.

Select data format based on MIME type

With Content as a Service pages, you can specify different representation of your web content for different MIME types. This way when you request Content as a Service pages, you can specify the preferred representation of your web content. There are different options to manage the presentation components that produce the output for the MIME types you like to support.

How to use well-defined element names in your content

The portal supports selecting different presentation components for the Content as a Service pages based on the requested MIME type. Depending on the MIME type parameter, different presentation components can be specified for generating the data format. For more information about the MIME type parameter, go to the section *Writing links to web content*.

You can use this MIME type-specific presentation component selection by adding component reference elements with well-defined names such as `ibm.design.json`, `ibm.design.xml`, and `ibm.design.html` to your content items. You can define the value for the component reference elements in your Authoring template that you use for creating your content items.

For example, specifying the parameter **`mime-type=application/json`** specifies the content element with the name `ibm.design.json` as the presentation component for representing your web content.

Important: If you choose this approach, there are implications that are related to how you manage your content. The presentation component elements are part of your content. Adding, removing, or updating one of these well-defined elements requires you to update all related content items individually. Restoring previous versions of content items can require you to update the presentation component again.

How to use presentation templates with the `pagedesign` query parameter

As an alternative to managing the well-defined elements in your content, you can also use alternate presentation templates based on the requested MIME type. If the portal does not find a content element for the requested MIME type, it renders the content using the presentation template specified by the `pagedesign` parameter from the web content link. (See “Writing links to web content” on page 1836). If the request does not include that parameter, the default presentation template of your content is used to produce the output.

The `pagedesign` parameter is another way of managing your presentation components for different MIME types to render your content. Furthermore, it also allows you to produce output for MIME types other than the ones supported by the well-defined content elements. For example, you can create a presentation

template that produces a "text/csv" (comma-separated values) representation of your content. You would then request your content with that MIME type with a URL like this:

```
http://hostname/context_root/virtual_portal_context/caas?current=true
&urile=wcm%3Apath%3Alibrary/site_area_path/content&mime-type=text/csv
&pagedesign=library/folder/presentation_template
```

Important: If you choose this approach, you must explicitly add the `pagedesign` parameter to the URLs when you request a representation of your content with a specific MIME type. If the output generated for a specific MIME type also includes web content links, the `pagedesign` parameter is not automatically added by the portal.

How to use presentation templates with a custom context processor

Instead of using the `pagedesign` query parameter, you can also implement a custom context processor that sets the presentation template dynamically based on the requested MIME type. This solution provides you with the full MIME type flexibility as the second option and does not require the `pagedesign` parameter in web content links. Your context processor can change the presentation template for a request based on the requested MIME type from the `mime-type`parameter. For more information on custom context processors, go to the topic "Creating a context processor class" on page 3230.

For example, you might use a naming convention to select the proper presentation template for a request: `authoring_template_name.mime_type`. Given this simple naming convention, requesting the JSON representation of the content item `Web Content/Articles/Sample Article` that is based on the authoring template `Web Content/Article` would mean that your context processor sets the alternative presentation for the request to `Web Content/Article.json`.

Important: You need to configure the instance of the Web Content Viewer portlet that is located on the CaaS page with the unique name `ibm.portal.caas.page` to use your context processor. To configure the context processor for the instance of the portlet, use the **Edit Shared Settings** mode of the portlet from the CaaS page. More information on configuring the Web Content Viewer portlet, go to the topic Advanced options. If you use Content as a Service pages with virtual portals, make sure to also configure the portlet instances on the CaaS pages in each virtual portal.

Limitations when using Content as a Service

- With Content as a Service, you always request web content items from a web content library. You cannot use this service to render components from a web content library.
- With Content as a Service, you cannot produce binary web content. For example, you cannot respond with the MIME type `image/jpg`. If you want to retrieve binary web content, such as images, from a web content library, include the URL to the web content in the Content as a Service response. The client can then use the URL to make a second request to load the binary web content.

Related concepts:

CF05 "Writing links to web content" on page 1836
Links to content items can be written as URLs.

Setting up Content as a Service

To be able to work with Content as a Service pages in WebSphere Portal Express, you must enable it by using an IBM WebSphere Portal Express configuration task.

Before you begin

For Content as a Service pages to work, you must have the new functions for WebSphere Portal Express and IBM Web Content Manager Version 8.5 CF05 enabled. For more information, see the *readme file for WebSphere Portal and Web Content Manager V8.5 CF05*.

About this task

The setup for Content as a Service pages comprises both resources that are shared across virtual portals and virtual portal scoped resources.

To set up Content as a Service pages, proceed with the following steps:

Procedure

1. Change the directory to the `wp_profile_root/ConfigEngine` directory. If you are using IBM z/OS, open a UNIX System Services command prompt to change directories.
2. Initiate the setup.
   - To initiate the setup of shared resources, run the configuration task `install-caas` as follows:`./ConfigEngine.sh install-caas -DPortalAdminID=user_id -DPortalAdminPwd=password -DWasUserid=user_id -DWasPassword=password`
   - To initiate the setup of virtual portal scoped resources in the default virtual portal, run the configuration task `install-caas-vp` as follows:`./ConfigEngine.sh install-caas-vp -DPortalAdminID=user_id -DPortalAdminPwd=password -DWasUserid=user_id -DWasPassword=password`
   - To initiate the setup of virtual portal scoped resources in a virtual portal, run the configuration task `install-caas-vp` as follows by adding the **`-DVirtualPortalContext`** or **`-DVirtualPortalHost`** as follows:`./ConfigEngine.sh install-caas-vp -DPortalAdminId=user_id -DPortalAdminPwd=password -DWasUserid=user_id -DWasPassword=password -DVirtualPortalContext=virtual_portal_context`
3. Restart your portal server.

Removing Content as a Service

To remove Content as a Service feature in WebSphere Portal Express, you must disable it by using an IBM WebSphere Portal Express configuration task.

Procedure

1. Change the directory to the `wp_profile_root/ConfigEngine` directory. If you are using IBM z/OS, open a UNIX System Services command prompt to change directories.
2. Initiate removal.
   - To initiate the removal of virtual portal scoped resources in the default virtual portal, run the configuration task `uninstall-caas-vp` as follows:`./ConfigEngine.sh uninstall-caas-vp -DPortalAdminID=user_id -DPortalAdminPwd=password DWasUserid=user_id -DWasPassword=password`

- To initiate the removal of virtual portal scoped resources in a virtual portal, run the configuration task `uninstall-caas-vp` by adding the **-DVirtualPortalContext** or **-DVirtualPortalHost**as follows:`./ConfigEngine.sh` *`uninstall-caas-vp`* `-DPortalAdminID=`*`user_id`* `-DPortalAdminPwd=`*`password`* `DWasUserid=`*`user_id`* `-DWasPassword=`*`password`* `-DVirtualPortalContext=`*`virtual_portal_context`*
- To initiate the removal of shared resources, run the configuration task `install-caas` as follows:`./ConfigEngine.sh` *`uninstall-caas`* `-DPortalAdminID=`*`user_id`* `_DPortalAdminPwd=`*`password`* `DWasUserid=`*`user_id`* `-DWasPassword=`*`passowrd`*

  Note: Uninstalling the shared resources while the virtual portal scoped installation is still present for other virtual portals leads Content as a Service feature to become dysfunctional for all virtual portals.

3. Restart your portal server.

Access Content as a Service

To access your Content as a Service pages, you can write links to your content that specifies the CaaS page as target.

When you construct the URLs for your links to web content, you can use the **`urile`** URL parameter to identify the content item. To access the content item by using the CaaS page, use one of the following methods:

- Include the friendly name of the CaaS page in the URL and add the URL parameter `current=true` to the URL.
- Specify the unique name of the CaaS page (`ibm.portal.caas.page`) in the URL by using the "*page*" URL parameter.

Generic URLs

Addressing the CaaS page from a virtual portal by using the page friendly name:

`http://`*`hostname`*`/`*`context_root`*`/`*`virtual_portal_context`*`/caas?current=true&urile=wcm:path:`*`libra`*

Addressing the CaaS page from a virtual portal by using the page parameter:

`http://`*`hostname`*`/`*`context_root_poc`*`/`*`virtual_portal_context`*`?page=ibm.portal.caas.page&urile=wc`

Example URLs

A URL to render the content item *Item1* that is stored in the site area *SiteArea1* of *Library1* by applying a JSON presentation template that is named *Presentation1* from the library that is named *Library2* looks like the following examples:

Addressing the CaaS page from a virtual portal by using the page friendly name:

`http://example.com/wps/myportal/vp1/caas?current=true&urile=wcm:path:Library1/SiteArea1/It`

Addressing the CaaS page from a virtual portal by using the page parameter:

`http://example.com/mypoc/vp1?page=ibm.portal.caas.page&urile=wcm:path:Library1/SiteArea1/I`

To render the content item that is found at location `Web Content/News/News1`, you can use a URL like the following example:

`http://example.com/wps/mypoc?urile=wcm:path:/Web+Content/News/News1&page=ibm.portal.caas.page`

If a JSON presentation component is associated to that content, you can request the JSON representation of this content item by using URL like the following example:

`http://example.com/wps/mypoc?urile=wcm:path:/Web+Content/News/News1&page=ibm.portal.caas.page&mime`

Related concepts:

CF05 "Writing links to web content" on page 1836
Links to content items can be written as URLs.

URLs

Portal URLs do not look like plain HTTP server URLs over a simple file system. Portal URLs have a complex structure and include a large compressed and encoded XML Navigation State document. The stream of random characters in a Portal URL is the Navigation State document. Full Portal function depends on correctly maintaining this Navigation State document during all the operations a user might do in Portal. Forcing Portal URLs to look like plain HTTP server URLs over a simple file system structure cripples the function of WebSphere Portal Express.

Some of the use cases that must be supported by Portal URLs include the following:

Backward and Forward button support
: The user can use the browser's back and forward buttons to switch between recent views.

Bookmark-ability
: The current view can be saved into a client-side bookmark and can be accessed at any time. The user might need to log on first before they can access the bookmark. The bookmark must remain valid across portal versions.

Crawl-ability
: Web-Crawlers such as Googlebot can crawl portal pages and index them.

Cacheability
: Views of pages must be cacheable.

Non-Functional requirements
: Non-functional requirements include RFC 1738 URL size limitations, URL generation performance, and markup size management when markup contains many Portal URLs.

A Portal URL can be self-contained, or can be a delta URL relative to a base. A Portal URL might contain human-readable tokens for URL Mappings, Friendly URLs, Vanity URLs, or Virtual Portal context, in addition to the encoded Navigational State document.

For users to find specific content in your website, you can provide friendly URLs or vanity URLs. These URLs are short URLs that people can easily remember. They are shorter than full WebSphere Portal Express URLs. These short URLs are sometimes also called marketing URLs. You can publish them for marketing campaigns through different channels, such as email or print. This way, you can use them to direct customers to a specific portal page or content item. Interested site visitors who want to view your campaign can then remember or copy the short URL and type it into the browser address field.

Friendly URLs
: Friendly URLs have human-readable strings in the URL that describe the path to a Portal page. These human-readable strings correspond to the Friendly URL Names that are associated with the pages or labels. In addition, there might also be Friendly Content Path tokens in the URL. The

Friendly Content Path tokens are human-readable strings that describe the site area path to Web Content Management library associated with the page.

Note: A friendly URL might also include an encoded Navigational State document. If it does not, it is a Stateless Friendly URL. There is a programming API specifically for working with Friendly URLs.

Vanity URLs
Vanity URLs are similar to Stateless Friendly URLs, in that they are human-readable and do not have an encoded Navigational State document. However, Vanity URLs are not tied to the Friendly URL Names associated with the Portal pages. Instead, Vanity URLs are intended to be aliases that are simple, easily remembered, and easily entered by hand if necessary. Vanity URLs are similar to Mapped URLs that were introduced in prior releases of WebSphere Portal. They are intended only as an initial entry point, and are not persistent in the browser address bar after interaction with the Portal site begins. There is a programming API specifically for working with Vanity URLs.

Deciding between vanity URLs and friendly URLs

Depending on your requirements, you can use vanity URLs, or friendly URLs:

- If you want to have a short URL as an entry point to a specific portal page or content item, use a vanity URL.
- If you want to have a friendly URL that your site visitors see when the portal shows the page, use a friendly name.
- If you want to be able to publish the page through the Web Content Manager workflow, use a vanity URL. For example, this URL can be useful for a marketing campaign.
- If you want to address a specific portal page through URL generation tags or APIs, use unique name IDs. For more information, see *URL generation in WebSphere Portal*.

You can create both vanity URLs and friendly URLs for the same portal page.

URL mappings

URL mappings were deprecated starting with WebSphere Portal Express Version 8.5. Instead, you can now use friendly URLs or Vanity URLs as an alternative to URL mappings.

Related concepts:

"URL generation in WebSphere Portal" on page 2835
Generating Portal URLs correctly is one of the most important tasks in programming a WebSphere Portal Express based application. There are several programming tools and techniques available for generating WebSphere Portal Express URLs in custom code. The following section introduces the programming tools available and discusses when it is most appropriate to use each of the tools.

Authoring tools

There are multiple ways to create and manage content including using the site toolbar, inline editing, and the authoring portlet.

"Site toolbar" on page 1888
The site toolbar is useful for managing and editing content day-to-day. It

provides convenient access to many capabilities directly from the website. Content authors can create pages, add content and application to pages, and manage page parameters.

"Edit mode" on page 1889
Edit mode is available from the site toolbar.

"Web content inline editing strategies" on page 1890
An inline editing system is used to deliver editable websites such as an intranet or a wiki. It combines the features of both an authoring system and a delivery system.

"Authoring portlet" on page 1897
The authoring portlet is another interface that can be used by content authors, site administrators, and site developers. From the authoring portlet, content authors use forms, called authoring templates, to add content to the website. You can use to customize the Web Content Manager user interface to simplify the content authoring process for your content creators.

Site toolbar

The site toolbar is useful for managing and editing content day-to-day. It provides convenient access to many capabilities directly from the website. Content authors can create pages, add content and application to pages, and manage page parameters.

The site toolbar is available for immediate use after you deploy the portal. The availability of the site toolbar is controlled by the theme. So you can disable it on your production server.

Figure 21. Screen capture of the site toolbar with annotations.

1 Menu options

There are three menu options: Site, Application, and Administration.

Use the **Site** menu to switch between the sites that are hosted on a single portal server.

Use the **Application** menu to access application such as Content, Collaboration, Messaging, Personalization, and Unified Task List.

Use the **Administration** menu to access the administration portlets.

2 Toolbar

The toolbar provides access to authoring tools such as **Projects**, **Create**, and **Page**.

From the Projects tab, the content authors can create new projects to manage multiple changes. The tab displays the content that has been created or changed in the website for that project. Each content item can be managed individually. The content author can also create project templates.

From the Create tab, content authors can create pages, add applications to a page, or add content to a page. The content shelf contains Web Content Viewer portlets that are preconfigured with content associations. When the content author drags content to the page, a copy of the associated content is added to the page. Then the content author can customize the content.

From the Page tab, content authors can view and modify page settings, such as layout, vanity URLs, and more.

3 Project indicator
The project indicator provides access to the project menu and displays the currently active project. If there is no active project, then the indicator shows "Published Site" and you are viewing the currently published website.

4 Edit Mode
Use the **Edit Mode** control to turn editing capability on and off.

5 Information Mode
Turns inline user assistance on and off. Inline assistance includes descriptions and examples that appear in the user interface and hover help.

Currently this mode is only implemented in the site toolbar.

6 Menu
Provides access to additional page and preview actions.

Page actions include: View Page Properties, Show Hidden Pages, Show Hidden Content, Move Page, and Delete Page.

Preview actions include: Preview as a specific user and preview as an unauthenticated user.

Site toolbar customization

The site toolbar is populated with the portlets that are immediately available after you deploy the portal. It includes a number of configured content viewers too. To customize your toolbar, you can:

- Add page templates to the Create Page tab
- Add new categories and applications to the Create Content tab
- Add configured content to the Create Content tab

Related concepts:

"Customizing the Content palette" on page 1904
The Content palette, which is accessed from the site toolbar, contains the content items a content author can add to a page.

"Customizing the Applications palette" on page 1909
WebSphere Portal Express includes ready to use portlets that you can browse through in the site toolbar. To ease browsing through this set of portlets, WebSphere Portal Express supports assigning portlets to one or more categories. As the site administrator, you can create, delete, and modify categories and assign portlets to them.

Edit mode

Edit mode is available from the site toolbar.

Edit mode and administration pages

Administration pages are not intended to be edited, those pages are excluded from the edit mode features provided by the site toolbar. The administration pages are not managed in the Web Content Manager. The edit mode features are suppressed by the page parameter **theme.disable.edit.mode** on the Administration page label (unique name wps.Administration). Setting the parameter on the top-level administration page also causes the child pages to be affected.

Disable edit mode

You can set the **theme.disable.edit.mode** parameter on any page where you want to disable edit mode in the site toolbar. Edit the properties of the page, and add the **theme.disable.edit.mode** parameter with a value of `true`.

Web content inline editing strategies

An inline editing system is used to deliver editable websites such as an intranet or a wiki. It combines the features of both an authoring system and a delivery system.

"Inline editing"
Inline editing enables users with edit access to a content item to edit that item from within the web page itself instead of using the authoring portlet. This feature is available when you display content with a web content viewer portlet.

"Creating an authoring tools element" on page 1892
The authoring tool element is used to add authoring portlet functions to web pages. When you create an authoring tool element, you need to define the layout of the authoring tool and any required actions, and select parameters for each action layout as required.

Inline editing

Inline editing enables users with edit access to a content item to edit that item from within the web page itself instead of using the authoring portlet. This feature is available when you display content with a web content viewer portlet.

The inline editing feature requires:

- WebSphere Portal version 8.0.0.1 or later, or Web Content Manager version 8.0.0.1 or later.
- A theme that supports Dojo. The themes that are included with WebSphere Portal are enabled for Dojo.

  If you create a custom theme, in addition to supporting Dojo, the new theme must reference the `wcm_inplaceEdit` theme module. For information about creating themes that support Dojo, see *Dojo and WebSphere Portal*.

Important: This feature can be enabled or disabled by enabling or disabling the content targeting feature and application objects. For more information, see *Installing content targeting features and application objects*.

Note: When a user tabs between inplace editing enabled fields, users need to click each field in turn to access inplace editing for each field.

"Updating sample template items for inline editing after an upgrade installation" on page 1891
IBM Web Content Manager includes sample content such as web content template pages and predefined portlets that you can add to pages to render content. If you upgrade, these sample web content template items continue to use the editing method of the earlier release. To use the inline editing method with the earlier template items, you must complete several manual steps.

"Enabling inline editing for content items" on page 1892
You can enable inline editing for content item fields in your site design.

Related information:

Creating an editable property tag

Creating an editable element tag

Creating an If Edit Mode tag

Creating an If Not Edit Mode tag

Updating sample template items for inline editing after an upgrade installation:

IBM Web Content Manager includes sample content such as web content template pages and predefined portlets that you can add to pages to render content. If you upgrade, these sample web content template items continue to use the editing method of the earlier release. To use the inline editing method with the earlier template items, you must complete several manual steps.

Procedure

1. Go to the *wp_profile_root*/ConfigEngine directory.
2. To redeploy the web content viewer portlets that are used with the template items, run the **action-deploy-templating-portlets** task.
   - Linux : `./ConfigEngine.sh action-deploy-templating-portlets -DPortalAdminPwd=password -DWasPassword=password`
   - Windows: `ConfigEngine.bat action-deploy-templating-portlets -DPortalAdminPwd=password -DWasPassword=password`
   - IBM i: `ConfigEngine.sh action-deploy-templating-portlets -DPortalAdminPwd=password -DWasPassword=password`
3. Log in to the portal as an administrator.
4. Go to the web content authoring portlet. For example, click **Applications** > **Content** > **Web Content Management**.
5. Edit the preferences for the authoring portlet, and ensure that the Portal Site library is listed with the libraries that are displayed in the portlet.
6. In the authoring portlet, click **Portal Site** > **Content** > **Content Root** > **Hidden Pages** > **Page Templates** > **Articles**.
7. Delete the following content items: **Article**, **No Items Found**, and **List of Articles**.
8. Log out of the portal.
9. Go to the *wp_profile_root*/ConfigEngine directory.
10. Run the **action-init-content-templating-pages** task.
    - Linux : `./ConfigEngine.sh action-init-content-templating-pages -DPortalAdminPwd=password -DWasPassword=password`
    - Windows: `ConfigEngine.bat action-init-content-templating-pages -DPortalAdminPwd=password -DWasPassword=password`
    - IBM i: `ConfigEngine.sh action-init-content-templating-pages -DPortalAdminPwd=password -DWasPassword=password`
11. Run the **action-internalize-content-mappings** task.
    - Linux : `./ConfigEngine.sh action-internalize-content-mappings -DPortalAdminPwd=password -DWasPassword=password`
    - Windows: `ConfigEngine.bat action-internalize-content-mappings -DPortalAdminPwd=password -DWasPassword=password`
    - IBM i: `ConfigEngine.sh action-internalize-content-mappings -DPortalAdminPwd=password -DWasPassword=password`

Results

After you complete these steps, the sample content items that are provided by default in the site toolbar are enabled for the updated inline editing feature.

- The following content items in the **Web Content** category of the **Content** tab are updated: **Article**, **Image**, **List of Articles**, **Rich Text**. These content items are modified to reference new versions of the items that are provided in the Template Page Content 3.0 library.
- The Articles page template is modified to reference a new version of the template that is provided in the Template Page Content 3.0 library.

Enabling inline editing for content items:

You can enable inline editing for content item fields in your site design.

About this task

Inline editing is enabled separately for each of the following content item fields:
- Name
- Title
- Description
- Author and owner
- Published date
- Expiry Date
- General Date 1
- General Date 2
- Each element that is stored in the content item.

You can enable inline editing in the design in two ways:
- Use editable element or editable property tags in presentation templates or component designs.
- Enable inline editing for new item creation with an authoring tool component.

Note to developers: When you create a custom theme, you must include a reference to the *wcm_inplaceEdit* module to enable inplace editing in the custom theme.

Creating an authoring tools element

The authoring tool element is used to add authoring portlet functions to web pages. When you create an authoring tool element, you need to define the layout of the authoring tool and any required actions, and select parameters for each action layout as required.

"The Authoring tools element" on page 1893
The authoring tool element is used to add authoring portlet functions to web pages.

"Inline item creation with authoring tool components" on page 1894
Inline item creation can be enabled for authoring tool components so that users can quickly create a new content item without having to access the authoring interface.

"How to use authoring tools components in a Web Content Viewer" on page 1895
When rendering authoring tool components in the Web Content Viewer, you must account for some changes in the way placeholder tags are specified and in the way users browse to pages that contain authoring tool components. Authoring tasks that are accessed through a Web Content Viewer, such as inline

editing, require the use of an instance of the IBM Web Content Manager authoring portlet that is reserved specifically for such tasks.

"How to reference an authoring tool" on page 1897
An authoring tool component can be referenced within presentation templates, menu element designs, and navigator element designs. When added to menus and navigators, the edit and delete functions are applied to each item displayed in a menu or navigator.

The Authoring tools element:

The authoring tool element is used to add authoring portlet functions to web pages.

You can add the following authoring portlet functions to a web page:

- Create a content item.
- Perform inline editing of a content item that is displayed in a web page.
- Delete the content item that is displayed in a web page.
- Approve or reject the current content that is being previewed. These options are visible to approvers only when they preview a draft item, or by opening the URL sent by an email action during a workflow.

Authoring tools can be referenced within presentation templates, menu element designs, and navigator element designs. When added to menus and navigators, the edit, delete, and approve functions are applied to each item displayed in a menu or navigator.

Creating an authoring tool element

You can use an authoring tool element only by creating an authoring tool component. You cannot add an authoring tool element to authoring templates, site areas, or content items.

How to use an authoring tool

When content is previewed, users with access to an authoring tool are able to run various authoring portlet functions.

Note: When content items that use a presentation template that includes an authoring tool are previewed, some functions are active and ready to use while other functions might not work as normal.

The authoring tool is also visible on the published site.

How to use an authoring tool on multiple servers

When an authoring tool is used on more than one server, you use two-way syndication to keep each server synchronized. This can lead to the occasional "save" conflict where an item updated on one server is overwritten with changes to the same item on another server when syndication occurs.

User access to an authoring tool

The authoring tools available to users on a web page are determined by:

- Whether a user has access to the authoring tool component.

- Which tools are enabled in the authoring tool element.
- The user's level of access to the content item displayed in a web page.
- Whether a user has access to the authoring portlet. Users are assigned at least contributor access to each web content library in a site to ensure that they have access to the authoring portlet.
- You grant "editor" access or higher to users who need to edit the authoring tool.
- You grant "user" access to users who also have access to the authoring server and who are using the authoring tool.
- In most cases, users who have access to the published site would not be granted access to an authoring tool as the tool is used as an authoring tool on an authoring server, not a published site.

Inline item creation with authoring tool components:

Inline item creation can be enabled for authoring tool components so that users can quickly create a new content item without having to access the authoring interface.

How to enable inline item creation with an authoring tool component

To enabled inline item creation with an authoring tool component, you select **Enable inline item creation** under the **New action properties** section of the authoring tools component form.

When enabled, when a user clicks the new action tool on the rendered web page, a small dialog is displayed where a user enters the name of the new item. No other fields are displayed.

For inline item creation with an authoring tool component to work:
- The authoring template must use a multiple stage workflow with the initial stage being a draft, or the authoring template has workflows disabled.
- The location for the new item must have been predefined in either the authoring template, or the authoring tool component.
- The portal page the authoring tool is displayed on must be in edit mode.

If any of these requirements are not met, the normal inline item creation dialog is displayed instead.

How to use inline item creation with an authoring tool component

Inline item creation with an authoring tool component works best when:
- The item created using the authoring tool has a valid page mapping so that the new item is automatically opened in the correct page.
- The authoring template used to create the content item contains some default content and metadata. For example, the **Title** field might be pre-filled with the sentence "Add your title here".
- The presentation template used by the new content item uses editable property and editable element tags so that the user can immediately update the new content item.
- If using a workflow, you should select **Allow templated items to be saved in the first draft stage even if the item fails field validation** when creating the workflow. This will allow users to create draft items that do not meet all field validation requirements.

- The new action properties section can use the tags `[Placeholder tag="titlelink"]`, `[Placeholder tag="namelink"]` or `<a href="javascript:;" onclick="[Placeholder tag="href"]">Create</a>` to create the link that opens the inline item creation dialog.

This ensures that the user is taken to an inline edit ready content item when they create a new item by using the inline authoring tool.

How to use authoring tools components in a Web Content Viewer:

When rendering authoring tool components in the Web Content Viewer, you must account for some changes in the way placeholder tags are specified and in the way users browse to pages that contain authoring tool components. Authoring tasks that are accessed through a Web Content Viewer, such as inline editing, require the use of an instance of the IBM Web Content Manager authoring portlet that is reserved specifically for such tasks.

"How to configure authoring tools components"
Authoring tools components that are rendered in a web content viewer allow you to create, read, edit, delete, approve, or reject content items directly in the web content viewer, instead of requiring you to browse to the IBM Web Content Manager authoring portlet to run the same action. The web content viewer either opens a window from the current page or redirects the user to another portal page that contains the authoring portlet.

How to configure authoring tools components:

Authoring tools components that are rendered in a web content viewer allow you to create, read, edit, delete, approve, or reject content items directly in the web content viewer, instead of requiring you to browse to the IBM Web Content Manager authoring portlet to run the same action. The web content viewer either opens a window from the current page or redirects the user to another portal page that contains the authoring portlet.

You can specify which behavior to use in the authoring tools element design. Typically placeholder tags are used to display authoring tools elements. The value of the `format` attribute of the placeholder tag determines what URL is created to run an authoring task:

`format="tag"`
The placeholder is rendered as a URL that opens a window that contains the authoring portlet.

`format="url"`
The placeholder is rendered as a URL that redirects the user to another portal page that is used by the web content viewer for inline editing.

Note: Authoring tasks that are run in the web content viewer are accomplished through a special instance of the authoring portlet that is reserved specifically for these tasks and is installed on a page that is hidden from the page navigation available to typical users. You can customize the authoring experience for these tasks by configuring the reserved authoring portlet and the page that is used to display it.

Authoring tools components that open in a window

When you use a window to display an authoring task, the window opens on the portal page and can be moved within the boundaries of the browser window while

still showing the portal page. After you complete the task that is triggered by the authoring tools element, the window closes automatically, and the portal page refreshes, updating the view in the web content viewer. You can cancel the authoring task by clicking the close icon in the window's title bar. When a task is canceled, no web content information is saved, unless you explicitly save changes before manually closing the window.

The default value of the `format` attribute for a placeholder tag is `tag`, so to use a window for inline editing, it is not necessary to specify a value for the `format` attribute. Either of the following design examples creates a URL that opens a window for authoring tasks:

```
<Placeholder tag="namelink"/>
<Placeholder tag="namelink" format="tag"/>

<a href="<Placeholder tag="href"/>">
  <Placeholder tag="name"/>
</a>
<a href="<Placeholder tag="href" format="tag"/>">
  <Placeholder tag="name"/>
</a>
```

Note: It is not possible to open the window in a separate browser window by adding `target="_blank"` to the HTML anchor tag in the design.

Authoring tools components that open on another page

Instead of running tasks from authoring tools elements in a window on the current page, you can open authoring tasks by browsing to a hidden portal page that contains a web content viewer that contains the reserved authoring portlet. Clicking a link for an authoring tools element automatically redirects you to the other page, but after you complete the authoring task, you must manually browse back to the original page. If the page with the reserved authoring portlet was opened in a new browser window or tab, you must close the window or tab and manually refresh the original page to see any changes.

To redirect users to another page for authoring tasks, specify a value of `url` for the `format` attribute in the placeholder tag in the authoring tools element design. Either of the following design examples creates a URL that redirects users to another portal page for authoring tasks:

```
<Placeholder tag="namelink" format="url"/>

<a href="<Placeholder tag="href" format="url"/>">
  <Placeholder tag="name"/>
</a>
<a href="<Placeholder tag="href" format="url"/>" target="_blank">
  <Placeholder tag="name"/>
</a>
```

Note: You can open the portal page in a separate browser window by adding `target="_blank"` to the HTML anchor tag in the design.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

How to reference an authoring tool:

An authoring tool component can be referenced within presentation templates, menu element designs, and navigator element designs. When added to menus and navigators, the edit and delete functions are applied to each item displayed in a menu or navigator.

Referencing an authoring tool component in a presentation template

An authoring tool element is referenced in a web content component tag:

```
[Component name="authoringtoolname" ]
```

Referencing an authoring tool element in a menu or navigator design

When you reference an authoring tool component in menu or navigator designs, you use a component tag with a parameter of `compute="always"`. For example:

```
[Component name="authoringtoolname" compute="always" ]
```

Authoring portlet

The authoring portlet is another interface that can be used by content authors, site administrators, and site developers. From the authoring portlet, content authors use forms, called authoring templates, to add content to the website. You can use to customize the Web Content Manager user interface to simplify the content authoring process for your content creators.

"Custom portal pages for authoring"
You do not have to use the default Web Content Manager page to create content. You can create a new portal page to act as the home page of your authoring system.

"Authoring system access strategies" on page 1898
The roles that you assign each library on your authoring system determines what views and features in an authoring portlet are accessible to your users.

"Custom authoring interfaces" on page 1898
You can use the Web Content Manager API and remote action parameters to create customized authoring interfaces specifically for your content creators.

Custom portal pages for authoring

You do not have to use the default Web Content Manager page to create content. You can create a new portal page to act as the home page of your authoring system.

You can create separate subpages under an authoring home page. For example:

- add an authoring portlet to one subpage
- add web content viewer portlets to other subpages to allow users to preview different parts of a website

You can also create pages specifically for different types of users. For example, you can create a separate page for your site designers and content creators. The authoring portlets that you add to each page can be configured specifically for each user type.

Creating new pages

When new pages are created that contain an authoring portlet, add the following metadata parameter to the advanced settings in the page properties:

- Parameter: **resourceaggregation.profile**
- Value: profiles/profile_wcmauthoring.json

Example authoring home page

In this example, you have two users groups; site designers and content creators. The website is split between a design library and a content library.

To create a shared authoring environment for both sets of users you would create a parent home page with separate subpages for each group, plus a third subpage that is used to preview the site:

Table 277. Example authoring subpages

Site designers page	Content creators page	Header
• Includes an authoring portlet that is configured to use both the design and content libraries. • Only site designers can access this page.	• Includes an authoring portlet that is configured to use only the content library. • Only content creators can access this page.	• Includes a web content viewer portlet that is used to preview the website. • Both site designers and content creators can access this page.

Authoring system access strategies

The roles that you assign each library on your authoring system determines what views and features in an authoring portlet are accessible to your users.

Grant each user or group access to roles and item types to match the kind of work they perform. For example:

- Assign website designers editor access to authoring templates and presentation templates as they are required to create new authoring templates.
- Assign both website designers and web content authors editor access to components if they are both required to create components.
- Content approvers are only assigned contributor access to content as they are not required to create new content items, but need approver access to content items during a workflow.

Custom authoring interfaces

You can use the Web Content Manager API and remote action parameters to create customized authoring interfaces specifically for your content creators.

You might not want to use an authoring portlet as the user interface for all your users. In some cases, it might be better to create a custom authoring interface by using the Web Content Manager API and remote action parameters. For example, you can create a simple content authoring interface for a specific content authoring team.

Custom launch pages

You can configure an authoring portlet to use a launch page of your own design instead of the default user interface. A custom launch page can either be a JSP or HTML file. You use remote actions to call different views and functions from the authoring portlet's user interface. You can also use the web content API to add other functions to your launch page. After you created a custom launch page, you then configure your authoring portlet to use the custom launch page instead of the default authoring portlet user interface.

Remote actions

Remote actions are used in the query string of a URL to trigger actions from the Web Content Manager application. You can use remote actions to add standard Web Content Manager functions to a custom user interface.

Related tasks:

Creating pages using the Manage Pages portlet
A page displays content, such as portlets and other pages, in a single area. By creating pages, you can organize your information and add new navigational elements to the site.

Preparing for content authors

Before content authors can create content you must prepare your site to be ready for content creation, including preparing the site toolbar, enabling inline editing, and creating templates.

"Creating project templates"
Projects help content authors group and manage changes across the website. As a site administrator, you can create templates for your authors to use.

"Enabling inline editing for content items" on page 1892
You can enable inline editing for content item fields in your site design.

"Preparing the site toolbar" on page 1900
The site toolbar enables content authors to work with content from the website instead of using the authoring portlet. Customize the site toolbars to improve the authoring experience for content authors.

"Customizing the Web Content Authoring portlet" on page 1915
To customize each Web Content Authoring portlet for the people who use that portlet, you can edit the shared settings of each Web Content Authoring portlet on your authoring system.

"Removing the site toolbar on a production server" on page 1920
The site toolbar provides access to editing features for managed pages, including adding and editing pages and web content. Although essential for an authoring server, it is recommended that you disable the site toolbar on a delivery server. You can disable the toolbar for an entire portal or for specific virtual portals.

Creating project templates

Projects help content authors group and manage changes across the website. As a site administrator, you can create templates for your authors to use.

About this task

Project templates include the recommended workflow and publish options. Selecting a predefined project saves the content author time as they work on the website. It also enables you to suggest or enforce workflow and approvers for content changes. To create a project template, access the Web Content Authoring portlet.

Content authors can create project templates from the Projects tab. Project templates that are created from the site toolbar are named by using the following convention: "Template from *project name*"

More information about creating the project templates is available in the help system.

Procedure

1. From the **Applications** menu, click **Content**.
2. Click **Web Content Authoring**.
3. Click **New** > **Project Template**.

Results

The project template appears in the list of available templates in the Projects tab of the toolbar.

Setting a default project template: CF07 To set a default project template to be used when new projects are created, select a project template in the library explorer view, and then click **More** > **Set as Default**. To clear the default template, click **More** > **Clear Default**. This setting is scoped to the virtual portal if managed pages are enabled.

Related information:

Authoring portlet: Creating a project template (from the Web Content Authoring portlet as a site administrator)

Creating a project template (from the site toolbar as an author)

Enabling inline editing for content items

You can enable inline editing for content item fields in your site design.

About this task

Inline editing is enabled separately for each of the following content item fields:

- Name
- Title
- Description
- Author and owner
- Published date
- Expiry Date
- General Date 1
- General Date 2
- Each element that is stored in the content item.

You can enable inline editing in the design in two ways:

- Use editable element or editable property tags in presentation templates or component designs.
- Enable inline editing for new item creation with an authoring tool component.

Note to developers: When you create a custom theme, you must include a reference to the *wcm_inplaceEdit* module to enable inplace editing in the custom theme.

Preparing the site toolbar

The site toolbar enables content authors to work with content from the website instead of using the authoring portlet. Customize the site toolbars to improve the authoring experience for content authors.

In the authoring environment, content authors use the site toolbar to access and add content and applications to pages. Content authors can also create and manage pages using the site toolbar.

Site toolbar customization includes:

- Adding content
- Adding applications

The WebSphere Portal Express 8.5 site toolbar is built with portal pages and portlets. The available toolbar tabs are implemented as portal pages, which are content labels or content pages, that are grouped under the toolbar content root. All content pages and content labels that are children of the toolbar content root will show up in the toolbar navigation as toolbar tabs.

By default, the toolbar content root has the unique name `ibm.portal.toolbar.ContentRoot` and can be found under the hidden pages root unique name `ibm.portal.HiddenPages`.

The toolbar navigation supports two levels only. Content pages or labels that are appended to an existing toolbar sub tab will not show up in the toolbar navigation.

To extend the site toolbar with custom toolbar tabs, you can:

- Create a new toolbar first level tab by appending a content page or content label to the toolbar content root.
- Create a new toolbar sub tab by appending a content page to one of the existing first level tabs or to a custom first level tabs.

You can influence the order of the toolbar tabs by setting ordinals on the toolbar content nodes. Use the theme that is set on the toolbar content root. To enable intercommunication with the main page, you must set the page parameter `ibm.portal.toolbar.isToolbarPage = true` for your custom toolbar content pages and content labels.

To use the XML configuration interface for creating and managing your toolbar tabs, set this parameter as follows:

```
<parameter name="ibm.portal.toolbar.isToolbarPage" type="string"
update="set"><![CDATA[true]]&gt;</parameter>
```

For performance reasons, this parameter has to be set on every toolbar content node. If it is not set, the toolbar tab will not have access to the context of the main page.

You can now start to add content to your toolbar. Add a portlet to the content page representing your toolbar tab. For more information, see *Implementing portlets for the site toolbar*.

The feed that displays all portlets is cached after being viewed. If you deploy a portlet after viewing the portlets on the toolbar, the newly deployed portlet will not be visible until the cache expires, which is one hour, or you look up the portlet in **Manage Portlets** and then go back to the toolbar.

HTML 5 considerations

All of the drag-and-drop actions are based on the native HTML 5 drag-and-drop capability. The content items that are displayed in the toolbar, for example on the **Create** > **Applications** or **Create** > **Content** tab, are marked as draggable in terms of HTML 5.

The Search controls that are displayed on some toolbar tabs, exploit the auto-completion capability of HTML 5. Internally it is based on the HTML 5 `<datalist>` tag. As a consequence, you might see different search result suggestions depending on the browser that you are using. Internet Explorer version 9 does not support the HTML 5 auto completion feature.

"Configuring the behavior of toolbar tabs"
You can use page parameters to configure the height of your toolbar tabs, and how the toolbar tabs respond to resizing events in the browser.

"Implementing portlets for the site toolbar" on page 1903
The toolbar tabs of the Portal 8.5 site toolbar are represented as portal pages. The content of the toolbar tabs can be implemented by using portlets.

"Customizing the Content palette" on page 1904
The Content palette, which is accessed from the site toolbar, contains the content items a content author can add to a page.

"Customizing the Applications palette" on page 1909
WebSphere Portal Express includes ready to use portlets that you can browse through in the site toolbar. To ease browsing through this set of portlets, WebSphere Portal Express supports assigning portlets to one or more categories. As the site administrator, you can create, delete, and modify categories and assign portlets to them.

"Customizing the Page tab" on page 1912
As an administrator, you can customize the Page tab to include other layouts and styles that content authors can use when they are editing a page.

CF05 "Controlling the visibility of the site toolbar and toolbar tabs" on page 1914
You can control the visibility of the site toolbar and single toolbar tabs per virtual portal by using Portal Access Control.

Related information:

Enabler ContextMenu API

Enabler API Quick Reference

Configuring the behavior of toolbar tabs

You can use page parameters to configure the height of your toolbar tabs, and how the toolbar tabs respond to resizing events in the browser.

You can set the page parameters for the site toolbar by selecting the **Edit Page Properties** icon for the hidden page Toolbar Content Root. Expand **Advanced options** and select **I want to set parameters**. Use the following parameters to control the size of your toolbar tabs.

`ibm.portal.toolbar.reloadOnResize = true | false`
Use this configuration parameter to indicate whether you want your toolbar tab to be reloaded automatically if a resize event occurs in the browser. Depending on how your content is rendered, it might be useful to set this parameter to true. Defaults to false. For example:

```
<parameter name="ibm.portal.toolbar.reloadOnResize" type="string"
update="set"><![CDATA[true]]&gt;</parameter>
```

ibm.portal.toolbar.minHeight = *floating point number*

Use this configuration parameter to specify the minimum height factor of your toolbar tab. The value must be greater (or equal) than 0.2 and less (or equal) than 0.9. The factor is relative to the available viewport height on the client. A value of "0.3" for example means that the toolbar tab must at least consume 30 percent of the available viewport height. This configuration parameter must be combined with the `ibm.portal.toolbar.maxHeight` parameter that is explained next. For example,

```
<parameter name="ibm.portal.toolbar.minHeight" type="string"
update="set"><![CDATA[0.35]]&gt;</parameter>
```

ibm.portal.toolbar.maxHeight = *floating point number*

Use this configuration parameter to specify the maximum height factor of your toolbar tab. The value must be greater than or equal to the value that is specified for the minimum height. It must not be greater (or equal) than 0.9. The factor is relative to the available viewport height on the client. A value of "0.75" for example means that the toolbar tab must not consume more than 75 percent of the available viewport height. This configuration parameter must be combined with the previous `ibm.portal.toolbar.minHeight` parameter. For example,

```
<parameter name="ibm.portal.toolbar.maxHeight" type="string"
update="set"><![CDATA[0.7]]&gt;</parameter>
```

If the values of the configuration parameters `ibm.portal.toolbar.minHeight` and `ibm.portal.toolbar.maxHeight` are equivalent, the toolbar framework enforces this height, regardless of the size of the tab content that is being loaded.

Implementing portlets for the site toolbar

The toolbar tabs of the Portal 8.5 site toolbar are represented as portal pages. The content of the toolbar tabs can be implemented by using portlets.

Use the Java Portlet API to implement these portlets. For more information, see Predefined public render parameters.

Accessing the main portal page running in the view area

If you want your toolbar portlet to access the main portal page that is currently displayed in the view area, you can use the following public render parameter. This parameter carries the serialized ObjectID of the main portal page, not the one of the toolbar content page. Use this ObjectID to locate the content page in the content model hierarchy of your portal using the public Model SPI or the Remote Model SPI if you want to query page information using REST. You can display properties of the main page in your toolbar portlet or even modify page properties by using the Controller SPIs.

```
{http://www.ibm.com/xmlns/prod/websphere/portal/publicparams}selection
```

Working with page edit mode and page view mode

You can configure your toolbar portlet to differentiate between page edit mode and page view mode. In page edit mode, your content authors, page editors and administrators can modify the main portal page that is currently displayed in the main view area. In page view mode, you can provide read-only information about the main portal page. You can use the following public render parameter to detect if page edit mode is active or inactive.

```
{http://www.ibm.com/xmlns/prod/websphere/portal/publicparams}editMode
```

Working with page information mode

Use the following public render parameter if you want your toolbar portlet to display additional help information for your content authors, page editors and administrators. Or, use it if page information is active.

`{http://www.ibm.com/xmlns/prod/websphere/portal/publicparams}infoMode`

Drag and Drop

If you want your toolbar tab to display a list of items that can be dragged to the portal page that is currently displayed in the main view area, you need to make these items draggable according to the Drag and Drop model of HTML 5. For more information, see *Customize drag and drop*.

Customizing the Content palette

The Content palette, which is accessed from the site toolbar, contains the content items a content author can add to a page.

These content items act as templates. When added to a page, the item is copied. Authors can adapt the item according to their needs. The top-level view of the Content palette contains categories. To find a certain content item, the content author can drill down in the category structure. You can use keywords or content associations to define the categories that are shown in the top-level view.

"Creating new categories for the Content palette by using Web Content Authoring Portlet"
For a site area, add a keyword to profiles to define categories in the Content palette. Content items in the site area appear in the new category of the site toolbar.

"Creating new categories for the Content palette by using the Manage Pages portlet" on page 1905
Associate content with the hidden Content page to define categories in the Content palette. Content items under the associated site area appear in the new category of the site toolbar.

"Creating Content Drag and Drop configurations" on page 1906
Configure the results that display when a content author searches for a portlet to add. You can also enable assistance for content authors by displaying supported markup in search results.

"Creating custom images for categories in the Content palette" on page 1906
You can assign a custom image to an item in the Content palette. You can either use a single image or a collection of images that are bundled into a `ZIP` file, called multi-scale icons. Supported image file formats are `PNG`, `JPG`, and `SVG`. If you are using multi-scale icons, the collection of images must be compressed into a `ZIP` file. No other format is supported. You can add custom images by either adding a file element directly into the content, or by creating a reference to a file component.

Creating new categories for the Content palette by using Web Content Authoring Portlet:

For a site area, add a keyword to profiles to define categories in the Content palette. Content items in the site area appear in the new category of the site toolbar.

About this task

You can create the site area and add the keyword at the same time. If you have a site area that contains content items that you want to add to the site toolbar, you can use the site area keywords to define the category in the toolbar. For example, you have a site area that is called, "Product Content". Add the keyword, `ibm.portal.toolbar.NewContent` to the Profile section of the site area to create a category that is called "Product Content" in the Content palette in the site toolbar.

Procedure

1. Click the **Applications** menu icon in the toolbar.
2. Click **Content** > **Web Content Authoring**.
3. Find or create the site area to use as the new category.
4. Expand the Profile section of the site area.
5. Add the keyword `ibm.portal.toolbar.NewContent`.
6. Save and close the dialog.

Related information:

Creating site areas

Creating new categories for the Content palette by using the Manage Pages portlet:

Associate content with the hidden Content page to define categories in the Content palette. Content items under the associated site area appear in the new category of the site toolbar.

Before you begin

The site area must exist.

About this task

The benefit to creating new categories for the Content palette by using the Manage Pages portlet is that you can view all libraries. You can therefore create a category from any of the libraries that are available.

Procedure

1. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
2. Select **Content Root** > **Hidden Pages** > **Toolbar Content Root** > **Create**.
3. Click **Edit Page Properties** of the Content palette.
4. Open **Advanced options**.
5. Select **I want to edit associations**.
6. Select **Add web content**.
7. Select the category that you want to show in the Content palette.
8. Click **OK**.

Related information:

Creating site areas

Creating Content Drag and Drop configurations:

Configure the results that display when a content author searches for a portlet to add. You can also enable assistance for content authors by displaying supported markup in search results.

About this task

When you drag an item from the Content palette to the page, the item is copied and a content viewer portlet that points to the new copy is added to the page. The Drag and Drop configurations are portal pages that contain a single portlet. The Drag and Drop configuration is defined by the preferences of the corresponding portlet entity. The Drag and Drop configuration pages can be found in **Content Root** > **Hidden Pages** > **DnD Configuration**. A Drag and Drop configuration can be set on categories or on single content items. When a configuration is set on a category, the configuration is applied to all child items, unless they have their own configuration. The available Drag and Drop configurations are listed on the configuration tab inside the Content palette.

To change the configuration for an item, Select the item so that the item details are shown. If you are in Advanced mode, open the **Configuration** tab. Select the configuration that you want to apply.

Procedure

1. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
2. Select **Content Root** > **Hidden Pages** > **DnD Configuration**.
3. Click **New Page** and give the page a name.
4. Click **OK**.
5. Click **Edit Page Layout** for the new page.
6. Click **Add Portlets**.
7. Select **Web Content Viewer** and click **OK**.
8. Click **Done**.
9. Click **Edit Page Layout** for the new page.
10. Open the context menu and click **Edit Shared Settings**.
11. Set the preferences for the new configuration.
12. Click **OK**.
13. Click **Done**.
14. Optional: To enter a description for your configuration, click **Edit Page Properties** for the new page.
    a. Open the **Advanced** options.
    b. Select **I want to set titles and descriptions**.
    c. Click **Edit** for the language you want to use.
    d. Enter the descriptions and click **OK**.

Creating custom images for categories in the Content palette:

You can assign a custom image to an item in the Content palette. You can either use a single image or a collection of images that are bundled into a `ZIP` file, called multi-scale icons. Supported image file formats are `PNG`, `JPG`, and `SVG`. If you are using multi-scale icons, the collection of images must be compressed into a `ZIP` file.

No other format is supported. You can add custom images by either adding a file element directly into the content, or by creating a reference to a file component.

Multi-scale icons

If you are using multi-scale icons, each individual image is a single entry in the ZIP file. Images with the same base name represent one logical icon. The variations of an image are size, device class, and locale. These variations are organized in the following structure:

```
<Width>x<Height>/<FileName>_<DeviceClass>_<Locale>_<TextDirection>.<Suffix>
```

Where <Width> and <Height> group the icons by size. These values are a mandatory top-level directory. <DeviceClass>, <Locale>, and <TextDirection> are optional identifiers. The variations are not a part of the base name. The base name of the image is formed by <FileName>.<Suffix>, for instance, Applications.png. The following example is a sample directory structure inside a compressed file. The sample directory structure defines one icon, Applications.png, in different variations:

```
32x32/
     Applications_ar.png
     Applications_he.png
     Applications_ltr.png
     Applications.png
64x64
     Applications_de_de.png
     Applications_de.png
     Applications_rtl.png
     Applications.png
     Applications_tablet_de.png
     Applications_tablet.png
```

Applications.png has two size variations, 32x32 and 64x64. The smaller variation (32x32) defines an icon for ar and he variations in addition to a generic fallback icon. The smaller variation also defines an icon for languages that are read left to right by including a <TextDirection> variation that specifies ltr. The large one (64x64) includes a variation for de_de that specifies the German language in the locale of Germany in addition to a fallback icon for de. The <TextDirection> variation that specifies rtl defines an icon for languages that are read right to left. <DeviceClass> variations are also included for tablet devices.

Note: The size subdirectory must be the top-level directory in the compressed file.

When the icon is rendered, the best match is found in the following way:

1. The size is specified based on the requested icon size, by using the Euclidean distance. If no size is requested, the largest available size is used.
2. For all variations of a specified size, the best matching device class is used.
3. For all variations of a specified size and device class, the best matching locale is used.

"Creating custom images for categories in the Content palette by adding a file element" on page 1908
You can create custom images for items in the Content palette by deploying the image file or multi-scale ZIP file directly into the item in the Content palette as an element.

"Creating custom images for categories in the Content palette by referencing a file component" on page 1908
You can create custom images for items in the Content palette by deploying the

image file or multi-scale ZIP file as a file component in a library. Then, put a component reference element into your item in the Content palette that refers to that file component.

Creating custom images for categories in the Content palette by adding a file element:

You can create custom images for items in the Content palette by deploying the image file or multi-scale ZIP file directly into the item in the Content palette as an element.

About this task

- Supported image file formats are PNG, JPG, and SVG.
- If you are using multi-scale icons, the collection of images must be compressed into a ZIP file.

Procedure

1. Click the **Applications** menu icon in the site toolbar.
2. Click **Content** > **Web Content Authoring**.
3. To assign a custom image to a content item or a site area, select the item and click **Edit**.
4. Click the **More** menu.
5. Click **Manage Elements**.
6. In the **Element type** dropdown menu, select **File**.
7. In the **Name** field, specify the name of the new element as `preview-image`.
8. Click **Add**.
9. Click **OK** to exit the Element Manager window.
10. Upload the compressed file or image file for the *my_icon* resource element.
11. Click **Save and Close**.
12. Clear your browser cache and refresh the page for the changes to take effect.

Creating custom images for categories in the Content palette by referencing a file component:

You can create custom images for items in the Content palette by deploying the image file or multi-scale ZIP file as a file component in a library. Then, put a component reference element into your item in the Content palette that refers to that file component.

About this task

- Supported image file formats are PNG, JPG, and SVG.
- If you are using multi-scale icons, the collection of images must be compressed into a ZIP file.

Procedure

1. Click the **Applications** menu icon in the site toolbar.
2. Click **Content** > **Web Content Authoring**.
3. Navigate to the library where you want to store the image as a file component.
4. Click **New** > **Component** > **File**.
5. Enter a unique name for the new component. For example, *my_icon*.
6. Upload the file and click **Save and Close**.

7. To assign a custom image to a content item or a site area, select the content item and click **Edit**.
8. Click the **More** menu.
9. Click **Manage Elements**.
10. In the **Element type** dropdown menu, select **Component Reference**.
11. In the **Name** field, specify the name of the new element as `preview-image`.
12. Click **Add**.
13. Click **OK** to exit the Element Manager window.
14. Click **Select Component** for the *my_icon* component reference. The Select a component window displays.
15. Navigate to the library where you saved the *my_icon* component reference.
16. Select the ***my_icon*** file component.
17. Click **OK**.
18. Click **Save and Close**.
19. Clear your browser cache and refresh the page for the changes to take effect.

Customizing the Applications palette

WebSphere Portal Express includes ready to use portlets that you can browse through in the site toolbar. To ease browsing through this set of portlets, WebSphere Portal Express supports assigning portlets to one or more categories. As the site administrator, you can create, delete, and modify categories and assign portlets to them.

Application categories

In WebSphere Portal Express, categories are represented as labels or external URLs and can be managed with the Manage Pages portlet. All categories are in the category root label that is identifiable by the unique name `com.ibm.portal.toolbar.applications.category.label.root`, which is a label in the hidden pages label.

Categories are shown by the toolbar's Applications palette where categories appear with said localized title.

The following categories are available for immediate use:

- All
- Administration
- Collaboration
- Web Content
- Tools

"Creating new applications and categories for the site toolbar" on page 1910
You can add new categories to the site toolbar to organize your applications. After you deploy portal, you can customize the **Create** > **Applications** user interface in the site toolbar. Add categories that reflect your business needs and website organization.

"Creating categories by using external URLs" on page 1910
You can also use external URLs to represent categories to organize your portlet entities. The advantage of representing labels, or external URLs, with categories is that administrators can easily add, delete, and modify categories by using the Manage Pages portlet as the administrative user interface. Since categories are

labels, or external URLs, they can be assigned localized titles and descriptions, which can be set with the Manage Pages portlet as well.

"Hiding portlets within Applications palette categories" on page 1911
If a portlet has been added to a category by adding it to a page that has been placed under a category label, remove the page it has been added to. Alternatively, if you do not want to delete the page, you can mark it as hidden.

"Modifying and deleting Applications palette categories" on page 1912
You can easily modify or delete Applications palette categories.

Creating new applications and categories for the site toolbar:

You can add new categories to the site toolbar to organize your applications. After you deploy portal, you can customize the **Create** > **Applications** user interface in the site toolbar. Add categories that reflect your business needs and website organization.

About this task

Portlets that are deployed on the portal server are automatically added to the All category.

The site toolbar structure is based on labels and pages. Create a label to add a category. Then, create a page for each application that you need to add to the site toolbar. The page title must match the name of the application.

Procedure

1. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
2. Locate the category root label. For example, search for a title that contains "Application" and then look for the unique name `com.ibm.portal.toolbar.applications.category.label.root` in the results.
3. Click **New Label**.
4. Enter the name of the category in the **Title** field. Click **OK**.
5. Click the new label.
6. Click **New Page** to add a child page.
7. Enter the name of the application that you want to add to the site toolbar in the **Title** field and click **OK**.
8. Click **Edit Page Layout** and add the portlet to the page. Click **Done**
9. Verify that the category and portlet appear on the Applications palette.

Results

If your content authors do not see the portlet that you added, the site toolbar might be cached. The feed that displays all portlets is cached after the content author views it. If you deploy a portlet and the site toolbar is already cached, the newly deployed portlet is not visible until the cache expires, which is 1 hour. To see the portlet sooner, look up the portlet in the **Manage Portlets** administration portlet and then go back to the site toolbar.

Creating categories by using external URLs:

You can also use external URLs to represent categories to organize your portlet entities. The advantage of representing labels, or external URLs, with categories is

that administrators can easily add, delete, and modify categories by using the Manage Pages portlet as the administrative user interface. Since categories are labels, or external URLs, they can be assigned localized titles and descriptions, which can be set with the Manage Pages portlet as well.

About this task

To assign portlets to these categories, you must point the external URL to a feed that returns either a list of portlet definitions or a list of portlet entities. These portlet definitions or portlet entities then represent the portlets that are shown as being part of the Applications palette.

Procedure

1. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
2. Locate the category root label. For example, search for the label with the unique name `com.ibm.portal.toolbar.applications.category.label.root`.
3. Click **New URL** to add a new label.
4. Add a title and description for the new category. Click **OK**.
5. In the **Advanced Options** section, select **HTML** as supported markup. Specify a URL pointing to a feed that returns a list of portlet definitions or portlet entities.
6. Click **OK**.
7. Verify that the category and portlet entities appear on the Applications palette.

Hiding portlets within Applications palette categories:

If a portlet has been added to a category by adding it to a page that has been placed under a category label, remove the page it has been added to. Alternatively, if you do not want to delete the page, you can mark it as hidden.

The following XMLAccess example marks a page as hidden.

```
<?xml version="1.0" encoding="UTF-8"?>
<request xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" type="update" xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd">
    <portal action="locate">

        <content-node action="locate" domain="rel" uniquename="wps.content.root"/>
        <content-node action="locate" domain="rel" uniquename="ibm.portal.HiddenPages"/>
        <content-node action="locate" domain="rel" uniquename="com.ibm.portal.toolbar.applications.category.label.root"/>

        <content-node action="update" active="true" content-parentref="com.ibm.portal.toolbar.applications.category.label.root" domain="rel" ordinal="700"
type="label" uniquename="com.ibm.portal.toolbar.applications.category.label.administration">
            <supported-markup markup="html" update="set"/>
            <localedata locale="en">
                <title>Administration</title>
            </localedata>
            <parameter name="com.ibm.portal.Hidden" type="string" update="set"><![CDATA[true]]&gt;</parameter>
            <parameter name="com.ibm.portal.IgnoreAccessControlInCaches" type="string" update="set"><![CDATA[false]]&gt;</parameter>
            <parameter name="com.ibm.portal.remote-cache-expiry" type="string" update="set"><![CDATA[86400]]&gt;</parameter>
            <parameter name="com.ibm.portal.remote-cache-scope" type="string" update="set"><![CDATA[NON-SHARED]]&gt;</parameter>
            <parameter name="param.sharing.scope.{http://www.ibm.com/xmlns/prod/datatype/content/resource-collections}" type="string" update="set">
<![CDATA[ibm.portal.sharing.scope.page]]&gt;</parameter>
            <parameter name="param.sharing.scope.{http://www.ibm.com/xmlns/prod/datatype/content}" type="string" update="set"><![CDATA[ibm.portal.
sharing.scope.page]]&gt;</parameter>
            <parameter name="param.sharing.scope.{http://www.ibm.com/xmlns/prod/websphere/portal/publicparams}path-info" type="string" update="set">
<![CDATA[ibm.portal.sharing.scope.page]]&gt;</parameter>
            <access-control externalized="false" owner="undefined" private="false">
                <role actionset="User" update="set">
                    <mapping subjectid="all authenticated portal users" subjecttype="user_group" update="set"/>
                </role>
            </access-control>
        </content-node>
    </portal>
</request>
```

If a portlet becomes part of a category because the category label points to a feed returning portlet definitions (which contains the portlet), you can force the portlet

to not appear by marking the portlet hidden. You can only mark the portlet hidden as long as the feed exposes portlet preferences, like the PortletDefinitionList feed does.

The following example marks a portlet as hidden.

```
<?xml version="1.0" encoding="UTF-8"?>
<request require-defined-oids="true" type="update" version="8.5.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd">
    <portal action="locate">
        <web-app action="update" active="true" objectid="Z1_00000000000000AGU4RF6L0004" uid="sample.webmod">
            <url>file:///sample.war</url>
            <servlet action="update" name="Sample" objectid="ZV_00000000000000AGU4RF6L0002"/>
            <servlet action="update" name="Sample" objectid="ZV_00000000000000AGU4RF6L0000"/>
            <portlet-app action="update" name="sample" objectid="Z2_00000000000000AGU4RF6L0006" uid="sample">
                <access-control externalized="false" owner="undefined" private="false">
                    <role actionset="User" update="set">
                        <mapping subjectid="all authenticated portal users" subjecttype="user_group" update="set"/>
                    </role>
                </access-control>
                <portlet action="update" name="Sample" objectid="Z3_00000000000000AGU4RF6L0001"
servletref="ZV_00000000000000AGU4RF6L0000" uniquename="wps.p.Sample">
                 <preferences name="com.ibm.portal.Hidden" update="set">
                     <value><![CDATA[true]]&gt;</value>
                 </preferences>
                </portlet>
            </portlet-app>
        </web-app>
    </portal>
</request>
```

Modifying and deleting Applications palette categories:

You can easily modify or delete Applications palette categories.

About this task

To delete a category, delete the label or external URL of the category's root label. To modify a category, edit the label or external URL of the category's root label.

Customizing the Page tab

As an administrator, you can customize the Page tab to include other layouts and styles that content authors can use when they are editing a page.

The styles and layouts that are available in the Page tab are included as a part of the theme. However, other styles and layouts can be added without modifying the existing theme.

"Creating new styles or layouts by using a JSON file"
You can add new styles or layouts to the site toolbar with a JSON file. Sometimes components outside of the theme want to contribute a new style or layout, without editing existing theme files. Components can easily be added to the theme through a catalog deliverable by using scripting and the Solutions Installer.

Creating new styles or layouts by using a JSON file:

You can add new styles or layouts to the site toolbar with a JSON file. Sometimes components outside of the theme want to contribute a new style or layout, without editing existing theme files. Components can easily be added to the theme through a catalog deliverable by using scripting and the Solutions Installer.

Procedure

1. Create new styles and layouts.

2. Upload the styles or layouts to the portal. The out of box style and layout examples are in WebDAV at `dav:fs-type1/themes/Portal8.5/css/` and `dav:fs-type1/themes/Portal8.5/layout-templates`.
3. Create a JSON file to list and define the new styles and layouts. Base the style JSON file on the format in the topic *Creating a theme style*. The default JSON files can be used as examples and are in WebDAV at `dav:fs-type1/themes/Portal8.5/system/`.
4. Add the new JSON files to the portal.
5. Add theme metadata for each JSON file. The metadata keys are `ibm.portal.shelf.style.json.`*component* and `ibm.portal.shelf.layout.json.`*component*, for the style JSON files and layout JSON files, where *component* can be any string that uniquely identifies the contribution. The metadata values are URLs pointing to the JSON file location. Values that start with a forward slash (/) are treated as absolute URLs, and all other values are read relative to the theme root folder. By default, the theme root folder is in WebDAV at `dav:fs-type1/themes/Portal8.5/` The following example shows style and layout theme metadata for a component called **myAdditions**:

```
<parameter name="ibm.portal.shelf.style.json.myAdditions" type="string" update="set">

     <![CDATA[system/myNewStyles.json]]&gt;</parameter>

     <!-- This URL will be read relative to the theme root folder -->

<parameter name="ibm.portal.shelf.layout.json.myAdditions" type="string" update="set">

     <![CDATA[/myContextRoot/layouts/myNewLayouts.json]]&gt;</parameter>

     <!-- This is an absolute URL to a custom war file -->
```

   a. To create and run an XML access script, use the following example as a model.

```
<?xml version="1.0" encoding="UTF-8"?>
<request build="wp85" type="update" version="8.5"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd">
 <portal action="locate">
  <theme action="update" active="true"
   default="true" domain="rel"
   resourceroot="dynamicSpots" uniquename="ibm.portal.85Theme">
   <parameter name="ibm.portal.shelf.style.json.myAdditions"
type="string" update="set">
       <![CDATA[system/myNewStyles.json]]&gt;</parameter>
       <!-- This URL will be read relative to the theme root folder -->
    <parameter name="ibm.portal.shelf.layout.json.myAdditions" type="string" update="set">
       <![CDATA[/myContextRoot/layouts/myNewLayouts.json]]&gt;</parameter>
       <!-- This is an absolute URL to a custom war file -->
  </theme>
 </portal>
</request>
```

   b. Copy the script to *PortalServer_root*`/bin`.
   c. Run the XML Access command: `xmlaccess -user` *user* `-password` *password* `-url localhost:`*port*`/wps/config -in AddCategories.xml -out AddCategoriesOutput`
6. Log in to the portal and click **Edit Mode**.
7. Verify that the new styles and layouts are present:
   - If you have portal with combined cumulative fix 7 or earlier, proceed as follows:
     a. Select the **Page** tab.

b. Then, select the **Style** or **Layout** tab.

Related information:

How to add a layout and style to the toolbar

Controlling the visibility of the site toolbar and toolbar tabs

You can control the visibility of the site toolbar and single toolbar tabs per virtual portal by using Portal Access Control.

The visibility of the action bar that contains the **Edit Mode** switch and the global menus can be controlled by assigning resource permissions to the toolbar master page. The toolbar master page has the unique name `ibm.portal.Toolbar` and is located within the Hidden Pages Root in the portal page hierarchy. By default the toolbar master page is accessible for anonymous portal users and authenticated portal users.

The visibility of the toolbar tabs can be controlled by assigning resource permissions to the toolbar content root or one of its sub pages. The toolbar content root has the unique name `ibm.portal.toolbar.ContentRoot` and is located within the Hidden Pages Root in the portal page hierarchy. By default the toolbar tabs are accessible to all authenticated portal users but not to anonymous portal users.

You can change the resource permissions for certain users or user groups. You can use the Resource Permissions portlet or the XML configuration interface to change the resource permissions.

You can completely disable the site toolbar along with the toolbar tabs for the entire system or for a certain virtual portal. To disable the site toolbar, follow the configuration settings that are documented in *Removing the site toolbar on a production server.*

CF05 "Limiting toolbar visibility to administrators only"
You can limit toolbar visibility to administrators only.

Related tasks:

CF05 "Removing the site toolbar on a production server" on page 1920
The site toolbar provides access to editing features for managed pages, including adding and editing pages and web content. Although essential for an authoring server, it is recommended that you disable the site toolbar on a delivery server. You can disable the toolbar for an entire portal or for specific virtual portals.

Limiting toolbar visibility to administrators only:

You can limit toolbar visibility to administrators only.

Procedure

1. Click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage pages**.
2. Select **Unique name contains** and search `ibm.portal.Toolbar`.
3. Select **Set Page Permission** for Toolbar.
4. From the Resource Permission Portlet, click **Edit Role** for User.
5. Click **Delete Member from Role**.
6. Click **OK**.
7. Click **Toolbar**to return to the toolbar menu.
8. Clear the check box for **Privileged User**.

9. Click **Apply**.
10. Click **OK**.

Customizing the Web Content Authoring portlet

To customize each Web Content Authoring portlet for the people who use that portlet, you can edit the shared settings of each Web Content Authoring portlet on your authoring system.

About this task

For example, you can configure authoring portlets in the following ways:

- Select only the libraries that your users use. For example, if you configure an authoring portlet that you want to be used only by content creators, select only libraries that are used to store content items.
- Edit the preview options to best suit your users and the type of website they are creating. You can choose to preview pages on a standard website, on a local web content viewer on the same server as the authoring portlet, or on a web content viewer portlet on a different server.
- Customize the appearance of the authoring portlet by defining various user interface settings. You can use these settings to change some of the default settings of an authoring portlet, or to select a custom launch page to use in place of the default user interface.
- Select an appropriate rich text editor for your users.

In IBM WebSphere Portal Express, the Web Content Authoring portlet is on a hidden page. If you want to configure the portlet to use only a subset of the available site libraries, proceed as follows:

Procedure

1. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
2. Search for a page with the unique name `com.ibm.wps.hiddenpage.wcm.Authoring_Portlet`.
3. Select **Edit Page Layout**.
4. From the portlet menu, select **Edit shared settings**. The portal shows a dialog window.
5. In the window, click **Library Selection**.
6. Configure which libraries you want to use for the authoring portlet.
7. Click **OK** in the dialog window.
8. Click **Done** in the main window.

Results

You successfully configured the libraries for the Web Content Authoring portlet.

"Authoring portlet settings"
An authoring portlet is used to create and manage web content. You can edit the settings of an authoring portlet from within the **Preferences** section of the authoring portlet.

Authoring portlet settings

An authoring portlet is used to create and manage web content. You can edit the settings of an authoring portlet from within the **Preferences** section of the authoring portlet.

Use the **Configure** mode to specify settings for all users of all instances of the authoring portlet, regardless of the page on which the portlet instance appears.

Use the **Shared Settings** mode to specify settings for the current instance of an authoring portlet.

"Selecting web content libraries"
You can select which libraries are displayed by default in the authoring portlet.

"Defining preview options" on page 1917
The preview options determine how content can be previewed. Preview options are defined within the **Previewing Options** section.

"Defining user interface options" on page 1917
You use the **User Interface Options** section to define the user interface options of an authoring portlet.

"Editor options" on page 1919
You can configure IBM Web Content Manager to use different editors for rich text and HTML fields.

Selecting web content libraries:

You can select which libraries are displayed by default in the authoring portlet.

Procedure

1. Select **Show selected libraries** to select the libraries you want to make visible in the authoring portlet.
   a. To add a library, select a library from the list of available libraries, then click **Add**.
   b. To remove a library, select a library from the list of selected libraries, and then click **Remove**.
   c. Use the arrows to change the order of the selected libraries. This selection determines the order that the libraries appear in the authoring portlet.
2. Select **Show new libraries in the library explorer** if you want any newly created libraries to automatically be shown in the library explorer.
3. Alternately, to make all libraries visible in the authoring portlet, select **Show all libraries**. You can then select individual libraries to hide in the authoring portlet.
   a. To hide a library, select a library from the list of available libraries, and then click **Add**.
   b. To remove a library from the list, select a library from the list of selected libraries, and then click **Remove**.

What to do next

Configuring or editing shared settings of an authoring portlet:

Libraries that are selected by using the "configure" view are available on all instances of the authoring portlet, regardless of the page on which the portlet appears. Libraries that are selected with the "edit shared settings" view are only available for the current instance of an authoring portlet.

The libraries available in the "insert links" and "insert images" dialogs are based on the libraries that are selected in the "configure" view. If you select a library that is not selected in the "configure" view, you cannot select items from this library in the "insert links" and "insert images" dialogs.

You can select libraries specifically for the "insert links" and "insert images" dialogs by following these steps:

1. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
2. Search for the page with the unique name of *com.ibm.wps.hiddenpage.wcm.Authoring_Portlet*.
3. Edit the page layout.
4. Edit the shared settings of the web Content Authoring portlet.
5. Select the required libraries and click **OK**.
6. Click **Done**.

Note:

If you syndicate or import a library, it is not automatically added to the list of configured libraries for an authoring portlet on the target server. You must add the syndicated or imported library to each authoring portlet on each server.

Defining preview options:

The preview options determine how content can be previewed. Preview options are defined within the **Previewing Options** section.

Procedure

1. Select **Allow authors to preview content in a Web page** to allow users to preview pages with the Web Content Manager servlet.
2. To allow users to preview content in portal pages, you must select specific portal pages from the list that is located under **Allow authors to preview content in the local portal pages selected**. The portal pages that are displayed here are the pages available on the same instance of the portal where your Web Content Manager application is installed. The selected pages must contain a web content viewer to display the content.
3. To allow users to preview content in a portlet that is on a different portal server, you must enter the URL to the remote portal page in the **Allow authors to preview content using the following URLs** field. The portal pages that are entered here must contain a web content viewer to display the content.

What to do next

Note: When you use a web content viewer to preview content, ensure that the web content viewer is configured to receive links from **Other portlets and this portlet**.

Defining user interface options:

You use the **User Interface Options** section to define the user interface options of an authoring portlet.

Procedure

1. Select a visibility option for the navigation bar. This section is the part of the authoring portlet that displays navigational links to the item views, group by views, and personal views.

   Show If this option is selected, the navigation bar is visible when the home page, launch page, or item forms are displayed.

Hide If this option is selected, the navigation bar is hidden when the home page, launch page, or item forms are displayed. This selection is for users who need to work on basic web content management tasks only.

Hide when home page or launch page is open
If this option is selected, the navigation bar is hidden when the home page or launch page is displayed, but is visible when the item forms are displayed. The initial interface is made simpler, while still having the navigation bar available in other views. This option is for users who need to work on more complex web content management tasks.

Note: The navigation bar is always displayed when you use the library explorer.

2. Select a default view:

Basic home page
If selected, the basic home page is displayed when you first access the authoring portlet. The basic home page is designed for content authors who need to create content items and other simple item types only. They would not usually need access to the more advanced library explorer. The basic home page allows the creation of content with up to six different authoring templates. The authoring templates that are displayed are the favorites of the current user, and the most recently used by the current user.

You can define a set of authoring templates to display on the home page when a user does not have any favorites or recently used items. Add the *wcm.authoringui.homePageTemplates* parameter to the authoring options in the `WCM WCMConfigService` service from the IBM WebSphere Application Server administration console. The paths of the templates must be specified, separated by colons. For example:

```
Library 1/Template Name:Library 2/folder/Template Name 2:Library 3/Template Name 3
```

Home page
If selected, the standard home page is displayed when you first access the authoring portlet. One section of the home page is used to create items and open their favorite locations. Another section of the home page is used to view recent activity. This interface is designed for content authors who need to complete tasks such as creating content, editing their draft content and approving and declining content. They would not usually need access to the more advanced library explorer.

Launch page
If you created a custom launch page to use in place of the default user interface, select **Launch Page**. For example, `file.jsp`.

A custom launch page is a JSP file and must be stored in the WAR file directory for the Authoring portlet. *AppServer_root*`/installedApps/`*cellname*`/PA_WCM_Authoring_UI.ear/ilwwcm-authoring.war/jsp/html`, where *cellname* is unique to your installation. Enter the name of the JSP file in the custom launch page field.

Library Explorer
To use the default user interface, select **Library Explorer**.

3. Select **Hide the open item and view lists** to hide these functions from users in the authoring portlet.
4. To improve performance, you can limit the number of tasks a user can open at the same time in the authoring portlet by entering a number in the **Maximum open tasks per Authoring Portlet instance** field.

5. To improve performance, you can limit the number of items a user can select in an index at the same time in the authoring portlet by entering a number in the **Maximum selected items per action before warning** and **Maximum selected items per action before denying the action** fields.
6. The number of rows that appear in an index is defined in the **Maximum rows per table** field. If blank, this setting defaults to 10. A maximum of 500 rows can set.
7. To enable People Awareness, select **Enable people awareness**. People Awareness helps users to select user names that appear in views and forms within the Authoring Portlet, and send those users an email or Sametime message.
8. Select the default display mode of the library explorer.

   List view mode:
   This mode displays lists of items only as you browse a library.

   Tree view mode:
   This mode displays both lists of items plus a navigational tree as you browse a library.

Editor options:

You can configure IBM Web Content Manager to use different editors for rich text and HTML fields.

Rich text editor

Default
Select this option to use the default JavaScript editor. This editor does not require a working Java runtime environment on the client computer.

EditLive! JavaScript Editor
Select this option to use the EditLive! JavaScript editor. This editor does not require a working Java runtime environment on the client computer. Browser pop-up windows must be enabled to use this editor.

EditLive! Java Editor
Select this option to use the EditLive! Java Editor. This editor requires a working Java runtime environment on the client computer. Browser pop-up windows must be enabled to use this editor.

Custom

Select **Custom** to use a third-party rich text editor as your default editor. Before you enable a compatible third-party rich text editor, read the installation and configuration instructions of the third-party rich text editor. These instructions should include steps for enabling the third-party rich text editor to be used in a Web Content Manager solution.

When you configure a third-party rich text editor, you need to copy a JSP file that is supplied by the third-party rich text editor. This file is used to start the third-party rich text editor. You enter the name of this JSP file in the Rich Text Options section of the authoring portlet configuration.

If the third-party rich text editor is not available, the standard rich text editor is used.

HTML editor

Default
Select this option to use the default HTML editor.

Custom

Select **Custom** to use a third-party HTML editor as your default editor. Before you enable a compatible third-party HTML editor, read the installation and configuration instructions of the third-party HTML editor. These instructions should include steps for enabling the third-party HTML editor to be used in a Web Content Manager solution.

When you configure a third-party HTML editor, you need to copy a JSP file that is supplied by the third-party HTML editor. This file is used to start the third-party HTML editor. You enter the name of this JSP file in the HTML Options section of the authoring portlet configuration.

If the third-party HTML editor is not available, the default HTML editor is used.

Storing JSP files: JSP files are stored within a web application that runs on the portal. To reference a JSP file in another web application, use the following path: `contextPath;jspPath`. For example: `/wps/customapplication;/jsp/jspFilename.jsp`.

A dynamic context path value can be defined by adding a token to the context path that corresponds to a key and value pair to the Web Content Manager configuration service environment provider. When this key is used as the token in the jsp value field, it is replaced dynamically at render time. For example: `mycustomkey;myfile`, where *mycustomkey* is a constant within the Web Content Manager configuration service.

Removing the site toolbar on a production server

The site toolbar provides access to editing features for managed pages, including adding and editing pages and web content. Although essential for an authoring server, it is recommended that you disable the site toolbar on a delivery server. You can disable the toolbar for an entire portal or for specific virtual portals.

About this task

The site toolbar function is not typically needed on a delivery server, and disabling the site toolbar can improve performance on the delivery server.

Procedure

1. Log in to the WebSphere Integrated Solutions Console as an administrator.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Click **WP VirtualPortalConfigService**.
4. Update the appropriate configuration properties, depending on whether you want to affect the entire portal or a specific virtual portal.
   - To affect the entire portal, complete the following steps:
     a. Click **Custom properties**.
     b. Edit the **global.toolbar.enabled** property, and set the value to `false`.

     This setting disables the site toolbar for all virtual portals.
   - To affect a specific virtual portal, complete the following steps:

a. Click **Custom properties**.
b. To disable the site toolbar for the default virtual portal, edit the **default.toolbar.enabled** property, and set the value to `false`.
c. For each virtual portal other than the default where you want to disable the site toolbar, specify the following properties.

context.*virtual_portal_context***.property.toolbar.enabled**
Set the value to `false`. Replace *virtual_portal_context* with the context of the target virtual portal (for example, `context.vp1.property.toolbar`.

hostname.*virtual_portal_hostname***.property.toolbar.enabled**
Set the value to `false`. Replace *virtual_portal_hostname* with the host name of the target virtual portal (for example, `hostname.vp.example.com.property.toolbar.enabled`.

If defined, the **global.toolbar.enabled** property acts as a fallback setting for virtual portals that have no values defined.

For more information about prefixes, placeholders, and the order in which properties are evaluated, see *Virtual Portal Configuration Service*

Related information:

Virtual Portal Configuration Service

Developing and managing content

Use these tools and processes to develop and manage your website.

"Previewing as another user" on page 1922
You can preview changes to your website without logging out and logging on again as another user. This preview capability is used to quickly verify that users with different access levels see only content that they are authorized to see. You can preview changes as a specific user or as an unauthenticated user.

"Projects" on page 1923
Projects are used to change a set of items on your site and ensure that they are published together at the same time. A new project has a default workflow that allows content to be reviewed and approved before it is published.

"Workflow and change management" on page 1929
You can manage changes to web content items either by creating drafts, by using workflows, or adding items to projects.

"IBM Web Content Integrator" on page 1936
The Web Content Integrator is a solution for integrating externally managed web content with WebSphere Portal Express. By using standard content syndication feed technologies based on RSS 2.0, the Web Content Integrator provides a loosely coupled mechanism for transferring published content and metadata to the portal after they are approved in the source system. When the content and metadata are transferred to the portal, it is possible to use the built-in content management features of Web Content Manager to secure, personalize, and display the content to users.

"WebDAV" on page 1986
With WebDAV for IBM WebSphere Portal Express, you can use standard operating system tools to create, modify, and delete web content rather than the standard authoring portlet.

"Blogs" on page 1999
Use blogs and blog libraries to provide news and commentary on a variety of subjects pertinent to your intranet and extranet sites. Blogs and blog libraries

typically combine text with graphics and links to other blogs and web sites. Entries that you create and post are arranged in reverse-chronological order, with the newest entry displayed first. Readers can post comments about your entries, fostering discussions and online networking. You can manage your own blog entries and comment on other blog entries. You can also incorporate tagging and rating as you would with other WebSphere Portal content.

"Wikis" on page 2005
Use wikis to share community content on a variety of subjects pertinent to your intranet and extranet sites. Wikis typically combine text with graphics and links to other wikis and web sites. You can monitor and manage your own wiki articles.

"Installed portlets" on page 2009
Learn about the portlets that are provided with WebSphere Portal Express.

CF04 "Renditions" on page 2013
Renditions are different versions of an image component or element. Renditions can be thumbnails or smaller versions of an image formatted for mobile devices.

CF04 "Video start and end points" on page 2015
File components or elements that have a video file that is stored in an HTML5 video format, which includes MP4, webM, or ogg, are stored in Brightcove and have a Brightcove ID.

Previewing as another user

You can preview changes to your website without logging out and logging on again as another user. This preview capability is used to quickly verify that users with different access levels see only content that they are authorized to see. You can preview changes as a specific user or as an unauthenticated user.

About this task

While in edit mode, click **More**, and then click either **As User** or **As Unauthenticated User**. The page is updated according to the user that you are impersonating.

To stop previewing, click **Stop Previewing**.

You can modify the preview function to tailor preview behavior in the following scenario:

- User1 creates a project and a page within the project. The new page contains a portlet.
- User1 then attempts to preview the page as User2. In this case, User2 has the permissions to view the page but does not have the permissions to view the portlet that is on the page.
- User1 receives the following message: `You are not authorized to use this portlet`. This message displays to make it clear that a portlet is rendered here, if the user has the required permissions.

You can suppress the message in this scenario.

For more information about the access permissions that are required for previewing as another user, see *Access control for managed pages.*

Procedure

1. Log on to the WebSphere Integrated Solutions Console as an administrator.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Click **WP ConfigService**, and then click **Custom properties**.
4. Locate and click the property `portlets.unauthorized.visible.project`. If the property is not listed yet, create it new.
5. Set the value of the property to `false` to suppress the message. The default value is `true`.

Related concepts:

"Access control for managed pages" on page 1256
Access control for managed pages provides more capabilities than access control for standard portal pages. In addition to the access control features available for pages through portal administration, you can also apply IBM Web Content Manager features, like workflow and syndication, to access control.

Projects

Projects are used to change a set of items on your site and ensure that they are published together at the same time. A new project has a default workflow that allows content to be reviewed and approved before it is published.

"Project review state" on page 1924
If you select at least one approver for a project, the project must be submitted for review before it can be published.

"Project States" on page 1924
When you use a project, the project progresses through a series of states.

"Ways to publish a project" on page 1924
Publishing a project refreshes the live website with your changes.

CF05 CF05 "Project Publishing" on page 1925
An outline of the process of publishing a project.

"Drafts and projects" on page 1925
You can create, update, and approve pages in a draft state, without affecting the published site. When you work with pages, you can edit pages as drafts, manage drafts with approval workflow, and then syndicate to publish changes.

"Scope of edits" on page 1926
When you edit pages in a project, you use workflow for approving changes before you publish them until they are approved. If you edit pages without selecting a project, your changes are published immediately.

CF05 CF05 "Project Validation" on page 1927
An outline of the validation process.

"Projects and syndication" on page 1927
Projects are included in syndication, the method that is used by IBM Web Content Manager to replicate data from a web content library on a syndicator server to a web content library on a subscriber server. Although projects are syndicated with other items in a library that is being syndicated, you cannot use the subscriber copy of the project to update or publish your project. Work with projects on the syndicator server only.

"Projects and custom workflow actions" on page 1928
Each workflow stage contains default actions, but you can also create custom workflow actions by creating a custom workflow plug-in.

"Best practices for projects" on page 1928
Use these tips and guidelines to develop and publish projects more effectively.

Project review state

If you select at least one approver for a project, the project must be submitted for review before it can be published.

To enforce a review, a user with editor access or higher clicks **Submit for Review** from the toolbar when all the items in the project are "pending". A project can either require the approval of only a single approver or all approvers. Only when the project is approved will the project itself become pending. The project is then either automatically or manually published depending on how the project is configured.

When a project is in the review state:

- Users with editor access or higher can return a project to an active state by clicking **Withdraw from Review**.
- Approvers can approve a project by clicking **Approve Project** . A project can either require the approval of only a single approver or all approvers.
- Approvers can decline a project by clicking **Reject Project**. The project is returned to the active state.
- If joint approval is specified, approvers can withdraw an approval while the project is still in review by clicking **Withdraw Approval**. The users's approval is withdrawn, but the project remains in a state of review.

Project States

When you use a project, the project progresses through a series of states.

Active A project that has draft items in it is considered "active". These items can be individually approved until they reach a state of "pending".

Syndicating
If "All items" or "Live and project" syndication is enabled for a library, a status of "Syndicating" is displayed on the project until all items on both the syndicator and subscriber reaches a state of pending.

Review
If any approvers are assigned to a project, the project can be submitted for review when all items in the project are in a pending state. If approved, the project then progresses to pending.

Pending
A project that contains only items in a "pending" state, or is approved, is itself considered "pending". If the project publish option is set to automatic this state is skipped. When the project is published manually or when the publish date is reached, the project moves to the "publishing" state.

Publishing
This state is the state where all items in the project move from pending to published.

Publish Failed
Indicates that one or more project items failed to publish.

Published
When all items are published, the project achieves a state of "published".

Ways to publish a project

Publishing a project refreshes the live website with your changes.

These publishing methods are used by projects:

Date When **Date** is selected, all items are published as soon as all the items in the project reach a state of "pending" and the publish date that is selected in the project is reached. You can also click **Publish** in the project form before the date is reached when all items in the project reach a state of "pending".

Manual
When **Manual** is selected, all items remain in a state of "pending" until the project is manually published by clicking **Publish** in the project form. The **Publish** function is not activated until all items in the project reach a state of "pending". Only users with editor access or higher to a project can publish a project.

Automatic
When **Automatic** is selected, all items are published as soon as all the items in the project reach a state of "pending".

When the project is published, you can continue to review the status and version history of the items in the project by opening the project form.

Note: You and other users can update the same items in multiple projects. Messages are displayed when you move an item to the publish state to warn you when this event occurs.

Project Publishing

An outline of the process of publishing a project.

- When a project publish action is initiated, all the project items are validated one by one until all items are processed.
- If any project item fails the validation, no attempt is made to publish any of the items in the project. The project state updates to "Publish Failed".
- The failed item is displayed in a table with the reason for failure. The user then modifies their content in preparation for the next validation or publish attempt.
- When all items pass the pre-publish validation, all project items are eligible to be published. The publishing process then starts, and each project item is attempted to be published one at a time.
- Any item that fails is displayed in a table with the reason for failure. The next item will then be published.
- If any items fail to validate or publish, the project state is "Publish Failed".
- If all items in the project are successfully published, the project state is "Published".

Drafts and projects

You can create, update, and approve pages in a draft state, without affecting the published site. When you work with pages, you can edit pages as drafts, manage drafts with approval workflow, and then syndicate to publish changes.

The page structure is replicated in the site structure of a web content library. Due to this integration, you can add web content directly underneath a page with the authoring portlet and use the portal page as a site area for organizing your content. Syndication then publishes the page, ensuring that changes to the page and to its associated web content are published at the same time. Update pages in a draft state first. Drafts are organized within projects in Web Content Manager. When you publish draft changes to the published site, a project coordinates the updates and ensures that all drafts are published at the same time.

The following changes inside of the scope of a project to a page managed with Web Content Manager result in a draft that is not visible on the published site and is only visible within the project:

- Any action that is run by using the toolbar, such as adding content to page, changing the page style, or changing the page layout.
- Changing the access control that is granted to a page.
- Changing the page properties.
- Creating a child page.
- Moving a page.
- Deleting a page.
- Changing the community or web content associations for a page.
- Creating or modifying portlet wires for communication between portlets on the same page or on different pages.
- Changing the portlet configuration with the **Shared Settings** mode of the portlet.

Changes to page draft elements result in either a new draft or an update to an existing draft. A page draft consists of the following elements:

- The page item itself, including title, description, metadata, and properties.
- The access control of the page.
- The page layout.
- Any public page wires that are connected to portlets on the same page or on another page.
- Any portlet preferences that are defined in **Edit Shared Settings** mode.
- The web content mappings of the current page.
- The Vanity URLs of the current page.

Scope of edits

When you edit pages in a project, you use workflow for approving changes before you publish them until they are approved. If you edit pages without selecting a project, your changes are published immediately.

When you edit pages, it is good practice to work within a project. When you are working in a project, any changes that you make to a page affect only the view within the project. After the project is approved and published, the changes are then available to all users.

Note: Changes to private pages and community pages are not part of a project. When you are working in a project, actions that personalize or create a private page are not visible.

When no project is selected, you are modifying the published site. Changes to public pages affect all users and are available immediately. While you work on the published site, changes that other users are making within active projects are not visible.

Related tasks:

"Adding workflow to managed pages" on page 1935
The default workflow that is used for published pages has only a draft and approved state. However, if the default workflow is not sufficient, you can define custom workflows and use them with published pages.

Default workflow for pages in projects: To make the page changes available on the published site, you must submit the page for review and approval. The steps for the workflow depend on what is defined for the page. By default, a simple workflow is assigned to page or URL mapping drafts. The default workflow (Express Workflow) has only two states: draft and published. For pages that use this workflow, submitting the draft moves the page to the publish pending state. When all other changes in the project are published, the pending draft page is also published. After all items in the project reach the pending state before publishing, you can publish the project by clicking **Publish** in the project menu.

Note: If you use the authoring portlet to edit the access settings for a page that is part of a workflow, you cannot modify the approvers. To change the approvers for the page, you must set the approvers on the workflow stage that currently contains the page.
However, you can change the assigned workflow in the user interface to include more review and approval stages on the project level in Web Content Manager. After the draft changes in the project are approved and published, the updated pages are visible to all users. For details on using a custom workflow, see *Adding workflow to managed pages.*

Project Validation

An outline of the validation process.

- Validating a project enables the user to check all items in a project for potential problems before the project is published. The project validation is the same validation process that is part of project publishing.
- If any item fails the project validation, they are added to a table that displays a list of items that have failed a project validation or publishing with the reason for failure.
- The user must modify their content to correct the problem before another project validation or project publishing action is initiated.
- When a validation completes, the project displays the list of items that failed the validation, the list of draft items in the project, the time that the validation was completed, and the number of items that failed.
- A validation can be initiated on a project only when it is in Pending, Publish Failed, Review, or Active State.
- A validation can be initiated by using the Authoring Portlet when a project is in read-mode.
- Only one validation per project can be initiated or running at any one time.
- A project publish action cancels any project validation for that project that is in progress because validation is part of publishing a project.

Projects and syndication

Projects are included in syndication, the method that is used by IBM Web Content Manager to replicate data from a web content library on a syndicator server to a web content library on a subscriber server. Although projects are syndicated with other items in a library that is being syndicated, you cannot use the subscriber copy of the project to update or publish your project. Work with projects on the syndicator server only.

Deferred to Syndication

When a project is syndicated, the publish method on the subscriber is automatically changed to "Deferred to Syndication", and the following actions are not enabled on the subscriber:

- CF05 Validate project
- Publish project
- Add to project
- Remove from project
- Mark for deletion
- Cancel deletion

This means that the project on the subscriber cannot be updated or published unless it receives updates from the syndicator.

When projects syndicate

Projects are not associated with a library. As syndication is library-based, a project is only syndicated when it contains unpublished items. Empty projects or previously published projects are not syndicated. Therefore, you might encounter some unexpected behavior. For example:

- If you delete a project on the syndicator and that project is empty on the subscriber, then the project deletion does not syndicate to the subscriber, because empty projects are not syndicated.
- However, if you remove all items from a project on a syndicator and then delete the project before syndication, then the project deletion is syndicated because there are still items in the subscriber version of the project.

Projects and custom workflow actions

Each workflow stage contains default actions, but you can also create custom workflow actions by creating a custom workflow plug-in.

You can assign custom workflow actions to run when a project enters a specific state in a project. For example, you might a create custom workflow action that can:

- verify web standards compliance for all assets within the project that includes items that are not traditionally workflowed, such as components and presentation templates.
- automatically reject a project if it is in review for a certain period.
- automatically delete a project when it is successfully published.

Best practices for projects

Use these tips and guidelines to develop and publish projects more effectively.

Use separate projects for different parts of the site structure

Coordinating projects is an important part of ensuring that changes to your website are published as expected. For example, you might have two authors working on the same part of the site structure but with each author editing pages in a different project. Because the authors are working in different projects, they cannot see draft changes that the other author makes until the projects are published. If both authors insert a new page between the same two published pages, the resulting page order might not be what was intended.

To prevent these situations, avoid editing the same part of the site structure with multiple projects at the same time. Instead, when multiple projects are used at the same time, ensure that you use each project for a different part of the site structure.

Coordinate changes to a project across users

When multiple users are working in the same project, one user might not be aware of the changes that another user makes. If you change the state of the project, such as deleting it or publishing it, any outstanding updates from other users are not saved. To ensure that you do not inadvertently discard changes from other users, coordinate with users of the project before the state of the project is changed.

Coordinate unique names across projects

When multiple projects are used for the same site, similar unique names might be introduced in different projects. For example, you might have two authors working on the same site but with each author editing in a different project. Because the authors are working in different projects, they cannot see draft unique names that the other author uses until the projects are published. If both authors insert a similar unique name, the result might not be what was intended.

To prevent these situations, avoid introducing new unique names with multiple projects at the same time. Instead, when using multiple projects at the same time, ensure that you use different unique names within each project.

Workflow and change management

You can manage changes to web content items either by creating drafts, by using workflows, or adding items to projects.

"Item status"
There are three major status levels that an item can be in at any one time; **Draft**, **Published**, or **Expired**. The current state of an item indicates where the item exists within a change management process, and where that item can be viewed and accessed.

"Workflow stages and actions" on page 1930
You use workflows to control the access to, verification and eventual approval of items. Only if an item is approved at all stages up to a published stage can it be viewed on your website.

"Adding workflow to managed pages" on page 1935
The default workflow that is used for published pages has only a draft and approved state. However, if the default workflow is not sufficient, you can define custom workflows and use them with published pages.

Item status

There are three major status levels that an item can be in at any one time; **Draft**, **Published**, or **Expired**. The current state of an item indicates where the item exists within a change management process, and where that item can be viewed and accessed.

Status types

Draft This indicates that the item is being updated.

Pending published
An item that uses a workflow can appear in a state of pending published. This indicates that the item has entered a workflow stage that includes a publish action but the action has yet to be processed.

Published
A published status indicates that an item is ready to be rendered in the live site.

Pending expired
An item that uses a workflow can appear in a state of pending expired. This indicates that the item has entered a workflow stage that includes an expire action but the action has yet to be processed.

Expired
An expired status indicates that an item is ready to be expired from the live site.

Changing status

The status of an item can change only in a linear fashion:

- **Draft** to **Published**.
- **Published** to **Expired**.
- **Expired** to **Published**.
- **Published** to **Draft**.

You cannot change an item's status from **Expired** to **Draft**.

The process of publishing and expiring items

When the status of an item changes to published or expired, this change does not mean that the item has been added or removed from the rendered site. A status of published or expired means that the process of publishing or expiring an item has begun.

The actual time a published item appears on a website, or the time an expired item is removed from a website, also depends on:

- how long it takes to syndicate updates to the delivery server
- how long it takes for the current cache to expire

Workflow stages and actions

You use workflows to control the access to, verification and eventual approval of items. Only if an item is approved at all stages up to a published stage can it be viewed on your website.

A workflow must have at least one stage, but typically has more, and it always flows in a linear pattern. You can use a workflow to:

- Review the accuracy of content.
- Review content for any legal implications.
- Review content to ensure that it meets accessibility guidelines.
- Ensure that no malicious code such as cross scripting attacks are added to content.

A reject stage can be specified, which is a stage that is run when a document is declined, before the document is moved it to the first stage of the workflow. If the item is rejected at any stage, someone with editor access needs to correct or amend the item and resubmit it into the selected workflow (for approval). All items that are rejected (regardless of the stage they are at in the approval process) are sent back to the first (creation) stage of the workflow.

You can also specify that a comment must be entered on every move a document makes in the workflow or only on specific stages. This comment is added to the document's history section.

Workflow stages:

Workflow stages are the building blocks of a workflow. You need to create at least one stage before you can create a workflow.

Workflow stages

Stages determine:

- What actions to run when an item enters or exits a workflow stage
- The access levels of users or groups within that stage.

In most cases, actions are run when an item enters a stage. For example, you add a scheduled move action to run on entering a stage so that it is enabled as soon as an item enters that stage. However, if you set a scheduled move action to run on leaving a stage, it is never run. The most common type of actions to run on leaving a stage are email actions, when you want to notify users that an item has exited a workflow stage.

Note: Some actions need to be run in a specific order. For example:

- A scheduled move action must always be the final action in a workflow stage, because any actions scheduled after a scheduled move action is not run.
- You cannot run a version action before a publish action because you cannot save versions of draft items.
- If you use a custom action, you can run the custom action before the email action is run, so that the draft content item is in a state ready to be reviewed by an approver.

Note: The access settings that are defined in the properties section of the workflow stage form are the security settings that are applied to items during a workflow, not the Security section of a workflow stage. The Security section defines who has access to the workflow stage item itself.

Reject stages

In addition to the workflow stages that make up a workflow, workflow stages are also used as part of rejecting an item. When an item is rejected, a reject stage can be triggered that runs pre-defined actions. When the actions are run, the item is returned to the first stage of the workflow.

Workflow actions:

Each workflow stage contains sets of actions; actions that are run when an item enters the stage, and actions run when an item exits the stage. The exit actions are restricted to non-scheduled actions, since they must be run immediately.

The following table describes the actions that you can choose for workflow stages.

Table 278. Workflow stage actions

Action	Details
Publish	Changes an item's Status from **Draft** to **Published**. The item is available on the rendered site. An item is published when it enters a workflow stage that contains a publish action, and when the selected published date and time are reached.
Expire	Changes an item's Status from **Published** to **Expired**. The document is no longer available on the site. An item is expired when it enters a workflow stage that contains an expire action, and when the selected expire date and time are reached.
Email	This action sends emails when run. You can create new email actions and specify who the recipients are. You can select to email approvers, authors, and owners. You can also create a list of other users or groups to email. A link to the Item to be reviewed is included in the email.
Scheduled Move	Runs a scheduled move to the next stage on a specified date. A list-box allows users to select one of four date types that are entered on each individual document, or you can specify a static date.
Version	This action creates a version of an item when run.
Custom	You can also create custom workflow actions by creating a custom workflow plug-in. These actions can be used and scheduled within a workflow like other workflow actions.

Access to items during a workflow:

If an item is participating in a workflow, the creator is given manager access to the item only in the first workflow stage. As the item progresses through a workflow, the item access is determined by the combined workflow and system defined access levels.

Table 279. Security matrix

Security level	First workflow stage	Other workflow stages
User **Contributor** **Editor** **Manager** **Reviewer** **Draft Creator**	• Administrator defined • Workflow defined	• Administrator defined • Workflow defined
Administrator	If you are assigned the administrator role to a library, you automatically inherit all administration access down to the item-level. It cannot be turned off.	If you are assigned the administrator role to a library, you automatically inherit all administration access down to the item-level. It cannot be turned off.

Joint approval:

Joint approval is used in cases where approval from multiple users is required before an item is moved to the next stage.

- You specify which stages you want to be jointly approved.
- If joint approval is active, then all the approvers that are specified for this stage must approve the document.
- The exception is an administrator, who can force a document to the next stage.
- If a group is specified as an approver, only one user per group is required to approve the document.
- If that user is a member of more than one Group that has Approve access, then all the groups are considered to have approved the document.

Workflow example:

This example describes the steps that are required to create a four stage Workflow.

Actions

The actions that are required for this workflow are:

Table 280. Actions

Action	Description
Publish	Makes a document visible as a published document on the website.
Scheduled Move	This scheduled move action detects when a document passes its expire date and moves the document into the Expired stage.
Expire	Stops a document from being visible on the website.
Email	Sends emails to selected users. In this example, you select "Email Stage Approvers".

Stages

The following stages make up the Workflow:

Table 281. Stages

Workflow Stage	Actions on Entering	Description
Stage 1, Draft		All content begins the Workflow in this Stage. All content authors are authorized to use the Draft stage.
Stage 2, Review	**Email**	Content requires a review stage before it is published. A small group of people are authorized to approve content in this stage and move it to the Publish stage. The email action sends an email to each approver.
Stage 3, Publish	**Publish** **Scheduled Move**	This is the stage where content from the workflow is published. The **Publish** action is triggered when the **Publish Date** is reached. At this point, the content is visible on the website. The **Scheduled Move** action is triggered when the **Expire Date** is reached.
Stage 4, Expired	**Expire**	The **Scheduled Move** action moves the content into this stage, where the **Expire** action stops the document from being visible on the website.

Progressing through the workflow

Table 282. Progressing through the workflow

Process	Stage	Status
An item is first created and saved:	**Stage 1, Draft**	**Draft**
When the Item Creator is ready, they move it to the next stage.	**Stage 2, Review**	**Draft**

Table 282. Progressing through the workflow (continued)

Process	Stage	Status
An Approver would then access the document and review it. If Approved, the item would be moved to the next stage. The Status is **Draft** pending **Published**. This is because the time specified in the **Publish Date** field of the item has not yet been reached.	**Stage 3, Publish**	**Draft** pending **Published**
Once the **Publish Date** is reached, the item's Status changes to **Published**. The item is still in **Stage 3, Publish**, but the Status has changed.	**Stage 3, Publish**	**Published**
Once the **Expire Date** is reached, the document is moved to the final stage. As both the **Scheduled Move** and **Expire** actions are using the **Expired** date in this example, the item's Status immediately changes to **Expired**. If the **Scheduled Moved** Action used a **General Date** prior to the **Expire Date**, then the item's Status would remain as **Published** pending **Expire** until the **Expire Date** was reached even though it had entered **Stage 4, Expire**.	**Stage 4, Expire**	**Expired**

Using workflows and access to workflow items:

Users do not require access to a workflow's actions or stages to participate in a workflow. Actions are performed using system access and are not determined by the access of the user who approved/rejected the item.

Adding workflow to managed pages

The default workflow that is used for published pages has only a draft and approved state. However, if the default workflow is not sufficient, you can define custom workflows and use them with published pages.

Before you begin

Before you do this task, ensure that you add the **Portal Site** library to the list of libraries that you can edit in the **Web Content Authoring** tab. In the **Web Content**

Authoring tab, click **Preferences**, and then click **Configure** or **Edit Shared Settings**.

About this task

To use a custom workflow with a published page, you must create a page template and specify the custom workflow on the template. When you create a page that is based on the template, the custom workflow is used for the page.

Important: Users with Privileged User access can personalize a page by adding content or by making other customizations to the page. Because Privileged User access is inherited by pages by default, your custom workflow can be bypassed by users with this access. If you want to ensure that users can update the page only through the workflow, disable the inheritance of Privileged User access for the page.

Procedure

1. In the **Web Content Authoring** tab, create any workflow stages that you require and then create the custom workflow.
2. Create a project to use for creating the page template.

   Note: Because you can set a workflow only on draft items, you must create the template as a draft in the context of a project. After you add your custom workflow to the draft, publish the draft to make the page template available.
3. In the **Web Content Authoring** tab, select **Project Views** > **All Projects** and select the new project as the current context.
4. Click the **Administration menu** icon . Then, click **Portal User Interface** > **Page Templates**.
5. Select **Create** from the site toolbar and create the page template.
6. Add the workflow to the template.
   a. In the site toolbar, select **Menu** > **Edit Page Properties** to display the Manage Page Properties window.
   b. Select the **Security** tab.
   c. In the **Workflow** section, click **Select** in the **Workflow** field.
   d. Select the custom workflow, and click **OK**.
   e. Exit out of the Manage Page Properties window.
7. Approve and publish the project to make the page template available for use.
   a. In the **Projects** menu of the site toolbar, select **Manage Project**.
   b. Select the page template in the list of project items, and click **More** > **Approve**.
   c. Click **Publish Project**.

What to do next

After you complete this task, you can select the new page template when you create a page in a project, and the custom workflow is automatically used.

IBM Web Content Integrator

The Web Content Integrator is a solution for integrating externally managed web content with WebSphere Portal Express. By using standard content syndication feed technologies based on RSS 2.0, the Web Content Integrator provides a loosely coupled mechanism for transferring published content and metadata to the portal

after they are approved in the source system. When the content and metadata are transferred to the portal, it is possible to use the built-in content management features of Web Content Manager to secure, personalize, and display the content to users.

To use the Web Content Integrator you must:

1. Create a feed on your source system by using the Web Content Integrator feed format specification.
2. Configure WebSphere Portal to consume the feed.

Feed format specification

RSS, or Really Simple Syndication, is an XML-based format that is widely used for syndicating content from sources such as websites and blogs to readers who are subscribed to the feeds. The input to the Web Content Integrator is a content feed that complies with the RSS 2.0 format. The core feed format is relatively simple, with only a limited number of elements that need to be specified for each item in the feed. However, the RSS 2.0 specification allows the base format to be extended by using XML namespaces to support extra functions. To enable a deeper level of control over how items are created in Web Content Manager, an RSS extension is defined which contains elements that map to many of the attributes of the Web Content Manager object model.

"Feed format overview"
RSS 2.0 is a dialect of XML, and all RSS files must comply with the XML 1.0 specification as published by the World Wide Web Consortium (W3C). RSS feed files typically have file extensions of either .rss or .xml. The Web Content Integrator does not impose any file naming conventions on the feed producer.

"RSS Namespace Extension for web content" on page 1940
Web Content Manager items contain a set of controls that are used to store information for various purposes. The elements in this namespace roughly map to the fields that are available on those controls. Depending on the type of the item, it can or can not contain certain controls so some of the elements in the namespace are only relevant to specific item types. All of the elements in this namespace are sub-elements of `<item>`, none are used at the `<channel>` level.

"How to handle embedded links" on page 1971
From time to time, the content that is being retrieved by the Web Content Integrator contains embedded links to images, files, and other content within the feed. You can use the link tag in your feed to represent embedded links so that they are converted into links to other Web Content Manager items that are created when the feed is processed.

"Processing images" on page 1976
The Web Content Integrator has an image-processing feature, which allows images that are referenced within HTML and rich text elements to be created as image components while processing a feed.

"Example feeds" on page 1979
View some example feeds that illustrate how to represent the various item types within a feed.

"RSS Namespace Extension for the Feed Service" on page 1982
The RSS namespace extension is used to exchange control information between the feed producer and consumer applications.

Feed format overview

RSS 2.0 is a dialect of XML, and all RSS files must comply with the XML 1.0 specification as published by the World Wide Web Consortium (W3C). RSS feed

files typically have file extensions of either .rss or .xml. The Web Content Integrator does not impose any file naming conventions on the feed producer.

RSS 2.0 feed file format

The most commonly used media type options are "text/xml" and "application/rss+xml". The choice of media type does impact how the Web Content Integrator is able to determine the character encoding of the feed. If the character encoding cannot be determined correctly, the Web Content Integrator will produce errors when parsing the feed. Therefore it is important to choose an appropriate media type for your environment.

Following the XML prolog, an RSS 2.0 file begins with a single <rss> element. This element has one required attribute, "version", which must be set to "2.0". The file must also contain a single <channel> element which contains a number of sub-elements that provide some metadata about the feed as a whole. The <channel> element must contain one or more <item> elements. The <item> elements in turn contain sub-elements which provide information about the content that is being syndicated. For example:

```
<?xml version="1.0"?>
<rss version="2.0">
 <channel>
  <title>Sample Feed</title>
  <link>http://www.ibm.com/feeds/sample.rss</link>
  <description>An example RSS Feed</description>
  <lastBuildDate>Tue, 31 Oct 2006 10:31:00 EST</lastBuildDate>
  <item>
   <title>News Item Two</title>
   <link>http://www.ibm.com/news/two.htm</link>
   <description>
   This is a summary of the second news article
   </description>
   <pubDate> Tue, 31 Oct 2006 10:31:00 EST</pubDate>
  </item>
  <item>
   <title>News Item One</title>
   <link>http://www.ibm.com/news/one.htm</link>
   <description>
   This is a summary of the first news article.
   </description>
   <pubDate> Tue, 31 Oct 2006 10:30:00 EST</pubDate>
  </item>
  </channel>
 </rss>
```

Note: If non-ascii data is used in a feed, then encoding="UTF-8" must be specified in the feed: `<?xml version="1.0" encoding="UTF-8"?>`

Channel-level Elements

Each RSS feed file must contain only one channel element. There are a number of allowable sub-elements of the channel that provide some metadata about the channel itself. The following elements are either required or used by the Web Content Integrator.

`title` This element is used to provide the name of the feed. This element is required by the RSS 2.0 specification but is not used by the Web Content Integrator.

link This element contains a URL that points to the Web page containing the feed. This element is required by the RSS 2.0 specification but is not used by the Web Content Integrator.

description
This element contains a brief description of the content of the channel. This element is required by the RSS 2.0 specification but is not used by the Web Content Integrator.

lastBuildDate
This element contains a date and time stamp representing the last time the content of the feed was changed. This date, as well as any others in the feed, must conform to the RFC 822 format. This is an optional element according to the RSS 2.0 specification, however some feed reader applications may depend on it. In certain cases the Web Content Integrator will store the value in the lastBuildDate element and then pass it back to the feed producer on the next request as a way of indicating which version of the feed it has already syndicated.

Item-level Elements

For the purposes of the Web Content Integrator, each item in the feed represents an item type. The following item types can be created or updated via the feed:

- Content items
- Site Areas
- Taxonomies
- Categories
- Component

The following sub-elements are either required or used by the Web Content Integrator:

title The value of this element is stored in the Name field of Web content items. For content items this becomes part of the URL to the content page. As this is used in the Name field of Web content items, the title can contain only alphanumeric characters (a-z, A-Z, 0-9), spaces, and the following characters: $ - _ . ! ( ) , This is a required sub-element.

link This is the URL to the source content. In some cases it will be used as the base URL from which any relative links embedded in the content are resolved.

description
The value of this element will be stored in the Description field of Web content items. Although the RSS specification allows entity-encoded or escaped HTML to be placed in this element, the Description field in Web content items is not designed to store HTML. For the purposes of the Web Content Integrator this element must only contain plain text.

pubDate
The value of this element must be an RFC 822 time and date stamp representing the time that the item was added to, or updated in, the feed. The Web Content Integrator will use this date in combination with the <guid> element to determine whether or not it has already processed the item. Each time an item is updated via the feed the value of the <pubDate> in the feed entry will be updated as well to indicate that something has changed. This is a required sub-element.

guid The <guid> element must contain an ID to uniquely identify the item. This will often be a unique ID from the source content management system. The Web Content Integrator will maintain a mapping of this ID to the item's internal Web Content Manager ID. This is necessary in order to be able to correctly update or delete items that already exist in Web Content Manager. This field is case sensitive and can contain any string of characters up to a maximum 256 characters in length. The isPermaLink attribute will be ignored. This is a required element.

category
Each <category> element will contain a hierarchical meta data tag that describes the content. The value of this element will be translated into taxonomy and category items in Web Content Manager. If the category tree specified in the <category> element does not already exist in Web Content Manager it will be created automatically by the Web Content Integrator when the feed entry is processed. The RSS 2.0 specification defines an optional domain attribute for the category element. Feed producers can use this attribute to store the name of the Web content library where the category tree is to be created. This element only applies to content items. A single <item> may contain multiple category elements. As this will be used in the Name field of Web content taxonomy and category items, the title can contain only alphanumeric characters (a-z, A-Z, 0-9), spaces, and the following characters: $ - _ . ! ( ) , This is a required sub-element.

author According to the RSS 2.0 specification this element contains the author's e-mail address. The specification only allows a single <author> element per item. Generally this will be the author of the content item in the source content management system. The Web Content Integrator will attempt to resolve the e-mail address into the common name of a portal user and then store the name of that user in the author field the Web Content Manager item. If this element is not present in the feed, or if the e-mail address cannot be resolved, then the name of the system user will stored in the author field of the Web Content Manager item instead.

RSS Namespace Extension for web content

Web Content Manager items contain a set of controls that are used to store information for various purposes. The elements in this namespace roughly map to the fields that are available on those controls. Depending on the type of the item, it can or can not contain certain controls so some of the elements in the namespace are only relevant to specific item types. All of the elements in this namespace are sub-elements of `<item>`, none are used at the `<channel>` level.

"Adding the custom namespace definition to the feed" on page 1941
The first requirement is to add the namespace reference to the `<rss>` element at the beginning of the feed.

"Process Control Elements" on page 1942
These elements are used to provide the Web Content Integrator with some information about how to handle the data that is contained within the `<item>`.

"Location Control Elements" on page 1943
These elements are used to provide the Web Content Integrator with some information about an item's relative location.

"Identity control elements" on page 1948
Most identification fields in Web Content Manager items map directly to core RSS elements; `<title>` maps to the name field, `<description>` maps to the description field, and `<author>` maps to the authors field. Other identification fields can be populated by using the following identity control elements.

"Profile Control Elements" on page 1949
Taxonomies and category fields are populated by using the RSS <category> element. To populate the **Keywords** field, a keywords element is required.

"The authoringTemplate element" on page 1950
The authoringTemplate element is used to specify the authoring template that is applied to a content item or site area. The value of this element must be the name of an existing authoring template.

"Element control element" on page 1951
The element control element is used to populate elements that are stored in components, site areas, and content items.

"Workflow control elements" on page 1966
Workflow elements is used to specify workflow-related parameters when content items are created that use a workflow.

"Security control elements" on page 1969
The security control element is used to set access controls on the item that is being created.

"Association control element" on page 1970
The association element is used to create an association between two or more separate content items. It is used to support some content item relationships that are not native to Web Content Manager but that are found in some other web content management systems. One example would be a technical journal article that needs to include some data from one or more author biography content items.

Adding the custom namespace definition to the feed:

The first requirement is to add the namespace reference to the <rss> element at the beginning of the feed.

About this task

The URL of the namespace is specified by using the following tag:

```
<rss version="2.0" xmlns:ibmwcm="http://purl.org/net/ibmfeedsvc/wcm/1.0" >
```

This is a fixed property value that the feed parser uses as the key to identify elements that belong to the custom Web Content Integrator namespace.

Example

For example:

```
<?xml version="1.0"?>
 <rss version="2.0" xmlns:ibmwcm="http://purl.org/net/ibmfeedsvc/wcm/1.0" >
 <channel>
  <title>Sample Feed</title>
  <link>http://www.ibm.com/feeds/sample.rss</link>
  <description>An example RSS Feed</description>
  <lastBuildDate>Tue, 31 Oct 2006 10:31:00 EST</lastBuildDate>
  <item>
   <title>News Item Two</title>
   <link>http://www.ibm.com/news/two.htm</link>
   <description>
   This is a summary of the second news article
   </description>
   <pubDate> Tue, 31 Oct 2006 10:31:00 EST</pubDate>
```

```
   <ibmwcm:itemType>Content</ibmwcm:itemType>
  </item>
 </channel>
</rss>
```

The namespace label "**ibmwcm**" can be changed if the label specified in the namespace declaration matches the label that is used on the extended elements in the feed.

Process Control Elements:

These elements are used to provide the Web Content Integrator with some information about how to handle the data that is contained within the <item>.

action

This element indicates the action to be run on the item that is represented by the <item>.

Table 283. action element

Element parameters:	Details for this element:
Applies to item types	All
Required for item types	All
Allowable Values	"add", "update", or "delete".
Required Attributes	None
Optional Attributes	None
Required sub-elements	None
Optional sub-elements	None

Example:

```
<ibmwcm:action>update</ibmwcm:action>
```

itemType

The itemType element indicates the type of item that is represented by the feed entry. These correspond to the Web Content Manager item types that can be created or updated via the feed. In some cases, the value of this element is combined with the value in the <ibmwcm:path> element to determine which Web Content Manager item type to create.

Table 284. itemType element

Element parameters:	Details for this element:
Applies to item types	All
Required for item types	All
Allowable Values	"siteArea" for site areas. "content" for content items. "component" for components. "category" for taxonomies and categories.
Required Attributes	None

Table 284. itemType element (continued)

Element parameters:	Details for this element:
Optional Attributes	None
Required sub-elements	None
Optional sub-elements	None

Example:

Site Area

```
<ibmwcm:itemType>siteArea</ibmwcm:itemType> + any other value for <ibmwcm:path>
or the lack of an <ibmwcm:path> element.
```

Taxonomy

```
<ibmwcm:itemType>category</ibmwcm:itemType> + <ibmwcm:path>/</ibmwcm:path>
```

Category

```
<ibmwcm:itemType>category</ibmwcm:itemType> + any other value for <ibmwcm:path>
or the lack of an <ibmwcm:path> element.
```

Component

```
<ibmwcm:itemType>component</ibmwcm:itemType>
```

Content

```
<ibmwcm:itemType>content</ibmwcm:itemType>
```

Location Control Elements:

These elements are used to provide the Web Content Integrator with some information about an item's relative location.

`library`

You must specify a default web content library name when configuring a task to consume an input feed. The Web Content Integrator performs all operations within that library including creating new Web Content Manager items, searching for existing versions of items, and searching for any associated design artifacts such as authoring templates and workflows. You can also override this setting so that a single feed can be used to create content in multiple libraries.

There are three ways that Web Content Integrator will check if a value was specified in this element:

- If the library specified in the library element matches an existing web content library, all operations during the processing of this feed entry is processed within the context of the library specified in the library element.
- If no library element is present in the feed entry, or if the library element is present but no value was specified then the default library from the task configuration is used.
- If a value is specified in the library element but no matching web content library is found, then an error message is logged and the feed entry will not be processed.

Table 285. `library` element

Element parameters:	Details for this element:
Applies to item types	All
Required for item types	None

Table 285. **library** *element (continued)*

Element parameters:	Details for this element:
Allowable Values	The name of an existing web content library.
Required Attributes	None
Optional Attributes	None
Required sub-elements	None
Optional sub-elements	None

Example:

```
<ibmwcm:library>LibraryName</ibmwcm:library>
```

path

The **path** element is used to indicate the hierarchical path to the web content item. For site areas and content items, this element contains the parent site area path. For categories, the value is the parent category path.

Site areas and categories can only have a single path element. If more than one path element is specified, only the first is used. Content items can have multiple parents, so multiple path elements can be used. The first path element is used as the main content item and the following path elements are treated as content links. An optional "library" parameter can be specified for path elements referring to linked content in a different library from the main content item.

New items are added to the end of the current list of items within Web Content Manager.

The path element cannot be used with components. Components can only be created under the "Components" preset folder.

Table 286. **path** *element*

Element parameters:	Details for this element:
Applies to item types	siteArea, Category, Content
Required for item types	Site Areas and Taxonomies.
Allowable Values	All values should start with a leading forward slash "/" character. For site area and taxonomy items the value should be just a single forward slash "/".
Required Attributes	None
Optional Attributes	"library" Contains the name of a library where the site path resides. This attribute is used where linked content resides on a different library from the main content. This attribute is ignored on the first path element that is listed in the feed entry.
Required sub-elements	None
Optional sub-elements	None

Examples:

```
<ibmwcm:path>/</ibmwcm:path>

<ibmwcm:path>/IBM/Products</ibmwcm:path>

<ibmwcm:path library="en_US">/Intranet/Home/News</ibmwcm:path>
```

createLinks

This is used to specify which parent items to link to when creating content items.

Table 287. **createLinks** *element*

Element parameters:	Details for this element:
Applies to item types	Content
Required for item types	None
Allowable Values	The **createLinks** element is a container for readability and has no attributes or expected values.
Required Attributes	None
Optional Attributes	None
Required sub-elements	**parentGuid** The **parentGuid** element should contain the unique ID of another item in the feed that describes the parent item.
Optional sub-elements	None

Examples:

```
<ibmwcm:createLinks>
 <ibmwcm:parentGuid>8234ad23fb29</ibmwcm:parentGuid>
</ibmwcm:createLinks>
```

The Orphan Container

If no valid parents or paths are specified in the feed, then the item is placed into an orphan container until it is able to be updated with a valid path. The Web Content Integrator will automatically create an orphans container in each Web content library as it is needed. For site areas and content items, the orphan container is a site area path name `WCI/Orphans`. For categories, the orphan container is a taxonomy and category path with the name `WCI/Orphans`. The name of the orphan containers can be controlled through a setting in the `WCMConsumerPlugin.properties` file.

children

The **children** element is used to specify the children of the current item. The value of this element should be the GUID of another entry in the feed that describes a child of the current item.

The referenced child item must be of an appropriate type. For site areas, the children must be site areas or content items. For taxonomies and categories, the children must be categories. If the type is not valid, the parent item will still be added or updated, but the child reference will not be created.

Multiple childGuid sub-elements can be contained within the children element. If multiple children are specified, they are added in the order in which they are listed in the feed. This allows the feed producer to control the order in which site areas and content items are linked to their parent site area which can be useful for setting up navigation displays.

There are two attributes of the children element that control how the children specified in the feed get combined with any children that may already be contained by the parent item. The action attribute controls whether the child list in the feed replaces the existing list of children. If the value of that attribute is set to "add", then the children that are specified in the feed is combined with the existing list of children. Any other value, including an empty string or the absence of this attribute, causes the list of children that are specified in the feed to completely replace the pre-existing list.

The position attribute is only relevant when the action attribute is set to "add". This attribute controls whether the children specified in the feed are added to the start or to the end of the pre-existing child list. If this attribute is not specified, the children will be added to the end of the pre-existing list. If the children element is not present in the feed, then the pre-existing child list remains intact.

Site areas and categories can only have one immediate parent. Any pre-existing parent relationships will be removed before adding them as a child of this item. As content items can have multiple parents, adding a content item as a child of this item does not remove it from any of its other parents

Table 288. **`children`** *element*

Element parameters:	**Details for this element:**
Applies to item types	siteArea, Category.
Required for item types	None
Allowable Values	The children element is just a container for readability. It has no expected values.
Required Attributes	None
Optional Attributes	**`action`** The action attribute can have a value of "add" or "replace". These values indicate whether to replace children that are existing or to add to them. **`position`** This attribute can have the value "start" or "end". These values indicate where the children are placed in sibling order as they are processed.
Required sub-elements	**`childGuid`** The childGuid element should contain the unique ID of another item in the feed that describes the child item.
Optional sub-elements	None

Examples:

```
<ibmwcm:children action="add" position="start">
 <ibmwcm:childGuid>8234cb51df43</ibmwcm:childGuid>
</ibmwcm:children>
```

defaultContent

The defaultContent element only applies to site areas. It allows the feed producer to specify which content item is used as the default content for a site area. If this element is missing, has a blank value, or cannot be resolved, then the default content of the site area is cleared.

Table 289. **defaultContent** *element*

Element parameters:	Details for this element:
Applies to item types	siteArea.
Required for item types	None
Allowable Values	Value should be the GUID of another item in the feed that describes a content item.
Required Attributes	None
Optional Attributes	None
Required sub-elements	None
Optional sub-elements	None

Examples:

```
<ibmwcm:defaultContent>8234cb5</ibmwcm:defaultContent>
```

templateMap

The **templateMap** element only applies to site areas. It allows the feed producer to specify the template maps to be used when rendering content that is contained within the designated site area. Use multiple instances of the **templateMap** element to create multiple template mappings on a site area.

When performing an update of an existing site area, the following steps are executed for each `<ibmwcm:templateMap />` element that is specified in the feed:

1. Get the names of the authoring and presentation templates specified in the **templateMap** element.
2. Attempt to locate an authoring template that matches the name that is specified in the feed.
3. If a matching authoring template item cannot be located, log an error and start processing the next **templateMap** element.
4. Attempt to locate a presentation template that matches the name that is specified in the feed.
5. If a matching presentation template cannot be located, log an error and start processing the next **templateMap** element.
6. Check if the site area already contains a mapping for the specified authoring template:
   a. If so, check whether it maps to the same presentation template as specified in the feed;
      1) If so, go to the next **templateMap** element.
      2) If not, remove the mapping and replace it with one to the specified presentation template.

b. If not, create a new mapping of the specified authoring template to the specified presentation template.

7. Process the next **templateMap** element

The WCM APIs don't provide a mechanism to get a complete list of template mappings that exist for a given site area. As a result, it is not possible to remove a template mapping via the feed. Once a mapping has been set on a site area, it can be updated via the feed but can only be removed manually through the WCM authoring portlet.

Table 290. **templateMap** *element*

Element parameters:	**Details for this element:**
Applies to item types	siteArea.
Required for item types	None
Allowable Values	This element does not take any values. The data is specified in the attributes.
Required Attributes	**authoring** The value of this attribute must be a string that exactly matches the name of an existing authoring template. **presentation** The value of this attribute must be a string that exactly matches the name of an existing presentation template.
Optional Attributes	None
Required sub-elements	None
Optional sub-elements	None

Examples:

```
<ibmwcm:templateMap authoring="AT News" presentation="PT News" />
<ibmwcm:templateMap authoring="AT Announcement" presentation="PT Announcement" />
```

Identity control elements:

Most identification fields in Web Content Manager items map directly to core RSS elements; `<title>` maps to the name field, `<description>` maps to the description field, and `<author>` maps to the authors field. Other identification fields can be populated by using the following identity control elements.

displayTitle

The displayTitle element allows the feed producer to specify a separate value for the **Display Title** field in Web Content Manager. If this element is not present in the feed entry, Web Content Manager sets the value of the **Display Title** field to match the **Name** field when the item is saved.

Table 291. displayTitle element

Element parameters:	**Details for this element:**
Applies to item types	All
Required for item types	None

Table 291. displayTitle element (continued)

Element parameters:	Details for this element:
Allowable Values	A string to be used as the **Display Title** in Web Content Manager. Unlike the **Name** field, this field can contain double-byte and non-ASCII characters.
Required Attributes	None
Optional Attributes	None
Required sub-elements	None
Optional sub-elements	None

Example:

```
<ibmwcm:displayTitle>
 Fourth Quarter Results Press Release
</ibmwcm:displayTitle>
```

owner

The owner element provides a mechanism to set the value of the **Owners** field by using the feed. Multiple owner elements can be specified. Each owner element must contain the common name of a single WebSphere Portal user or group. If the Web Content Integrator is unable to resolve a specified name to an actual user or group then that user or group is not added to the **Owners** field but all other processing continues as normal.

Table 292. owner element

Element parameters:	Details for this element:
Applies to item types	All
Required for item types	None
Allowable Values	[all users] [all authenticated portal users] The common name of any valid WebSphere Portal user or group
Required Attributes	None
Optional Attributes	None
Required sub-elements	None
Optional sub-elements	None

Example:

```
<ibmwcm:owner>[all authenticated portal users]</ibmwcm:owner>
<ibmwcm:owner>jsmith</ibmwcm:owner>
```

Profile Control Elements:

Taxonomies and category fields are populated by using the RSS `<category>` element. To populate the **Keywords** field, a keywords element is required.

keywords

The keywords element must contain a comma-delimited list of metadata tags that describe the content. The list is stored in the **Keywords** field of content items.

Table 293. keywords element

Element parameters:	Details for this element:
Applies to item types	Content
Required for item types	None
Allowable Values	A comma-delimited list of keywords in plain text format.
Required Attributes	None
Optional Attributes	None
Required sub-elements	None
Optional sub-elements	None

Example:

```
<ibmwcm:keywords>software,testing</ibmwcm:keywords>
```

The authoringTemplate element:

The authoringTemplate element is used to specify the authoring template that is applied to a content item or site area. The value of this element must be the name of an existing authoring template.

authoringTemplate

If the authoring template is in a different library from the item, you can specify the library name as well. If no library is specified, the Web Content Integrator searches the library that was specified in the `<ibmwcm:library>` element. If no `<ibmwcm:library>` element is present, the Web Content Integrator searches the default library that is specified in the task configuration.

Note: When an item is created, it is not possible to change its authoring template. When an update is processed, if the authoring template specified in the feed entry does not match the name of the authoring template that was originally used to create the item the Web Content Integrator generates an error message.

Table 294. authoringTemplate element

Element parameters:	Details for this element:
Applies to item types	Content, site area
Required for item types	None
Allowable Values	The name of an existing authoring template, or the name of an existing library and authoring template that is separated by a forward slash. For example: • News-AT • TmpltLib/News-AT
Required Attributes	None
Optional Attributes	None

Table 294. authoringTemplate element (continued)

Element parameters:	Details for this element:
Required sub-elements	None
Optional sub-elements	None

Example:

```
<ibmwcm:authoringTemplate>News</ibmwcm:authoringTemplate>
<ibmwcm:authoringTemplate>TmpltLib/News</ibmwcm:authoringTemplate>
```

Element control element:

The element control element is used to populate elements that are stored in components, site areas, and content items.

Element overview

The Web Content Integrator uses the following business logic to process each component element in the feed entry:

1. It checks whether the component site area or content item contains a field whose name matches the value in the feed element's name attribute. This is a case-sensitive comparison so the names must match exactly.
2. If a matching field is found, it checks whether the data type matches what was specified in the feed element's sub-element.
3. If both the name and data type match, the element in the site area or content item is updated to match the data contained the feed element.
4. If an element is found which matches the feed element's name, but not its data type, the Web Content Integrator removes the old field from the site area or content item and attempt to replace it with a new field that matches the data type specified in the feed.
5. If no matching field can be found on the content, the Web Content Integrator attempts to create a new element.

Note: When creating elements in Web Content Manager using an authoring portlet, it is possible to specify a number of field validation criteria such as the maximum size of a text field or the allowable range of a date field. These are validated during the save operation. If the data in the field does not meet the validation criteria the entire save operation fails, meaning that none of the changes in the feed entry is applied. The Web Content Integrator does not have the ability to check that the data in the feed element is valid before attempting the save operation. Therefore, if you elect to implement validations on your authoring templates, it is important for the feed producer to ensure that the content is valid during the generation of the feed.

Text element

Table 295. text element

Element parameters:	Details for this element:
Applies to item types	Site areas and content items
Required for item types	None.
Allowable Values	A text component must contain a plain text value.

Table 295. text element (continued)

Element parameters:	Details for this element:
Required Attributes	**name** The value of this attribute must match the name of an existing text element in a site area or content item.
Optional Attributes	None
Required sub-elements	**type** The value of this sub-element must be "text". This value is not case-sensitive. **value** The value of this sub-element must be plain text.
Optional sub-elements	None

Short Text element

Table 296. short text element

Element parameters:	Details for this element:
Applies to item types	Site areas and content items
Required for item types	None.
Allowable Values	A short text component must contain a plain text value.
Required Attributes	**name** The value of this attribute must match the name of an existing short text element in a site area or content item.
Optional Attributes	None
Required sub-elements	**type** The value of this sub-element must be "shorttext". This value is not case-sensitive. **value** The value of this sub-element must be plain text.
Optional sub-elements	None

Example:

```
<ibmwcm:element name="Headline">
 <ibmwcm:type>shorttext</ibmwcm:type>
 <ibmwcm:value>New Product Released</ibmwcm:value>
</ibmwcm:element>
```

HTML element

Table 297. html element

Element parameters:	Details for this element:
Applies to item types	HTML components, site areas, and content items
Required for item types	HTML components.

Table 297. html element (continued)

Element parameters:	**Details for this element:**
Allowable Values	The value of the value sub-element must be HTML that is stored in the corresponding HTML element. The HTML can either be entity-encoded or contained within a CDATA element. Alternatively, the feed producer can provide a URL to an HTML file whose contents are retrieved by the Web Content Integrator
Required Attributes	**name** The value of this attribute must match the name of an existing HTML element in a site area or content item.
Optional Attributes	None
Required sub-elements	**type** The value of this sub-element must be "html". **value** Used when adding HTML markup that is either entity-encoded or contained within a CDATA element. **source** Used with a fully qualified URL to an HTML file. When the Web Content Integrator processes the feed it retrieves the file and store its contents in the HTML element in the site area or content item.
Optional sub-elements	None

Examples:

```
<ibmwcm:element name="footer">
 <ibmwcm:type>html</ibmwcm:type>
 <ibmwcm:value>
  &lt;strong&gt;Copyright 2006&lt;/strong&gt;
 </ibmwcm:value>
</ibmwcm:element>


<ibmwcm:element name="footer">
 <ibmwcm:type>html</ibmwcm:type>
 <ibmwcm:value>
  <![CDATA[<strong>Copyright 2006</strong>]]>
 </ibmwcm:value>
</ibmwcm:element>


<ibmwcm:element name="footer">
 <ibmwcm:type>html</ibmwcm:type>
 <ibmwcm:source>
  http://sourcecms.yourco.com/pages/footer.htm
 </ibmwcm:source>
</ibmwcm:element>
```

Rich text element

Table 298. Rich text element

Element parameters:	Details for this element:
Applies to item types	Site areas and content items
Required for item types	None
Allowable Values	The value of the value sub-element must be HTML that is stored in the corresponding rich text element. The HTML can either be entity-encoded or contained within a CDATA element.
Required Attributes	**name** The value of this attribute must match the name of an existing rich text element in a site area or content item.
Optional Attributes	None
Required sub-elements	**type** The value of this sub-element must be "rich text". These values are not case-sensitive. **value** The value of this sub-element must be HTML markup that is either entity-encoded or contained within a CDATA element.
Optional sub-elements	None

Examples:

```
<ibmwcm:element name="body">
 <ibmwcm:type>rich text</ibmwcm:type>
 <ibmwcm:value>
  &ltp&gt;This is the content&ltp/p&gt;
 </ibmwcm:value>
</ibmwcm:element>
```

```
<ibmwcm:element name="body">
 <ibmwcm:type>rich text</ibmwcm:type>
 <ibmwcm:value>
  <![CDATA[<p>This is the content</p>]]>
 </ibmwcm:value>
</ibmwcm:element>
```

File resource element

Table 299. File resource element

Element parameters:	Details for this element:
Applies to item types	File resource components, site areas, and content items
Required for item types	File resource components.
Allowable Values	The value must be a fully qualified URL that points to the binary file that is to be uploaded into file resource element.

Table 299. File resource element (continued)

Element parameters:	Details for this element:
Required Attributes	**name** The value of this attribute must match the name of an existing file resource element in a site area or content item.
Optional Attributes	None
Required sub-elements	**type** The value of this sub-element must be "file". These values are not case-sensitive. **source** The value must be a fully qualified URL that points to the binary file that is to be uploaded into a file resource element. **Note:** If the URL contains non-ascii characters, the non-ascii characters must be encoded. For example: http://server:port/wps/wcm/%E4%B8%AD%E6%96%87/%E7%BB%84%E4%BB%B6.pdf
Optional sub-elements	**fileName** The value of this sub-element must be the file name and extension to be applied to the file attachment when added to the file resource element. If this sub-element is present in the feed entry, then its value is used as the file name of the attachment in Web Content Manager regardless of what the name of the file was on the source server. This is useful if the URL to the binary file does not include the file name as is the case with some content management systems. If the fileName sub-element is not present in the feed, then the Web Content Integrator attempts to determine the file name from the URL in the value sub-element by taking all of the characters following the last forward slash in the URL.

Examples:

```
<ibmwcm:element name="attachment">
 <ibmwcm:type>file</ibmwcm:type>
 <ibmwcm:source>
  http://sourcecms.yourco.com/files/plan.doc
 </ibmwcm:source>
</ibmwcm:element>


<ibmwcm:element name="attachment">
 <ibmwcm:type>file</ibmwcm:type>
 <ibmwcm:source>
```

```
  http://sourcecms.yourco.com/files/plan.doc
 </ibmwcm:source>
 <ibmwcm:fileName>MktgPlan.doc</ibmwcm:fileName>
</ibmwcm:element>
```

Image element

Table 300. Image element

Element parameters:	Details for this element:
Applies to item types	Image components, site areas, and content items
Required for item types	Image components.
Allowable Values	The value must be a fully qualified URL that points to the binary file that is to be uploaded into image element.
Required Attributes	**name** The value of this attribute must match the name of an existing image element in a site area or content item.
Optional Attributes	None
Required sub-elements	**type** The value of this sub-element must be "image". These values are not case-sensitive. **source** The value must be a fully qualified URL that points to the binary file that is to be uploaded into an image element. **Note:** If the URL contains non-ascii characters, the non-ascii characters must be encoded. For example: http://server:port/wps/wcm/%E4%B8%AD%E6%96%87/%E7%BB%84%E4%BB%B6.jpg

Table 300. Image element (continued)

Element parameters:	Details for this element:
Optional sub-elements	**fileName** The value of this sub-element must be the file name and extension to be applied to the file attachment when added to the image element. If this sub-element is present in the feed entry, then its value is used as the file name of the attachment in Web Content Manager regardless of what the name of the file was on the source server. This is useful if the URL to the binary file does not include the file name as is the case with some content management systems. If the fileName sub-element is not present in the feed, then the Web Content Integrator attempts to determine the file name from the URL in the value sub-element by taking all of the characters following the last forward slash in the URL. **htmlName** The HTML tag name for the image. **altText** The alternative text describing the image. **height** The height of the image. The value must be a number (for pixels) or a percentage. **width** The width of the image. The value must be a number (for pixels) or a percentage. **border** The width of a border around the image. The value must be a number (for pixels).

Examples:

```
<ibmwcm:element name="image">
 <ibmwcm:type>image</ibmwcm:type>
 <ibmwcm:source>
  http://sourcecms.yourco.com/images/logo.gif
 </ibmwcm:source>
</ibmwcm:element>


<ibmwcm:element name="image">
 <ibmwcm:type>image</ibmwcm:type>
 <ibmwcm:source>
  http://sourcecms.yourco.com/images/logo.gif
 </ibmwcm:source>
 <ibmwcm:fileName>yourco_logo.doc</ibmwcm:fileName>
</ibmwcm:element>
```

Date element

Table 301. date element

Element parameters:	Details for this element:
Applies to item types	Date and time components, site areas, and content items
Required for item types	Date and time components.
Allowable Values	A date or time value to be stored in a date and time element.
Required Attributes	**name** The value of this attribute must match the name of an existing date and time element in a site area or content item.
Optional Attributes	None
Required sub-elements	**type** The value of this sub-element must be "date". This value is not case-sensitive. **value** The value of this sub-element must be a date and time value in the RFC 822 format.
Optional sub-elements	**format** This allows the feed producer to specify the format of the date or time. A value of "date" causes the date and time element to be set to display only the date portion of the data. Likewise, a value of "time" displays only the time portion of the data. Any other value, or the absence of this sub-element, results in both portions of the data being displayed. The values for this sub-element are not case-sensitive.

Example:

```
<ibmwcm:element name="StartDate">
 <ibmwcm:type>date</ibmwcm:type>
 <ibmwcm:value>
  Thu, 14 Apr 1966 15:15:00 EDT
 </ibmwcm:value>
 <ibmwcm:format>date</ibmwcm:format>
</ibmwcm:element>
```

Link element

Table 302. link element

Element parameters:	Details for this element:
Applies to item types	Link components, site areas, and content items
Required for item types	Link components.

Table 302. link element (continued)

Element parameters:	**Details for this element:**
Allowable Values	This component contains the information that is required to configure a link element. There are a number of optional sub-elements that can be used to set the various parameters of the Link field.
Required Attributes	**name** The value of this attribute must match the name of an existing link element in a site area or content item.
Optional Attributes	None
Required sub-elements	**type** The value of this sub-element must be "link". This value is not case-sensitive. **format** • ExternalLink: creates a link to a URL external to your Web Content Manager system. • ManagedContent: creates a link to another Web Content Manager item. • ExistingLink: creates a link to a pre-existing link component **value** The value of this sub-element depends on what was specified in the "format" sub-element: • ExternalLink: a URL. • ManagedContent: the GUID of an existing Web Content Manager item. • ExistingLink: the GUID of an existing link component.

Table 302. link element (continued)

Element parameters:	Details for this element:
Optional sub-elements	**linkText** This sub-element allows the feed producer to specify what is rendered between the `<a href="">` and `</a>` portions of the anchor tag. Possible values include: • A plain text string • A string of HTML markup that is either entity-encoded or enclosed in a CDATA tag • The GUID of a feed entry that describes an image component A required type attribute indicates which of the previous value types to use. It can be set to "text", "html", or "imageGuid". If the linkText sub-element is not included in the feed, the link text default to the value set in either "ExternalLink", "ManagedContent" or "ExistingLink". **linkDescription** This sub-element provides a mechanism to specify a description for the link element. If this sub-element is not present, the description on the link element defaults to the value of the description of the site area or content item. **linkTarget** The sub-element is used to set the target browser window where the link is displayed when it is clicked. Allowable values are: "none", "_blank", "_top", "_self", "_parent", and "{NEW_WINDOW_NAME}". If this element is not present in the feed the link target defaults to "none" meaning that the link will open in the same browser window as the page containing it. **queryString** If present, the value of this sub-element is appended to the generated HREF of the link. This value must be encapsulated in a CDATA tag to prevent parsing problems. **additionalAttributes** The value of this sub-element is inserted into the `<a>` tag as extra HTML attributes. **allowClear** This sub-element controls whether the value in the link element can be deleted. It must be set to either "true" or "false". It defaults to "true"

Simple Link to external URL for IBM.com

```
<ibmwcm:element name="Link">
 <ibmwcm:type>link</ibmwcm:type>
 <ibmwcm:value>http://www.ibm.com</ibmwcm:value>
 <ibmwcm:format>ExternalLink</ibmwcm:format>
 <ibmwcm:linkText type="text">IBM</ibmwcm:linkText>
</ibmwcm:element>
```

Expanded link to external URL for IBM.com

```
<ibmwcm:element name="Link">
 <ibmwcm:type>link</ibmwcm:type>
 <ibmwcm:value>http://www.ibm.com/search</ibmwcm:value>
 <ibmwcm:format>ExternalLink</ibmwcm:format>
 <ibmwcm:linkText type="text">RSS Feed Format Resources</ibmwcm:linkText>
 <ibmwcm:linkDescription>Search for RSS Feed Format</ibmwcm:linkDescription>
 <ibmwcm:queryString><![CDATA[?q=rss+feed+format]]></ibmwcm:queryString>]]>
 <ibmwcm:linkTarget>_blank</ibmwcm:linkTarget>
 <ibmwcm:additionalAttributes>class="extLink"</ibmwcm:additionalAttributes>
</ibmwcm:element>
```

Simple link to a file resource component

```
<ibmwcm:element name="Link">
 <ibmwcm:type>link</ibmwcm:type>
 <ibmwcm:value>63000001</ibmwcm:value>
 <ibmwcm:format>ManagedContent</ibmwcm:format>
 <ibmwcm:linkText type="imageGuid">290df293e20a</ibmwcm:linkText>
 <ibmwcm:allowClear>true</ibmwcm:allowClear>
</ibmwcm:element>
```

Simple Link to another content item

```
<ibmwcm:element name="Link">
 <ibmwcm:type>link</ibmwcm:type>
 <ibmwcm:value>80220102</ibmwcm:value>
 <ibmwcm:format>ManagedContent</ibmwcm:format>
 <ibmwcm:linkText type="html"><![CDATA[<b>Marketing Plan</b></ibmwcm:linkText>]]>

</ibmwcm:element>
```

Number element

Table 303. number element

Element parameters:	**Details for this element:**
Applies to item types	Number components, site areas, and content items
Required for item types	Number components.
Allowable Values	A numerical value to be stored in a number element.
Required Attributes	**name** The value of this attribute must match the name of an existing number element in a site area or content item.
Optional Attributes	None
Required sub-elements	**type** The value of this sub-element must be "number". This value is not case-sensitive. **value** A numerical value to be stored in a number element.

Table 303. number element (continued)

Element parameters:	Details for this element:
Optional sub-elements	**format** This allows the feed producer to specify the format for the number element. For example, if a value of "integer" is specified, then only data in the format of whole numbers are imported.

Example:

```
<ibmwcm:element name="FileSize">
 <ibmwcm:type>number</ibmwcm:type>
 <ibmwcm:value>34082</ibmwcm:value>
 <ibmwcm:format>integer</ibmwcm:format>
</ibmwcm:element>
```

Option selection element

Table 304. option selection element

Element parameters:	Details for this element:
Applies to item types	Site areas and content items
Required for item types	None
Allowable Values	This component contains a list of values that are set as the selected options in an option selection element.
Required Attributes	**name** The value of this attribute must match the name of an existing option selection element in a site area or content item.
Optional Attributes	None
Required sub-elements	**type** The value of this sub-element must be "option". This value is not case-sensitive.

Table 304. option selection element (continued)

Element parameters:	Details for this element:
Optional sub-elements	**optionType** This is used to define the type of option selection element. • Specify `user` when referencing from a list of user-defined options. • Specify `taxonomy` when referencing options from an existing taxonomy. You can only specify 1 option type per option selection element. If no option type is defined, "user" will be used by default. **selectedCategory** This is used with the option type of "taxonomy" and is used to specify the path to a category you want to use in the option selection element. You can use as many selectedCategory elements as you require. If the taxonomy exists in a different library from the option selection element, you can also specify the library name. For example: `<ibmwcm:selectedCategory library="shared">Days/Monday</ibmwcm:selectedCategory>` You must specify the name of each category and taxonomy you want to reference. If they do not exist, they are created when the feed is processed. **selectedOption** This is used with the option type of "user" and is used to specify a list of user-defined options. You can use as many selectedOption elements as you require.

Example 1:

Selecting a single category where the category is in the same library as the item. In this example "Days" is the name of a taxonomy and "Monday" is the name of a category.

```
<ibmwcm:element name="DayOfTheWeek">
 <ibmwcm:type>option</ibmwcm:type >
    <ibmwcm:optionType>taxonomy</ibmwcm:optionType>
    <ibmwcm:selectedCategory>Days/Monday</ibmwcm:selectedCategory>
< /ibmwcm:element>
```

Example 2:

Selecting multiple categories where the categories are in the same library as the item.

```
<ibmwcm:element name="DayOfTheWeek">
  <ibmwcm:type>option</ibmwcm:type >
    <ibmwcm:optionType>taxonomy</ibmwcm:optionType>
    <ibmwcm:selectedCategory>Days/Monday</ibmwcm:selectedCategory>
  <ibmwcm:selectedCategory>Days/Tuesday</ibmwcm:selectedCategory>
< ;/ibmwcm:element>
```

Example 3:

Selecting a single category where the category is in a different library to the item.

```
<ibmwcm:element name="DayOfTheWeek">
  <ibmwcm:type>option</ibmwcm:type >
    <ibmwcm:optionType>taxonomy</ibmwcm:optionType>
    <ibmwcm:selectedCategory library="shared">Days/Monday</ibmwcm:selectedCategory>
</ibmwcm:element>
```

Example 4:

Selecting a user-defined option. In this example "False" is an option that is defined by a user on the Authoring Template for an item.

```
<ibmwcm:element name="Enable">
    <ibmwcm:type>option</ibmwcm:type >
    <ibmwcm:optionType>user</ibmwcm:optionType>
    <ibmwcm:selectedOption>False</ibmwcm:selectedOption>
</ibmwcm:element>
```

Component reference element

Table 305. component reference element

Element parameters:	Details for this element:
Applies to item types	Site areas and content items
Required for item types	None
Allowable Values	This component contains the GUID of a component.
Required Attributes	**name** The value of this attribute must match the name of an existing component reference element in a site area or content item.
Optional Attributes	None
Required sub-elements	**type** The value of this sub-element must be "reference". This value is not case-sensitive. **value** This component contains the GUID of a component.
Optional sub-elements	None

Example:

```
<ibmwcm:element name="Footer">
 <ibmwcm:type>reference</ibmwcm:type>
 <ibmwcm:value>29bc2daf3289</ibmwcm:value>
</ibmwcm:element>
```

User selection element

Table 306. user selection element

Element parameters:	Details for this element:
Applies to item types	User selection components, site areas, and content items
Required for item types	User selection components.
Allowable Values	A list of names of users to be selected in a user selection element.
Required Attributes	**name** The value of this attribute must match the name of an existing user selection element in a site area or content item.
Optional Attributes	None
Required sub-elements	**type** The value of this sub-element must be "userSelect". This value is not case-sensitive. **value** The value of this sub-element must be a comma-separated list of user names. When processed the Web Content Integrator attempts to resolve each name to a valid portal user. If a name cannot be resolved, it is skipped.
Optional sub-elements	None

Example:

```
<ibmwcm:element name="Users">
 <ibmwcm:type>userSelect</ibmwcm:type>
 <ibmwcm:value>wpsadmin,John Smith</ibmwcm:value>
</ibmwcm:element>
```

Style sheet element

Table 307. style sheet element

Element parameters:	Details for this element:
Applies to item types	Stylesheet components
Required for item types	Stylesheet components
Allowable Values	A URL that points to a CSS file that is uploaded into a stylesheet component.
Required Attributes	None
Optional Attributes	None
Required sub-elements	**type** The value of this sub-element must be "styleSheet". This value is not case-sensitive. **source** A URL that points to a CSS file that is uploaded into the stylesheet component.

Table 307. style sheet element (continued)

Element parameters:	Details for this element:
Optional sub-elements	**format** This sub-element is used to specify the type of the stylesheet. Valid options are: "preferred", "alternate", and "persistent". If this sub-element is not present in the feed, or if any value other than the previous values is specified, the type will default to "preferred". **mediaType** This sub-element specifies the media type of the stylesheet. Valid values are: "all", "aural", "braille", "handheld", "print", "projection", "screen", "tty", "tv", and "unspecified". If the mediaType sub-element is not present in the feed, or if any value other than the previous values is specified, the media type defaults to "unspecified". **cssTitle** The value of this sub-element must be a string that is set as the value of the title attribute when the link to the CSS file is rendered.

Example:

```
<ibmwcm:element name="cssFile">
 <ibmwcm:type>styleSheet</ibmwcm:type>
 <ibmwcm:source>http://www.yourco.com/styles/news.css</ibmwcm:source>
 <ibmwcm:format>alternate</ibmwcm:format>
 <ibmwcm:mediaType>print</ibmwcm:mediaType>
</ibmwcm:element>
```

Workflow control elements:

Workflow elements is used to specify workflow-related parameters when content items are created that use a workflow.

workflow element

This element specifies the workflow name and stage in which a content item is created. The value in the name attribute must match that of an existing workflow.

If the workflow is in a different library from the content item, you can specify the library name as well. If no library is specified, the Web Content Integrator searches the library that was specified in the `<ibmwcm:library>` element. If no `<ibmwcm:library>` element is present, the Web Content Integrator searches the default library that is specified in the task configuration.

Table 308. workflow element

Element parameters:	Details for this element:
Applies to item types	Content items

Table 308. workflow element (continued)

Element parameters:	Details for this element:
Required for item types	None
Allowable Values	Workflow name and related workflow stage.
Required Attributes	The name of an existing workflow, or the library and name of an existing library and workflow that is separated by a forward slash. For example: • 3-Stage-Workflow • Library1/3-Stage-Workflow
Optional Attributes	None
Required sub-elements	**workflowStage** The value of this sub-element must be the name of a workflow stage that is included in the named workflow.
Optional sub-elements	None

Examples:

```
<ibmwcm:workflow name="StdWorkflow">
 <ibmwcm:workflowStage>Live</ibmwcm:workflowStage>
</ibmwcm:workflow>


<ibmwcm:workflow name="DesignLib/StdWorkflow">
 <ibmwcm:workflowStage>Live</ibmwcm:workflowStage>
</ibmwcm:workflow>
```

date elements

This element is used to specify an RFC 822 formatted date that is used as the date of one of the date fields in the content item.

Table 309. date elements

Element parameters:	Details for this element:
Applies to item types	Content items
Required for item types	None

Table 309. date elements (continued)

Element parameters:	Details for this element:
Allowable Values	**publishDate** Used to set the published date of an item. If this element is not present in the <item>, the value in the <pubDate> element is used to set the publish date of the content item. This requires a publish action to be included in a workflow stage in the specified workflow. **expirationDate** Used to set the expiry date of an item. This requires an expire action to be included in a workflow stage in the specified workflow. **genDateOne and genDateTwo** Used to populate the general date fields of an item.
Required Attributes	An RFC 822 formatted date.
Optional Attributes	None
Required sub-elements	None
Optional sub-elements	None

Examples:

```
<ibmwcm:publishDate>
 Fri, 31 Oct 2008 15:32:00 EST
</ibmwcm:publishDate>


<ibmwcm:expirationDate>
 Sun, 1 Nov 2009 12:00:00 EST
</ibmwcm:expirationDate>


<ibmwcm:genDateOne>
 Wed, 1 Nov 2006 09:30:00 EST
</ibmwcm:genDateOne>


<ibmwcm:genDateTwo>
 Thu, 2 Nov 2006 09:30:00 EST
</ibmwcm:genDateTwo>
```

additionalViewer element

This element allows the additional viewers field to be populated via the feed. It should contain the common name of a single user or group that is to be added to the field. To specify multiple users and groups, use multiple additionalViewer elements.

Table 310. additionalViewer element

Element parameters:	Details for this element:
Applies to item types	Content items
Required for item types	None
Allowable Values	• [all users] • [all authenticated portal users] • The common name of any valid portal user or group.
Required Attributes	None
Optional Attributes	None
Required sub-elements	None
Optional sub-elements	None

Examples:

```
<ibmwcm:additionalViewer>[all users]</ibmwcm:additionalViewer>

<ibmwcm:additionalViewer>Sales</ibmwcm:additionalViewer>

<ibmwcm:additionalViewer>jsmith</ibmwcm:additionalViewer>
```

Security control elements:

The security control element is used to set access controls on the item that is being created.

access element

The access element provides a mechanism to set the access controls on Web Content Manager items via the feed. A feed entry can contain multiple access sub-elements. Each access element should contain the common name of a single, valid, portal user or group. The users and groups specified in the access element are added to the system defined security fields on the Web Content Manager items.

Table 311. access element

Element parameters:	Details for this element:
Applies to item types	Content
Required for item types	None
Allowable Values	• [all users] • [all authenticated portal users] • The common name of any valid portal user or group.
Required Attributes	None

Table 311. access element (continued)

Element parameters:	Details for this element:
Optional Attributes	**type** This attribute allows the feed producer to specify the exact access level that should be granted to the user or group. The allowable values for the type attribute correspond to the access levels available in Web Content Manager: "user", "contributor", "editor" and "manager". If the type attribute is not specified, "user" access is applied. **inheritance** You can specify whether inheritance is enabled or by adding either `inheritance="enabled"` or `inheritance="disabled"`. If not specified, the default or current inheritance setting for an item is used.
Required sub-elements	None
Optional sub-elements	

Examples:

```
<ibmwcm:access type="user">
 [all users]
</ibmwcm:access>

<ibmwcm:access type="editor" inheritance="enabled">
 Sales
</ibmwcm:access>
```

Association control element:

The association element is used to create an association between two or more separate content items. It is used to support some content item relationships that are not native to Web Content Manager but that are found in some other web content management systems. One example would be a technical journal article that needs to include some data from one or more author biography content items.

An association element is tied to a text element. The value of the element in the feed is a comma-delimited list of GUIDs that represent the other external content items that are to be associated with the text element. When the Web Content Integrator processes the feed, it attempts to resolve the GUIDs with the corresponding Web Content Manager items. A custom JSP component is then used in the presentation template to locate the linked content items at render time and display them in the context of the container content item.

Since this type of linkage is not a natural feature of Web Content Manager it relies on a custom JSP component for its implementation and there are some restrictions that are related to using an association element:

- As Web Content Manager DocumentIds are used to create the association to the other content items, the format of the DocumentId is not guaranteed to be

constant from version to version so there is some risk that the linkages might break following a Web Content Manager upgrade or migration

- The referential integrity of Web Content Manager does not apply to association element links.

Table 312. association element

Element parameters:	**Details for this element:**
Applies to item types	Content
Required for item types	None
Allowable Values	A list of GUIDs that map to other feed entries that are linked to associated content items.
Required Attributes	**name** The value of this attribute must match the name of an existing text element.
Optional Attributes	None
Required sub-elements	**type** The value of this sub-element must be "association". It is not case-sensitive. **value** The value of this sub-element must be a comma-delimited list of GUIDs that point to the feed entries that are linked to associated content items.
Optional sub-elements	None

Example:

```
<ibmwcm:element name="authors">
 <ibmwcm:type>association</ibmwcm:type>
  <ibmwcm:value>234ed298cf,7023bc23f1e</ibmwcm:value>
</ibmwcm:element>
```

How to handle embedded links

From time to time, the content that is being retrieved by the Web Content Integrator contains embedded links to images, files, and other content within the feed. You can use the link tag in your feed to represent embedded links so that they are converted into links to other Web Content Manager items that are created when the feed is processed.

Three types of embedded links can be processed: images, files, and content. In all cases, the link tag must include a GUID that points to another item in the feed, which describes the target item.

Content links

In this example, the following feed content contains a link to another web page:

```
<p>
<a href="http://cmsserver.ibm.com/news/one.htm">Press Release</a>
</p>
```

To ensure that the embedded link is converted into a link to a related content item that is being created when the feed is processed, you add the following code to your feed. It contains a GUID to the HTML file also being processed by the Web Content Integrator:

```
<![CDATA[<p>
<a href='<link type="content" guid="80000002"/>'>Press Release</a>
This is some text.
</p>]]>
```

When the content of the feed is saved within Web Content Manager, a link to a content item is added to the content.

```
<p>
<a href="wps/wcm/myconnect/Portal/Press/Announcement1?contentIDR=29e04295034efb">Press Release</a>
This is some text.
</p>
```

Example feed

```
<?xml version="1.0"?>
<rss version="2.0" xmlns:ibmwcm="http://purl.org/net/ibmfeedsvc/wcm/1.0" >
<channel>
<title>Sample Feed</title>
<link>http://www.ibm.com/feeds/sample.rss</link>
<description>An example RSS Feed</description>
<lastBuildDate>Tue, 31 Oct 2006 10:31:00 EST</lastBuildDate>
<item>
<title>News Item Two</title>
<link>http://www.ibm.com/news/two.htm</link>
<description>
This is a summary of the second news article
</description>
<pubDate>Tue, 31 Oct 2006 10:31:00 EST</pubDate>
<guid>80000001</guid>
<ibmwcm:action>add</ibmwcm:action>
<ibmwcm:itemType>content</ibmwcm:itemType>
<ibmwcm:authoringTemplate>News</ibmwcm:authoringTemplate>
<ibmwcm:path>/Portal/News</ibmwcm:path>
<ibmwcm:workflow name="Std Workflow">
<ibmwcm:workflowStage>Live</ibmwcm:workflowStage>
</ibmwcm:workflow>
<ibmwcm:component name="BODY">
<ibmwcm:type>rich text</ibmwcm:type>
<ibmwcm:value>
<![CDATA[<p>
<a href='<link type="content" guid="80000002" />'>some content link</a>
This is some text.
</p>]]>
</ibmwcm:value>
</ibmwcm:component>
</item>
<item>
<title>Announcment1</title>
<link>http://cmsserver.ibm.com/news/one.htm</link>
<description>This is an announcment press release.</description>
<pubDate>Tue, 31 Oct 2006 10:11:00 EST</pubDate>
<guid>80000002</guid>
<ibmwcm:action>add</ibmwcm:action>
<ibmwcm:itemType>Content</ibmwcm:itemType>
<ibmwcm:authoringTemplate>Press Release</ibmwcm:authoringTemplate>
<ibmwcm:path>/Portal/Press</ibmwcm:path>
<ibmwcm:workflow name="Std Workflow">
<ibmwcm:workflowStage>Live</ibmwcm:workflowStage>
</ibmwcm:workflow>
<ibmwcm:component name="BODY">
<ibmwcm:type>rich text</ibmwcm:type>
<ibmwcm:value>This is some more text.</ibmwcm:value>
```

```
</ibmwcm:component>
</item>
</channel>
</rss>
```

File links

In this example, the following feed content contains a link to a PDF file:

```
<p>
<a href="http://cmsserver.ibm.com/files/spec.pdf">Product Spec</a>
This is some text.
</p>
```

To ensure that the embedded link is converted into a link to a related file resource component that is being created when the feed is processed, you add the following code to your feed. It contains a GUID to the PDF file also being processed by the Web Content Integrator:

```
<![CDATA[<p>
<a href='<link type="file" guid="50000002"/>'>Product Spec</a>
This is some text.
</p>]]>
```

When the content of the feed is saved within Web Content Manager, a component tag is added to the content.

```
<p>
<a href="<Component name="ProductSpec" />">Product Spec</a>
This is some text.
</p>
```

Example feed

```
<?xml version="1.0"?>
<rss version="2.0" xmlns:ibmwcm="http://purl.org/net/ibmfeedsvc/wcm/1.0" >
   <channel>
      <title>Sample Feed</title>
      <link>http://www.ibm.com/feeds/sample.rss</link>
      <description>An example RSS Feed</description>
      <lastBuildDate>Tue, 31 Oct 2006 10:31:00 EST</lastBuildDate>

      <item>
         <title>News Item Two</title>
         <link>http://www.ibm.com/news/two.htm</link>
         <description>
            This is a summary of the second news article
         </description>
         <pubDate> Tue, 31 Oct 2006 10:31:00 EST</pubDate>
         <guid>80000001</guid>
         <ibmwcm:action>add</ibmwcm:action>
         <ibmwcm:itemType>Content</ibmwcm:itemType>
         <ibmwcm:authoringTemplate>News</ibmwcm:authoringTemplate>
         <ibmwcm:path>/Portal/News</ibmwcm:path>
         <ibmwcm:workflow name="Std Workflow">
            <ibmwcm:workflowStage>Live</ibmwcm:workflowStage>
         </ibmwcm:workflow>
         <ibmwcm:component name="BODY">
            <ibmwcm:type>rich text</ibmwcm:type>
            <ibmwcm:value>
<![CDATA[<p>
<a href="<link type="file" guid="50000002" />">some file link</a>
This is some text.
</p>]]>
            </ibmwcm:value>
         </ibmwcm:component>
      </item>
```

```
      <item>
         <title>Product Spec</title>
         <link>http://cmsserver.ibm.com/files/spec.pdf</link>
         <description>This is the product spec document.</description>
         <pubDate> Tue, 31 Oct 2006 10:11:00 EST</pubDate>
         <guid>50000002</guid><ibmwcm:action>add</ibmwcm:action>
         <ibmwcm:itemType>component</ibmwcm:itemType>
         <ibmwcm:component name="File">
            <ibmwcm:type>file</ibmwcm:type>
            <ibmwcm:value>
              http://cmsserver.ibm.com/files/spec.pdf
            </ibmwcm:value>
         </ibmwcm:component>
      </item>
   </channel>
</rss>
```

Image links

In this example, the following feed content contains a link to an image file:

```
<p>
<img src="http://cmsserver.ibm.com/images/mylogo.gif" />
This is some text.
</p>
```

To ensure that the embedded image link is converted into a link to a related image component that is being created when the feed is processed, you add the following code to your feed. It contains a GUID to the image file also being processed by the Web Content Integrator:

```
<![CDATA[<p>
<link type="image" guid="50000001" />
This is some text.
</p>]]>
```

When the content of the feed is saved within Web Content Manager, a component tag is added to the content.

```
<p>
<Component name="My Logo" />
This is some text.
</p>
```

Example feed

```
<?xml version="1.0"?>
<rss version="2.0" xmlns:ibmwcm="http://purl.org/net/ibmfeedsvc/wcm/1.0" >
   <channel>
      <title>Sample Feed</title>
      <link>http://www.ibm.com/feeds/sample.rss</link>
      <description>An example RSS Feed</description>
      <lastBuildDate>Tue, 31 Oct 2006 10:31:00 EST</lastBuildDate>

      <item>
         <title>News Item Two</title>
         <link>http://www.ibm.com/news/two.htm</link>
         <description>
            This is a summary of the second news article
         </description>
         <pubDate> Tue, 31 Oct 2006 10:31:00 EST</pubDate>
         <guid>80000001</guid>
         <ibmwcm:action>add</ibmwcm:action>
         <ibmwcm:itemType>Content</ibmwcm:itemType>
         <ibmwcm:authoringTemplate>News</ibmwcm:authoringTemplate>
         <ibmwcm:path>/Portal/News</ibmwcm:path>
```

```
         <ibmwcm:workflow name="Std Workflow">
            <ibmwcm:workflowStage>Live</ibmwcm:workflowStage>
         </ibmwcm:workflow>
         <ibmwcm:component name="BODY">
            <ibmwcm:type>rich text</ibmwcm:type>
            <ibmwcm:value>
<![CDATA[<p>
<link type="image" guid="50000001" />
This is some text.
</p>]]>
            </ibmwcm:value>
         </ibmwcm:component>
      </item>

      <item>
         <title>My Logo</title>
         <link>http://cmsserver.ibm.com/images/mylogo.gif</link>
         <description>This is our logo image.</description>
         <pubDate> Tue, 31 Oct 2006 10:21:00 EST</pubDate>
         <guid>50000001</guid><ibmwcm:action>add</ibmwcm:action>
         <ibmwcm:itemType>component</ibmwcm:itemType>
         <ibmwcm:component name="Image">
            <ibmwcm:type>image</ibmwcm:type>
            <ibmwcm:value>
              http://cmsserver.ibm.com/images/mylogo.gif
            </ibmwcm:value>
         </ibmwcm:component>
      </item>
   </channel>
</rss>
```

Automatic image tag processing

The Web Content Integrator has an optional feature, which can be used in place of the `<link type="image" />` tag. If enabled, the Web Content Integrator automatically parses the values of any html or rich text elements and search for HTML `<img/>` tags that are embedded within the content. If any are found, the Web Content Integrator attempts to retrieve the referenced image file, create an image component, and then rewrite the reference so that it points to the new image component.

This feature can be enabled by editing the `disable.img.proc` setting in the `WCMConsumerPlugin.properties` file and setting it to "false". Automatic image tag processing is disabled by default.

If enabled:

- Only the content that is imported into HTML elements and components or Rich Text elements and components are processed.
- All image references matching the pattern "`<img src= ...`" are processed. Image references that are included within JavaScript code or CSS styles are not processed.
- The URLs specified in the "src" attributes of those image tags are converted to fully qualified URLs by using other information about the content item in the feed. Relative links must be relative to the URL in the <link> element of the feed that follows the standard rules for relative links.
- The image files are stored as shared image components. The access controls on the image component are set to match that of the content item, which references it. The name of the image component is based on the server-relative path to the image. For example, an image that is at `http://<host_name>/resources/images/sm_logo.gif` is named `resources.images.sm_logo.gif`.

- The URLs specified in the "src" attributes of the image tags must not contain a query string because everything after a question mark is ignored when the image component is created.

  For example, For example, `<img src="http://<host_name>/program/path?param1=hello&param2=a.jpg"....>` and `<img src="http://<host_name>/program/path?param1=world&param2=b.jpg" ...>` both creates or updates the same image component named "program.path".

Processing images

The Web Content Integrator has an image-processing feature, which allows images that are referenced within HTML and rich text elements to be created as image components while processing a feed.

Image-processing configuration

The following parameters can be set in the `WCMConsumerPlugin.properties` file:

Enable and disable image processing

By default, image processing is enabled. To disable image processing set the property: *disable.img.processing=true*

If image processing is enabled, then the URLs specified in the "src" attributes of the image tags must not contain a query string because everything after a question mark is ignored when the image component is created.

For example, `<img src="http://<host_name>/program/path?param1=hello&param2=a.jpg" ...>` and `<img src="http://<host_name>/program/path?param1=world&param2=b.jpg" ...>` both creates or updates the same image component named "program.path".

Enable and disable embedded rich text images

Images can be embedded directly into the rich text elements instead of first creating an image component. This function makes images within rich text elements that are consumed by the Web Content Integrator be processed identically to images added to rich text elements in the authoring UI. By default embedded rich text images are disabled. To enable them, set the properties:

- *disable.img.processing=false*
- *richtext.embedded.images.enabled=true*

Absolute image source URLs

Images within HTML and rich text elements can be specified by using absolute HTTP URLs within the image source attribute. The Web Content Integrator processes absolute image source URLs as specified in the feed.

Example feed item:

```
<item>
    <title>RichText Component With Image 1</title>
    <pubDate>Thu, 30 Mar 2011 16:00:00 EDT</pubDate>
    <guid>Image_Example_1</guid>
    <ibmwcm:action>add</ibmwcm:action>
    <ibmwcm:itemType>Component</ibmwcm:itemType>
    <ibmwcm:element>
      <ibmwcm:type>rich text</ibmwcm:type>
      <ibmwcm:value>
         <![CDATA[
```

```
        <p>Image 1:</p><img src="http://wci-feed-server/img/f/flower.jpg"/>
        <p>Image 2:</p><img src="http://wci-feed-server/img/g/guitar.jpg"/>
        ]]>
      </ibmwcm:value>
    </ibmwcm:element>
  </item>
```

Relative image source URLs

Images within HTML and rich text elements can be specified by using relative HTTP URLs within the image source attribute and a base URL from the link element of the item. The Web Content Integrator processes relative image source URLs as a concatenation of the item's link element and image source attributes in the HTML.

Example feed item:

```
<item>
    <title>RichText Component With Image 2</title>
    <pubDate>Thu, 30 Mar 2011 16:00:00 EDT</pubDate>
    <guid>Image_Example_2</guid>
    <link>http://wci-feed-server</link>
    <ibmwcm:action>add</ibmwcm:action>
    <ibmwcm:itemType>Component</ibmwcm:itemType>
    <ibmwcm:element>
      <ibmwcm:type>rich text</ibmwcm:type>
      <ibmwcm:value>
         <![CDATA[
         <p>Image 1:</p><img src="/img/p/penguin.jpg"/>
         <p>Image 2:</p><img src="/img/s/sunset.jpg"/>
         ]]>
      </ibmwcm:value>
    </ibmwcm:element>
  </item>
```

How to use both absolute and relative image source URLs

Images within HTML and rich text elements can be specified by using a combination of relative and absolute HTTP URLs within the image source attributes. The Web Content Integrator processes relative image source URLs as a concatenation of the item's link element and image source attributes in the HTML. The Web Content Integrator processes absolute image source URLs as specified in the feed.

Example feed item:

```
<item>
    <title>RichText Component With Image 3</title>
    <pubDate>Thu, 30 Mar 2011 16:00:00 EDT</pubDate>
    <guid>Image_Example_3</guid>
    <link>http://wci-feed-server</link>
    <ibmwcm:action>add</ibmwcm:action>
    <ibmwcm:itemType>Component</ibmwcm:itemType>
    <ibmwcm:element>
      <ibmwcm:type>rich text</ibmwcm:type>
      <ibmwcm:value>
         <![CDATA[
         <p>Image 1:</p><img src="http://www.ibm.com/favicon.ico"/>
         <p>Image 2:</p><img src="/img/f/fish.jpg"/>
         <p>Image 3:</p><img src="/img/g/grapes.jpg"/>
         ]]>
      </ibmwcm:value>
    </ibmwcm:element>
  </item>
```

Expected results for a rich text component

Example rich text component feed item:

```
<item>
    <title>RichText Component With Image 4</title>
    <pubDate>Thu, 30 Mar 2011 16:00:00 EDT</pubDate>
    <guid>Image_Example_4</guid>
    <ibmwcm:action>add</ibmwcm:action>
    <ibmwcm:itemType>Component</ibmwcm:itemType>
    <ibmwcm:element>
      <ibmwcm:type>rich text</ibmwcm:type>
      <ibmwcm:value>
         <![CDATA[
         <p>Image 1:</p><img src="http://wci-feed-server/img/l/leaf.jpg"/>
         ]]>
      </ibmwcm:value>
    </ibmwcm:element>
  </item>
```

Rich text component with image processing disabled:

- A rich text component named: "RichText Component With Image 4" is created in the feed library.
- HTML Source of the rich text Component: `<p>Image 1:</p><img src="http://wci-feed-server/img/l/leaf.jpg">`
- The rich text HTML is not modified.
- The image source is unmanaged because Web Content Manager cannot detect a broken image source.

Rich text component with image processing enabled:

- A rich text component named: "RichText Component With Image 4" is created in the feed library.
- An Image Component that is named `"img.l.leaf.jpg"` is created in the feed library
- HTML Source of the rich text Component: `<p>Image 1:</p><img src="/wps/wcm/myconnect/[IMAGE COMPONENT ID]/leaf.jpg?MOD=AJPERES">`
- The image source is replaced with a URL pointing to the newly created image.
- The image source is managed by Web Content Manager and is updated as the image component is updated.

Rich text component with embedded images enabled:

- A rich text component named: "RichText Component With Image 4" is created in the feed library.
- HTML Source of the rich text Component: `<p>Image 1:</p><img src="/wps/wcm/myconnect/[RT COMPONENT UUID]/1/img.l.leaf.jpg?MOD=AJPERES">`
- The rich text HTML is modified replacing the image source with a URL for the image that is embedded within the rich text Component.
- The image source is managed by Web Content Manager and is updated as the rich text changes.

Expected results for a HTML component

Example feed item:

```
<item>
    <title>HTML Component With Image 1</title>
    <pubDate>Thu, 30 Mar 2011 16:00:00 EDT</pubDate>
    <guid>Image_Example_5</guid>
    <ibmwcm:action>add</ibmwcm:action>
    <ibmwcm:itemType>Component</ibmwcm:itemType>
    <ibmwcm:element>
      <ibmwcm:type>html</ibmwcm:type>
      <ibmwcm:value>
         <![CDATA[
         <p>Image 1:</p><img src="http://wci-feed-server/img/l/leaf.jpg"/>
         ]]>
      </ibmwcm:value>
    </ibmwcm:element>
  </item>
```

HTML element with image processing disabled

- HTML Component that is named "HTML Component With Image 1" is created in the feed library.
- HTML Source of the HTML Component: `<p>Image 1:</p><img src="http://wci-feed-server/img/l/leaf.jpg"/>`
- The HTML is not modified.
- The image source is unmanaged because Web Content Manager cannot detect a broken image source.

HTML element with image processing enabled

- HTML Component that is named "HTML Component With Image 1" is created in the feed library.
- Image Component that is named `"img.l.leaf.jpg"` is created in the feed library
- HTML Source of the HTML Component: `<p>Image 1:</p><img src="<Component id="[IMAGE COMPONENT ID]" name="img test 1/img.l.leaf.jpg" format="url"/>"/>`
- The image source is replaced with a component tag that points to the newly created image.
- The image source is managed by Web Content Manager and is updated as the image component is updated.

Example feeds

View some example feeds that illustrate how to represent the various item types within a feed.

Note: If non-ascii data is used in a feed, then encoding="UTF-8" must be specified in the feed: `<?xml version="1.0" encoding="UTF-8"?>`

Site area feed

```
<?xml version="1.0"?>
<rss version="2.0" xmlns:ibmwcm="http://purl.org/net/ibmfeedsvc/wcm/1.0" >
   <channel>
      <title>Sample Feed</title>
      <link>http://www.ibm.com/feeds/sample.rss</link>
      <description>An example RSS Feed</description>
      <lastBuildDate>Tue, 31 Oct 2006 10:31:00 EST</lastBuildDate>

      <item>
         <title>News</title>
         <link>http://www.ibm.com/news/index.htm</link>
         <description>This is a Site Area.</description>
```

```
         <guid>20000001</guid>
         <pubDate> Tue, 31 Oct 2006 10:31:00 EST</pubDate>
         <author>jsample@ibm.com</author>
         <ibmwcm:action>add</ibmwcm:action>
         <ibmwcm:itemType>siteArea</ibmwcm:itemType>
         <ibmwcm:path>/SiteAreaName</ibmwcm:path>
         <ibmwcm:owner>jsample</ibmwcm:owner>
         <ibmwcm:defaultContent>80000002</ibmwcm:defaultContent>
         <ibmwcm:children>
            <ibmwcm:childGuid>20000011</ibmwcm:childGuid>
            <ibmwcm:childGuid>80000002</ibmwcm:childGuid>
         </ibmwcm:children>
    <ibmwcm:templateMap authoring="AT News" presentation="PT News" />
    <ibmwcm:access type="user">[all users]</ibmwcm:access>
      </item>
   </channel>
</rss>
```

Taxonomy feed

```
<?xml version="1.0"?>
<rss version="2.0" xmlns:ibmwcm="http://purl.org/net/ibmfeedsvc/wcm/1.0" >
   <channel>
      <title>Sample Feed</title>
      <link>http://www.ibm.com/feeds/sample.rss</link>
      <description>An example RSS Feed</description>
      <lastBuildDate>Tue, 31 Oct 2006 10:31:00 EST</lastBuildDate>

      <item>
         <title>Audience</title>
         <link>http://www.ibm.com/index.htm</link>
         <description>This is the top-level Taxonomy.</description>
         <guid>30000001</guid>
         <pubDate> Tue, 31 Oct 2006 10:31:00 EST</pubDate>
         <author>jsample@ibm.com</author>
         <ibmwcm:action>add</ibmwcm:action>
         <ibmwcm:itemType>category</ibmwcm:itemType>
         <ibmwcm:path>/</ibmwcm:path>
         <ibmwcm:owner>jsample</ibmwcm:owner>
         <ibmwcm:access type="user">[all users]</ibmwcm:access>
      </item>
   </channel>
</rss>
```

Category feed

```
<?xml version="1.0"?>
<rss version="2.0" xmlns:ibmwcm="http://purl.org/net/ibmfeedsvc/wcm/1.0" >
   <channel>
      <title>Sample Feed</title>
      <link>http://www.ibm.com/feeds/sample.rss</link>
      <description>An example RSS Feed</description>
      <lastBuildDate>Tue, 31 Oct 2006 10:31:00 EST</lastBuildDate>

      <item>
         <title>Employees</title>
         <link>http://www.ibm.com/index.htm</link>
         <description>This is a second-level Category.</description>
         <guid>40000001</guid>
         <pubDate> Tue, 31 Oct 2006 10:31:00 EST</pubDate>
         <author>jsample@ibm.com</author>
         <ibmwcm:action>add</ibmwcm:action>
         <ibmwcm:itemType>category</ibmwcm:itemType>
         <ibmwcm:path>/Audience</ibmwcm:path>
         <ibmwcm:createLinks>
            <ibmwcm:parentGuid>30000001</ibmwcm:parentGuid>
         </ibmwcm:createLinks>
         <ibmwcm:owner>jsample</ibmwcm:owner>
```

```
         <ibmwcm:children>
            <ibmwcm:childGuid>40000011</ibmwcm:childGuid>
         </ibmwcm:children>
         <ibmwcm:access type="user">[all users]</ibmwcm:access>
      </item>
   </channel>
</rss>
```

Component feed

```
<?xml version="1.0"?>
<rss version="2.0" xmlns:ibmwcm="http://purl.org/net/ibmfeedsvc/wcm/1.0" >
   <channel>
      <title>Sample Feed</title>
      <link>http://www.ibm.com/feeds/sample.rss</link>
      <description>An example RSS Feed</description>
      <lastBuildDate>Tue, 31 Oct 2006 10:31:00 EST</lastBuildDate>

      <item>
         <title>IBM Footer</title>
         <link>http://www.ibm.com/files/footer.htm</link>
         <description>This is a shared chunk of HTML.</description>
         <guid>53000001</guid>
         <pubDate> Tue, 31 Oct 2006 10:31:00 EST</pubDate>
         <author>jsample@ibm.com</author>
         <ibmwcm:action>add</ibmwcm:action>
         <ibmwcm:itemType>component</ibmwcm:itemType>
         <ibmwcm:owner>jsample</ibmwcm:owner>
         <ibmwcm:element name="footer">
            <ibmwcm:type>html</ibmwcm:type>
            <ibmwcm:value>
<![CDATA[<span style="font-size: 8pt;">Copyright 2006 <a href="http://www.ibm.com">
  International Business Machines, Inc.</a> All rights reserved.</span>]]>
            </ibmwcm:value>
         </ibmwcm:element>
         <ibmwcm:access type="user">[all users]</ibmwcm:access>
      </item>
   </channel>
</rss>
```

Content feed

```
<?xml version="1.0"?>
<rss version="2.0" xmlns:ibmwcm="http://purl.org/net/ibmfeedsvc/wcm/1.0" >
   <channel>
      <title>Sample Feed</title>
      <link>http://www.ibm.com/feeds/sample.rss</link>
      <description>An example RSS Feed</description>
      <lastBuildDate>Tue, 31 Oct 2006 10:31:00 EST</lastBuildDate>

      <item>
         <title>Release 2.0 Announcement</title>
         <link>http://www.ibm.com/news/Rel2Announce.htm</link>
         <description>This is a content item.</description>
         <guid>80000001</guid>
         <pubDate> Tue, 31 Oct 2006 10:31:00 EST</pubDate>
         <author>jsample@ibm.com</author>
         <category>/Audience/Employees</category>
         <category>/Audience/Customers</category>
         <ibmwcm:action>add</ibmwcm:action>
         <ibmwcm:itemType>content</ibmwcm:itemType>
         <ibmwcm:authoringTemplate>News</ibmwcm:authoringTemplate>
         <ibmwcm:path>/SiteAreaName/News</ibmwcm:path>
         <ibmwcm:createLinks>
            <ibmwcm:parentGuid>20000001</ibmwcm:parentGuid>
         </ibmwcm:createLinks>
         <ibmwcm:owner>jsample</ibmwcm:owner>
         <ibmwcm:keywords>software,content management</ibmwcm:keywords>
```

```
         <ibmwcm:workflow name="Std Workflow">
            <ibmwcm:workflowStage>Live</ibmwcm:workflowStage>
         </ibmwcm:workflow>
         <ibmwcm:publishDate>
            Wed, 1 Nov 2006 06:00:00 EST
         </ibmwcm:publishDate>
         <ibmwcm:expirationDate>
            Sun, 31 Dec 2006 23:59:00 EST
         </ibmwcm:expirationDate>
         <ibmwcm:genDateOne>Fri, 15 Dec 2006 09:00:00 EST</ibmwcm:genDateOne>
         <ibmwcm:additionalViewer>All Dealers</ibmwcm:additionalViewer>
         <ibmwcm:element name="Headline">
            <ibmwcm:type>text</ibmwcm:type>
            <ibmwcm:value>Release 2.0 Ships Today</ibmwcm:value>
         </ibmwcm:element>
         <ibmwcm:element name="Summary">
            <ibmwcm:type>text</ibmwcm:type>
            <ibmwcm:value>
The long anticipated Release 2.0 of our flagship product became available earlier today.
            </ibmwcm:value>
         </ibmwcm:element>
         <ibmwcm:element name="Body">
            <ibmwcm:type>rich text</ibmwcm:type>
            <ibmwcm:value>
<![CDATA[<p>Suspendisse posuere commodo turpis.</p>
<p>Vivamus nunc nulla, volutpat in, luctus a, facilisis eu, mi.</p>]]>
            </ibmwcm:value>
         </ibmwcm:element>
         <ibmwcm:element name="Image">
            <ibmwcm:type>image</ibmwcm:type>
            <ibmwcm:source>
               http://cmsserver.ibm.com/images/rel20box.gif
            </ibmwcm:source>
         </ibmwcm:element>
         <ibmwcm:access type="user">[all users]</ibmwcm:access>
      </item>
   </channel>
</rss>
```

RSS Namespace Extension for the Feed Service

The RSS namespace extension is used to exchange control information between the feed producer and consumer applications.

To implement this namespace, specify the <rss> element as follows:

```
<rss version="2.0" xmlns:ibmwcm="http://purl.org/net/ibmfeedsvc/wcm/1.0"
xmlns:ibmfs="http://purl.org/net/ibmfeedsvc/feedsvc/1.0" />
```

The input feeds need to include this namespace only in certain scenarios where control information is going to be passed in the feed itself rather than in the HTTP headers.

"The Handshake Protocol" on page 1983
In many Web Content Integrator implementations the input content feeds are produced by an application that generates them dynamically in response to requests from the feed consumer application. In those cases, it is useful for the consumer to be able to indicate to the producer application which versions of a feed the Consumer has already seen so that the producer can make some intelligent decisions about what to include in the next feed.

"Results Feeds" on page 1984
When feed processing is initiated by using a call to the feed service servlet, the Web Content Integrator responds with an output feed. The first entry in this feed contains status information for the feed as a whole. Each of the subsequent entries in the output feed corresponds to an item that was in the input feed.

These latter entries contain status information about the results of processing each item. Feed producers might use this information to attempt to automatically recover from certain types of errors.

"Channel-level Elements" on page 1984
The following element must be a direct child of the feed's <channel /> element since it applies to the feed as a whole rather than to an individual item in the feed.

"Item-level Elements" on page 1985
The following elements are applied at the <item /> level since they contain information that is specific to each individual feed entry. These elements are only used in the output feed. They have no meaning in the context of an input feed.

The Handshake Protocol:

In many Web Content Integrator implementations the input content feeds are produced by an application that generates them dynamically in response to requests from the feed consumer application. In those cases, it is useful for the consumer to be able to indicate to the producer application which versions of a feed the Consumer has already seen so that the producer can make some intelligent decisions about what to include in the next feed.

The handshake protocol consists of the exchange of two feed attributes: a Last Modified Date, and an arbitrary Entity Tag. The values of both of these attributes are managed exclusively by the producer. The consumer receives the values from the producer, stores them, and passes them back unchanged on the next request. These attributes can be passed in one of two ways:

1. As HTTP header fields:
   - Requests from the consumer contain:

     ```
     If-Modified-Since : {last-modified_value}
     If-None-Match: {etag_value}
     ```

   - Responses from the producer contain:

     ```
     ETag: {etag_value}
     Last-Modified: {last-modified_value}
     ```

2. As elements in the feed and query string parameters:
   - Requests from the consumer are in the form:

     ```
     http://cmsserver.example.org/ProducerApp?etag={etag_value}&lastMod={last-modified_value}
     ```

   - Feeds sent back from the producer contain the following channel-level elements:

     ```
     <lastBuildDate>{last-modified_value}</lastBuildDate>
     <ibmfs:etag>{etag_value}</ibmfs:etag>
     ```

How the handshake works

1. The consumer makes its first request to the producer. This request does not contain any special header fields.
2. The producer receives the request, and since there were no special headers, it responds with a feed that contains items which represent all transactions that have occurred up to that point. Before sending the response it sets the Last-Modified header to the current time and the ETag header to a value that enables the producer identify this specific instance of the feed.

   ```
   Last-Modified: Thu, 7 Sep 2006 12:00:00 GMT
   ETag: ABC0011
   ```

3. The consumer receives the response, processes the feed, and stores the Last-Modified and ETag values only if the feed was processed successfully.
4. The next time the consumer is triggered, it again makes a request to the producer for the transaction feed. This time it sets the If-Modified-Since header to the Last-Modified date it received on the last request and the If-None-Match field to the value of the ETag it received on the last request.

   ```
   If-Modified-Since: Thu, 7 Sep 2006 12:00:00 GMT
   If-None-Match: ABC0011
   ```

5. The producer receives the request, and checks the values of the header fields. It then uses its own internal logic to generate a new feed which only contains changes that occurred since the last feed it sent to this consumer. It sends back a feed with the following header values:

   ```
   Last-Modified: Thu, 7 Sep 2006 13:00:00 GMT
   ETag: ABC0032
   ```

6. The consumer receives the response, processes the feed, and updates its stored values for the Last-Modified date and ETag.

Ideally, if no changes have occurred since the last time the consumer requested the feed, the producer should respond with an HTTP 304 (Not Modified) response code. This will signal the consumer that there are no changes and thus that it does not need to attemp to parse the feed and do the subsequent processing. However, if the producer is unable to implement this feature it won't affect any of the other processing.

The consumer will always send a request containing the Etag label of the last transaction feed it successfully received. In this case the producer application will be responsible for re-sending any entries that might have been missed due to a communication failure between the servers.

Results Feeds:

When feed processing is initiated by using a call to the feed service servlet, the Web Content Integrator responds with an output feed. The first entry in this feed contains status information for the feed as a whole. Each of the subsequent entries in the output feed corresponds to an item that was in the input feed. These latter entries contain status information about the results of processing each item. Feed producers might use this information to attempt to automatically recover from certain types of errors.

Each entry in the output feed has the following general format:

```
<item>
   <title>Results for: [INPUT_TITLE]</title>
   <link>[INPUT_LINK]</link>
   <pubDate>[CREATION_TIME_OF_OUTPUT_FEED_ENTRY]</pubDate>
   <guid permalink="false">[INPUT_GUID]</guid>
   <ibmfs:resultCode>[ OK | WARN | ERROR | FAIL ]</ibmfs:resultCode>
   <!-- 0 .. n resultMsg elements -->
   <ibmfs:resultMsg level="[ WARN | ERROR ]" code="[ERR_CODE]">[MESSAGE_TEXT]</ibmfs:resultMsg>
   <ibmfs:resultMsg level="[ WARN | ERROR ]" code="[ERR_CODE]">[MESSAGE_TEXT]</ibmfs:resultMsg>
   <ibmfs:documentId>[WCM_DOCUMENT_ID]</ibmfs:documentId>
   <description>[LIST_OF_PROGRESS_MESSAGES]</description>
</item>
```

Channel-level Elements:

The following element must be a direct child of the feed's `<channel />` element since it applies to the feed as a whole rather than to an individual item in the feed.

etag

Some producer applications are not able to manipulate the HTTP headers to set the handshake data. To support those applications, there is an alternative method that entails passing the same information directly in the feed by using the RSS `<lastBuildDate>` element and a custom namespace element, `<ibmfs:etag>`.

Table 313. etag element

Element parameters:	**Details for this element:**
Allowable Values	A string value that represents some meaningful label for the specific instance of the feed.
Required Attributes	None
Optional Attributes	None
Required sub-elements	None
Optional sub-elements	None

Example:

```
<ibmfs:etag>ABC0012</ibmfs:etag>
```

Item-level Elements:

The following elements are applied at the `<item />` level since they contain information that is specific to each individual feed entry. These elements are only used in the output feed. They have no meaning in the context of an input feed.

resultCode

Table 314. resultCode element

Element parameters:	**Details for this element:**
Allowable Values	This element has one of the following values depending on what happened when the input feed entry was processed: **OK** The entry was processed completely with no warnings or errors. **WARN** The entry was processed completely but one or more warnings were logged. **ERROR** One or more error messages were logged. The entry might not have been saved in Web Content Manager. **FAIL** The content item could not be saved or updated.
Required Attributes	None
Optional Attributes	None
Required sub-elements	None
Optional sub-elements	None

Example:

```
<ibmfs:resultCode>OK</ibmfs:resultCode>
<ibmfs:resultCode>WARN</ibmfs:resultCode>
```

resultMsg

Table 315. resultMsg element

Element parameters:	Details for this element:
Allowable Values	Each instance of this element contains the details of a warning or error message.
Required Attributes	**level** The value of this attribute indicates the severity of the message. Allowable values are: "WARN" and "ERROR". **code** This attribute should hold the error code that is associated with the message.
Optional Attributes	None
Required sub-elements	None
Optional sub-elements	None

Examples:

```
<ibmfs:resultMsg level="WARN" code="WCM0014">
Default Content GUID does not point to a content item. Default content set to null.
</ibmfs:resultMsg>


<ibmfs:resultMsg level="ERROR" code="WCM0030">
A WCMException was thrown when attempting to move the content.
</ibmfs:resultMsg>
```

documentId

Table 316. documentId element

Element parameters:	Details for this element:
Allowable Values	This element contains the ID that was generated by Web Content Manager.
Required Attributes	None
Optional Attributes	None
Required sub-elements	None
Optional sub-elements	None

Example:

```
<ibmfs:documentId>
com.ibm.workplace.wcm.api.WCM_Content/Connections Announcement/f6f60a00498c0d759ee9ffe695695374/PUBL
</ibmfs:documentId>
```

WebDAV

With WebDAV for IBM WebSphere Portal Express, you can use standard operating system tools to create, modify, and delete web content rather than the standard authoring portlet.

Before you begin

Before you can use WebDAV with web content, you need to set up a WebDAV client. After your client is set up, you can access the web content libraries with WebDAV with the following URL:

`http://server:port/portal_context_root/mycontenthandler/dav/content/libraries/`

For example:

`http://www.example.com:10027/wps/mycontenthandler/dav/content/libraries/`

About this task

By using tools like file system explorers, WebDAV is used to work with your web content items through familiar, everyday actions. Here are a few examples:

- You can create components or presentation templates by dragging a file into a corresponding folder.
- You can run actions on several items at the same time. For example, you can create five images at the same time by dragging five image files into the image component folder. This creates five separate image components, and for each image component the file name is used for the component's name and title.
- Modifying items is also straightforward through a WebDAV client. For example, you can open a presentation template by using your preferred HTML editor, make changes, and then save the changes. The WebDAV client takes care of accessing the web content library, downloading the template, and then uploading the changes.

In addition to modifying the actual content of an item, you can also modify any item's metadata or access control settings by modifying XML files that define the item's metadata and access control characteristics. You can also drag an existing XML file into the appropriate folder, enabling you to easily set the same data for different items.

You can create, modify, or delete the following items: libraries, taxonomies, categories, site areas, folders, components, and presentation templates.

Note: Be aware that the following features are not supported when using WebDAV with web content:

- Content items, except for managing metadata and access control
- Authoring templates, except for managing metadata and access control
- Nested items within site areas
- Server-side copy and move
- Unauthenticated users
- Exporting of web content libraries with WebDAV to be imported into another web content server

When using WebDAV with web content, be aware of the following considerations.

Locked item support

Locking or unlocking an item through WebDAV locks or unlocks the item in Web Content Manager and the JCR database. Because some items are represented by multiple files and folders, locking or unlocking one of these files causes locking or unlocking of the other associated files at the same time. If you lock an item, folders and files that are related to the content of the item, its metadata, and its access control settings are also locked.

Workflow support

There is no representation of a workflow itself in the WebDAV tree, but if a file is part of a workflow and the workflow indicates that the file is in a state that allows users to modify it, WebDAV will allow you to modify the file as well.

File names and file type suffixes

Files representing data items are always named exactly like the corresponding content item. For example if you have an image component that is named `myImage`, the corresponding image file is also named `myImage`, without any suffix indicating the file type, such as `.gif` or `.jpg`. This can sometimes cause a problem when opening the file through WebDAV because the appropriate application for editing the file cannot be chosen automatically. To account for this, you can either rename the component itself to include the file type (for example, `myImage.gif`), or you can manually start the editing application and open the file from within the client.

Missing items

If an item no longer displays or can no longer be modified, this could be due to a changed state for the item in the web content server where the item is stored. For example creating or modifying an item on the server could lead to a changed state that prevents you from accessing this item with WebDAV, depending on how workflow is set up. Expiration is another reason an item's state might change and so affect whether you can access the item with WebDAV.

Configuring a HTTP server front end

When you use an HTTP server front end to work with WebDAV, you need to set **Accept content for all requests** to `true` for the Web server plugin in the WebSphere Integrated Solutions Console under **Web servers** > **webserver1** > **Plug-in properties** > **Request and response**.

"Web content items in the WebDAV tree" on page 1989
The WebDAV tree that contains your web content items begins at the WebDAV root `/libraries/`, which displays all libraries to which you have access. All web content items within the libraries are organized with folders and files.

"Metadata and access control for web content items in WebDAV" on page 1991
WebDAV uses XML files to represent metadata and access control information for a web content item. You can change an item's metadata and access control settings by modifying these files, and you can specify settings for multiple files by copying the XML files to their appropriate locations in the WebDAV tree.

"Creating taxonomies and categories with WebDAV" on page 1993
Taxonomies and categories are profiling methods that are used to group content items, and you can work with taxonomies and categories directly through WebDAV. Taxonomies and categories are represented in WebDAV as folders, and you can set up your taxonomy by creating and nesting folders.

"Managing content with site areas in WebDAV" on page 1994
Site areas are used to organize content items in your web content system. In WebDAV site areas are represented as folders, and you can set up your site structure by creating and nesting folders. A content item within a site area is represented as a folder that contains the metadata and access control settings for the content item.

"Creating components with WebDAV" on page 1995
Components are used to store elements in your web content system, and you can use WebDAV to create and manage components. Each component type is

represented as a folder in WebDAV, with individual components represented as files in the appropriate component folder.

"Creating presentation templates in WebDAV" on page 1997
With WebDAV, you can create and maintain presentation templates to define the layout and appearance characteristics of web pages that are used to display content. You can also create nested image components for use with the presentation templates. Presentation templates are stored in a folder with nested image components in an associated folder.

"Managing metadata and access control settings for authoring templates with WebDAV" on page 1999
With WebDAV, you can change the metadata information for an authoring portlet or update the template's access control settings.

Related tasks:

"Working with WebDAV clients" on page 1358
WebDAV is an HTTP extension framework with a plug point for the access and management of hierarchical data, for example, in content management systems. WebDAV stores the data in collections and allows you to work with the data in a user interface view that is similar to that of a file system. Various tools are available for integrating WebDAV resources into the client file system, known as WebDAV clients. To use WebDAV for WebSphere Portal Express, you must first set up your WebDAV clients.

Web content items in the WebDAV tree

The WebDAV tree that contains your web content items begins at the WebDAV root `/libraries/`, which displays all libraries to which you have access. All web content items within the libraries are organized with folders and files.

Items that do not pertain to content, such as site areas or categories, are represented as folders that contain only the item's metadata folder and any child items like other site areas or categories. No data files are present.

- sites
  - wcm.siteArea.siteArea1
    - meta-data
    - wcm.siteArea.siteArea1.1
      - meta-data
      - wcm.siteArea.siteArea1.1.1
        - meta-data
      - wcm.siteArea.siteArea1.1.2
        - meta-data
    - wcm.siteArea.siteArea2
      - meta-data

Data-oriented items like image components or presentation templates are represented as files so you can manipulate them with drag and drop operations. The corresponding metadata for the item is managed the same way as for the non-data items but in separate subfolders within the metadata folder.

- wcm.comps.image
  - image1.jpg
  - image2.jpg
  - meta-data
    - wcm.comp.image1.jpg

- wcm.comp.image2.jpg

In addition to folders that represent actual web content items, there are folders in the WebDAV tree to structure the data or to allow for better scalability. Each library contains folders for components, presentation templates, sites, and taxonomies.

- libraries
  - wcm.library.my_library
    - authoringTemplates
    - components
    - presentationTemplates
    - sites
    - taxonomies
  - wcm.library.contentlibrary
    - components
    - presentationTemplates
    - sites
    - taxonomies

Within the components folder, there are subfolders for the component types for better scalability and management of the different types of components.

- libraries
  - wcm.library.my_library
    - components
      - wcm.comps.authoring.tools
      - wcm.comps.component.references
      - wcm.comps.data.and.time
      - wcm.comps.federated.content
      - wcm.comps.file
      - wcm.comps.html
      - wcm.comps.image
      - wcm.comps.jsp
      - wcm.comps.link
      - wcm.comps.menu
      - wcm.comps.navigator
      - wcm.comps.number
      - wcm.comps.page.navigation
      - wcm.comps.personalization
      - wcm.comps.richt.text
      - wcm.comps.search
      - wcm.comps.short.text
      - wcm.comps.style.sheet
      - wcm.comps.taxonomy
      - wcm.comps.text
      - wcm.comps.user.name
      - wcm.comps.user.selection

To organize your authoring templates, presentation templates, and components, you can create custom web content folders. These web content folders are represented as folders in WebDAV and contain a set of component types that are structured in the same way as the root `components` folder. Here an example of a component folder structure:

- `CustomComponentFolder1`
  - `wcm.comps.authoring.tools`
  - `wcm.comps.component.references`
  - `wcm.comps.data.and.time`
  - `wcm.comps.federated.content`
  - `wcm.comps.file`
  - `.`
  - `.`
  - `.`
  - `wcm.comps.user.selection`
- `CustomComponentFolder2`
  - `wcm.comps.authoring.tools`
  - `wcm.comps.component.references`
  - `wcm.comps.data.and.time`
  - `wcm.comps.federated.content`
  - `wcm.comps.file`
  - `.`
  - `.`
  - `.`
  - `wcm.comps.user.selection`
- `wcm.comps.authoring.tools`
- `wcm.comps.component.references`
- `wcm.comps.data.and.time`
- `wcm.comps.federated.content`
- `wcm.comps.file`
- `.`
- `.`
- `.`
- `wcm.comps.user.selection`

Metadata and access control for web content items in WebDAV

WebDAV uses XML files to represent metadata and access control information for a web content item. You can change an item's metadata and access control settings by modifying these files, and you can specify settings for multiple files by copying the XML files to their appropriate locations in the WebDAV tree.

Metadata

An item's metadata is represented by the `meta-data.xml` file, which describes identification information for the item, including the name and title for the item and the list of authors and owners that are associated with the item.

Here is a sample `meta-data.xml` file:

```
<?xml version="1.0" encoding="UTF-8"?>
<meta-data>
  <item>
    <title lang="en" value="test1.JPG"/>
    <description lang="en" value="description"/>
    <wcm-group id="authors">
      <member DN="all_auth_portal_users" type="group"/>
      <member DN="uid=wpsadmin,o=defaultWIMFileBasedRealm" type="user"/>
    </wcm-group>
    <wcm-group id="owners">
      <member DN="uid=wpsadmin,o=defaultWIMFileBasedRealm" type="user"/>
      <member DN="all_users" type="group"/>
    </wcm-group>
  </item>
</meta-data>
```

Access control

An item's access control information is represented by the following files:

- `access-control-system.xml`: Contains access control settings for the system that are specified by the administrator.
- `access-control-user.xml`: Contains access control settings that are defined by the user.

In addition to these item-specific files, the `access-control.xml` file is provided for folders that represent resource types, like the `components` folder, and contains access control settings for the resource type.

Here is a sample `access-control.xml` file for resource access control settings:

```
<?xml version="1.0" encoding="UTF-8"?>
<access-control>
  <resource-config>
    <role-block role-type="Editor" type="inheritance"/>
    <role-block role-type="User" type="inheritance"/>
    <role-block role-type="Editor" type="propagation"/>
    <role-block role-type="User" type="propagation"/>
  </resource-config>
  <role-list>
    <role type="Administrator">
      <member DN="uid=wpsadmin,o=defaultWIMFileBasedRealm" type="user"/>
    </role>
    <role type="Contributor">
      <member DN="all_auth_portal_users" type="group"/>
    </role>
    <role type="Manager">
      <member DN="all_auth_portal_users" type="group"/>
    </role>
  </role-list>
</access-control>
```

Here is a sample `access-control-system.xml` file for an item's administrator-defined access control settings:

```
<?xml version="1.0" encoding="UTF-8"?>
<access-control>
  <resource-config>
    <role-block role-type="Contributor" type="inheritance"/>
    <role-block role-type="Manager" type="inheritance"/>
  </resource-config>
  <role-list>
    <role type="Editor">
      <member DN="authors" type="virtual"/>
    </role>
```

```
    <role type="User">
      <member DN="all_auth_portal_users" type="group"/>
    </role>
  </role-list>
</access-control>
```

Here is a sample `access-control-user.xml` file for an item's user-defined access control settings:

```
<?xml version="1.0" encoding="UTF-8"?>
<access-control>
  <role-list>
    <role type="Contributor">
      <member DN="owners" type="virtual"/>
    </role>
    <role type="Editor">
      <member DN="all_auth_portal_users" type="group"/>
    </role>
    <role type="Manager">
      <member DN="uid=wpsadmin,o=defaultWIMFileBasedRealm" type="user"/>
    </role>
  </role-list>
</access-control>
```

Related tasks:

"Managing metadata and access control settings for authoring templates with WebDAV" on page 1999
With WebDAV, you can change the metadata information for an authoring portlet or update the template's access control settings.

Creating taxonomies and categories with WebDAV

Taxonomies and categories are profiling methods that are used to group content items, and you can work with taxonomies and categories directly through WebDAV. Taxonomies and categories are represented in WebDAV as folders, and you can set up your taxonomy by creating and nesting folders.

About this task

All taxonomies and categories for a library are listed under the `taxonomies` folder for that library.

```
taxonomies
   - wcm.taxonomy.taxonomy1
       - meta-data
           access-control-system.xml
           access-control-user.xml
           meta-data.xml
       - wcm.category.category1
           - meta-data
               access-control-system.xml
               access-control-user.xml
               meta-data.xml
           - wcm.category.category1.1
               - meta-data
                   access-control-system.xml
                   access-control-user.xml
                   meta-data.xml
           - wcm.category.category1.2
               - meta-data
                   access-control-system.xml
                   access-control-user.xml
                   meta-data.xml
   - wcm.taxonomy.taxonomy2
       - meta-data
```

```
          access-control-system.xml
          access-control-user.xml
          meta-data.xml
   access-control.xml
```

Procedure

To create a new taxonomy or category for your library, create a new folder with the `wcm.taxonomy` prefix or the `wcm.category` prefix. Taxonomies can be created under the generic `taxonomies` folder only, while categories can be created in either a `wcm.taxonomy.*` folder or a `wcm.category.*` folder.

Important: Some WebDAV clients create a folder with a default name, such as `New Folder`, and as soon as you enter the name of the new folder, the client sends a request to rename the already created folder. Because taxonomy and category folders require a corresponding prefix for creation, this client behavior does not work. If your WebDAV client uses this method to create new folders, you can first create the new taxonomy or category folder locally and then copy it into the WebDAV tree.

Deleting taxonomies and categories: To delete taxonomies or categories delete the corresponding folder. Taxonomies or categories that contain categories cannot be deleted until you have also first deleted the child items. Also if a category is still being referenced by another item, it cannot be deleted until you have first removed the corresponding references by using the authoring portlet.

Managing content with site areas in WebDAV

Site areas are used to organize content items in your web content system. In WebDAV site areas are represented as folders, and you can set up your site structure by creating and nesting folders. A content item within a site area is represented as a folder that contains the metadata and access control settings for the content item.

About this task

All site areas and content items for a library are listed under the `sites` folder for that library.

```
sites
  - wcm.siteArea.siteArea1
      - meta-data
          access-control-system.xml
          access-control-user.xml
          meta-data.xml
      - wcm.siteArea.siteArea1.1
          - meta-data
              access-control-system.xml
              access-control-user.xml
              meta-data.xml
          - wcm.content.content1.1.1
              - meta-data
                  access-control-system.xml
                  access-control-user.xml
                  meta-data.xml
      - wcm.siteArea.siteArea1.2
          - meta-data
              access-control-system.xml
              access-control-user.xml
              meta-data.xml
   access-control.xml
```

Note: Support for content items is limited to modifying the metadata and access control settings. You cannot create or delete content items by using WebDAV.

Procedure

To create a new site area for your library, create a new folder with the `wcm.siteArea` prefix.

Important: Some WebDAV clients create a folder with a default name, such as `New Folder`, and as soon as you enter the name of the new folder, the client sends a request to rename the already created folder. Because site area folders require a corresponding prefix for creation, this client behavior does not work. If your WebDAV client uses this method to create new folders, you can first create the new site area folder locally and then copy it into the WebDAV tree.

Deleting site areas: To delete site areas delete the corresponding folder. Parent site areas containing site areas or content items cannot be deleted until you have also first deleted the child items. To delete child content items before deleting a site area, you must use the authoring portlet rather than WebDAV.

Creating components with WebDAV

Components are used to store elements in your web content system, and you can use WebDAV to create and manage components. Each component type is represented as a folder in WebDAV, with individual components represented as files in the appropriate component folder.

About this task

All components for a library are listed as folders under the `components` folder for that library. Within the `components` folder, you can also create custom folders that you can use to organize your components. Like the root `components` folder, custom folders contain folders for each type of component.

```
libraries
  - wcm.library.my_library
      - components
          - CustomComponentFolder1
              - wcm.comps.authoring.tools
              - wcm.comps.component.references
              - wcm.comps.data.and.time
              .
              .
              .
              - wcm.comps.user.selection
          - CustomComponentFolder2
              - wcm.comps.authoring.tools
              - wcm.comps.component.references
              - wcm.comps.data.and.time
              .
              .
              .
              - wcm.comps.user.selection
          - wcm.comps.authoring.tools
          - wcm.comps.component.references
          - wcm.comps.data.and.time
          - wcm.comps.federated.content
          - wcm.comps.file
          - wcm.comps.html
          - wcm.comps.image
          - wcm.comps.jsp
          - wcm.comps.link
          - wcm.comps.menu
```

```
        - wcm.comps.navigator
        - wcm.comps.number
        - wcm.comps.page.navigation
        - wcm.comps.personalization
        - wcm.comps.rich.text
        - wcm.comps.search
        - wcm.comps.short.text
        - wcm.comps.style.sheet
        - wcm.comps.taxonomy
        - wcm.comps.text
        - wcm.comps.user.name
        - wcm.comps.user.selection
        access-control.xml
```

Components are data-oriented items and represented as files and metadata folders.

```
libraries
  - wcm.library.my_library
      - components
        - wcm.comps.authoring.tools
        - wcm.comps.component.references
        - wcm.comps.data.and.time
        - wcm.comps.federated.content
        - wcm.comps.file
        - wcm.comps.html
        - wcm.comps.image
            image1.jpg
            image2.jpg
            - meta-data
              - wcm.comp.image1.jpg
                    access-control-system.xml
                    access-control-user.xml
                    meta-data.xml
              - wcm.comp.image2.jpg
                    access-control-system.xml
                    access-control-user.xml
                    meta-data.xml
        - wcm.comps.jsp
        - wcm.comps.link
        - wcm.comps.menu
        - wcm.comps.navigator
        .
        .
        .
         - wcm.comps.user.selection
        access-control.xml
```

Important: Although displayed in WebDAV, the following components cannot be created or modified through WebDAV and are represented by empty files:

- Authoring tools
- Component references
- Federated content
- JSP
- Menu
- Navigator
- Page navigation
- Personalization
- Search
- Taxonomy
- User name
- User selection

To change these components, you must use the authoring portlet.

Link component limitation: Currently, link components are not fully supported by WebDAV. The WebDAV file that represents the link component contains only the URL of the link itself but no other information, such as the link text. For example, if you use WebDAV to modify a link component that contains an HTML representation of `<a href='www.lotus.com'>lotus software</a>` and change the URL to`www.ibm.com`, the link text is rendered as `lotus software`, because that information cannot be modified with WebDAV.

Procedure

To create components for your library, drag one or more files into the appropriate component type folder. When you create a new component in this way, the object's file name is used as the name and title of the new component, and the file's content is stored as the component's data. In addition, the user who is authenticated with the WebDAV client is specified as the author and owner of the new component.
For example, you might drag an HTML file into the `wcm.comps.html` folder for a new HTML component or into the `wcm.comps.rich.text` folder for a new rich text element.

Important: Placing an incompatible file into a component type folder (for example, putting a JPEG file into the `wcm.comps.html` folder) can cause errors during component creation and might result in an unusable component.

Updating components: To update an existing component, you can replace the corresponding file in the WebDAV tree with a new file that has the same name. For example you can place `myCoolPic.jpg` into the image components folder that already contains `myCoolPic.jpg`, and the component is automatically updated with the new file's content. If you place a file with a different name, a new component with that name is created.

Creating presentation templates in WebDAV

With WebDAV, you can create and maintain presentation templates to define the layout and appearance characteristics of web pages that are used to display content. You can also create nested image components for use with the presentation templates. Presentation templates are stored in a folder with nested image components in an associated folder.

About this task

All presentation templates for a library are listed under the `presentationTemplates` folder for that library. Because they are data-oriented items, presentation templates are represented as files and `meta-data` folders. Nested image components are stored in a folder that is named after its associated presentation template–for example, *`template_name`*`_files`.

```
libraries
  - wcm.library.my_library
      - presentationTemplates
         template1.html
         myTemplate.html
         - meta-data
             - wcm.presentationTemplate.template1.html
                   access-control-system.xml
                   access-control-user.xml
                   meta-data.xml
             - wcm.presentationTemplate.myTemplate.html
```

```
            access-control-system.xml
            access-control-user.xml
            meta-data.xml
        access-control.xml
    - template1.html_files
        nested_image.jpg
    - myTemplate.html_files
```

Procedure

1. To create presentation templates for your library, drag one or more files into the `presentationTemplates` folder. When you create a new presentation template in this way, the object's file name is used as the name and title of the new template, and the file's content is stored as the template's data. In addition, the user who is authenticated with the WebDAV client is specified as the author and owner of the new template.

   Important: Placing an incompatible file into the `presentationTemplates` folder (for example, a JPEG file) can cause errors during template creation and might result in an unusable presentation template.

   Deleting presentation templates: To delete a presentation template delete the corresponding file. If the presentation template is being referenced by another item, such as a site area, it cannot be deleted until you have first removed the corresponding references by using the authoring portlet.

   Updating presentation templates: To update an existing presentation template, you can replace the corresponding file in the WebDAV tree with a new file that has the same name. For example, you can place `myTemplate.html` into the `presentationTemplates` folder, replacing the `myTemplate.html` file that is already there, and the presentation template will automatically be updated with the new file's content. If you place a file with a different name, a new template with that name is created.
2. Create any nested image components for your presentation template by adding the image files to the *`template_name`*`_files` folder for your template. For example, if your template is `template1.html`, you would add the image files to the `template1.html_files` folder.

   Note: When you add an image to the nested components folder, a temporary image is created initially, and the image is only permanently added to the list of nested components when a reference to that image is added to the presentation template's HTML code. This is done to prevent orphaned components within the presentation template.
3. If you have added a nested image component, update the presentation template's HTML code to reference the component according to the relative WebDAV path to the component.

   For example, to reference a nested image component, you would update the `template1.html` file with the following code:

   ```
   <img src="./template1.html_files/nestedImage.jpg" border="0"
   title="my nested image"/>
   ```

   To reference a standard image component, you would use HTML code similar to the following example:

   ```
   <img src="../components/wcm.comps.image/test2.jpg" border="0"
   title="my test image"/>
   ```

Managing metadata and access control settings for authoring templates with WebDAV

With WebDAV, you can change the metadata information for an authoring portlet or update the template's access control settings.

About this task

All authoring templates for a library are listed under the `authoringTemplates` folder for that library. Because they are data-oriented items, authoring templates are represented as files and `meta-data` folders.

```
libraries
  - wcm.library.my_library
      - authoringTemplates
         auth_template1.html
         myAuthTemplate.html
         - meta-data
             - wcm.presentationTemplate.auth_template1.html
                   access-control-system.xml
                   access-control-user.xml
                   meta-data.xml
             - wcm.presentationTemplate.myAuthTemplate.html
                   access-control-system.xml
                   access-control-user.xml
                   meta-data.xml
             access-control.xml
```

Note: You cannot modify the authoring template itself in WebDAV. To edit the authoring template, use the authoring portlet.

Procedure

1. To change the access control settings for an authoring template, edit the `access-control-system.xml` file for administrator settings or the `access-control-user.xml` file for user-defined settings.
2. To change the metadata for an authoring template, edit the `meta-data.xml` file for the authoring template.

Related concepts:

"Metadata and access control for web content items in WebDAV" on page 1991
WebDAV uses XML files to represent metadata and access control information for a web content item. You can change an item's metadata and access control settings by modifying these files, and you can specify settings for multiple files by copying the XML files to their appropriate locations in the WebDAV tree.

Blogs

Use blogs and blog libraries to provide news and commentary on a variety of subjects pertinent to your intranet and extranet sites. Blogs and blog libraries typically combine text with graphics and links to other blogs and web sites. Entries that you create and post are arranged in reverse-chronological order, with the newest entry displayed first. Readers can post comments about your entries, fostering discussions and online networking. You can manage your own blog entries and comment on other blog entries. You can also incorporate tagging and rating as you would with other WebSphere Portal content.

About this task

Tip: Use a blog for sharing information on a single subject. Use a blog library to share information on multiple discussions.

"Learn about the template libraries used by blogs and blog libraries"
Blogs and blog libraries use the template libraries provided by IBM Web Content Manager. Blog libraries use Web Resources v70 library and Blog v70 library. Blogs use the Web Resources v70 and Blog Solo v70 library. The page hierarchy that is provided for these components is the common one defined by the Web Content Manager template libraries.

"Adding a blog or blog library to a page" on page 2001
With Editor access to the portal or the portal page, you can add a blog or blog library to a page. Choose a blog to collaborate with your team on a single topic. Choose a blog library to collaborate with your team on multiple topics in a centralized view.

"Adding existing blogs or blog libraries to a page" on page 2002
If you created a blog or blog library for another page and now want to use it again, add a web content viewer to the new page and edit its settings to point to the existing blog or blog library.

"Assigning blog access to users" on page 2003
The Portal administrator can assign Editor access to you if you need to create and manage blogs within the site. If you are given Editor access to a blog, you can create or modify posts in that blog. If you are given Editor rights to a blog library, you can create and modify blogs and you can create and modify posts in that blog library. If you have Manager rights to a blog, you can create, modify, and delete posts and delete comments in that blog. If you are given Manager rights to a blog library, you can create, modify, and delete blogs. You can also create, modify, and delete posts and delete comments within the blogs.

"Viewing blogs and blog posts" on page 2004
You can view the blogs that are created in a specific library and you can view the posts in a blog by descending order. If you are a contributor, you can view blog pages. By default, all portal users can view content in a blog or blog post once it has been created.

"Deleting blogs or blog libraries" on page 2004
If you are the owner of a blog site, you have Manager access and can delete any blog, pages, comments, or posts on that site.

Related concepts:

"Tagging and rating portal content" on page 34
Users can tag or rate portal content and view the tags and ratings. Tagging and rating allow users to better organize, categorize, and find portal content. This task includes Web Content Manager, Connections, and custom content. For example, users can tag or rate books in an online bookstore.

Learn about the template libraries used by blogs and blog libraries

Blogs and blog libraries use the template libraries provided by IBM Web Content Manager. Blog libraries use Web Resources v70 library and Blog v70 library. Blogs use the Web Resources v70 and Blog Solo v70 library. The page hierarchy that is provided for these components is the common one defined by the Web Content Manager template libraries.

Note: The Web Content Manager libraries described in this topic only work with the theme customizer used in the default theme. Customization to these libraries will affect all blogs and blog libraries on the site.

Web Resources v70 library

The Web Resources v70 library provides the authoring and presentation templates for blogs and blog libraries. All blogs on the site use this single shared library. Changes to the main shared library affects all blogs.

Blog Solo v70 library

When you add a blog to a page, a copy of the Blog Solo v70 library is created using the name that you provided when adding a blog to a page. This copy of the Blog Solo v70 library is used to store posts and comments.

Blog v70 library

When you add a blog library to a page, a copy of the Blog v70 library is created using the name that you provided when adding the blog library to a page. The copy of the Blog v70 library is used to store posts and comments when new pages are added to the blog library.

Adding a blog or blog library to a page

With Editor access to the portal or the portal page, you can add a blog or blog library to a page. Choose a blog to collaborate with your team on a single topic. Choose a blog library to collaborate with your team on multiple topics in a centralized view.

Before you begin

Refer to the WebSphere Portal family Wiki for an example of how you can modify your custom theme to enable the capability of adding blogs and wikis to a page.

Procedure

1. Create a project or access an existing project, and open the project in edit mode.
2. If you did not already, create a page for the existing blog or blog library or open an existing page.

   Note: Blog and Blog Library require a certain amount of horizontal screen space to render properly, and are not supported on mobile devices. For desktop applications with limited horizontal screen space, use the portlets in a one-column layout.
3. Click the **Content** tab, then click **Communications**.
4. Click **Blog Library** to add a team blog or click **Blog** to create a single-topic blog.
5. Click **Add to page**.
6. Provide a name. Then, click **Add**.

   The name is used for the portlet title, library display title, and library name. It can contain alphanumeric characters, spaces, and the following characters: `$ - _ . ! ( ) ,`. If you use other characters, they are replaced by characters that are supported by the Web Content Manager library.
7. Click **Save Draft**.

   This action copies the appropriate IBM Web Content Manager blog template library and places an instance of the web content viewer on the page.

What to do next

After you publish the site with a blog library, click **Create Blog** and **Create Post** to add content. Click **Edit** to modify content. To delete a post, click **Delete**.

After you publish the site with a blog, click **Create Post** to add content. Click **Edit** to modify content. To delete a post, click **Delete**.

Note: Tagged web content that is displayed in the web content viewer is only available when there is a single instance of this portlet on the page. When you click a tag result, the Tag Center broadcasts the information on what content displays in the viewer with a public render parameter. If you have multiple instances of web content displayed in the web content viewer, then these instances display the content that you tagged rather than display the original content of these instances.

Adding existing blogs or blog libraries to a page

If you created a blog or blog library for another page and now want to use it again, add a web content viewer to the new page and edit its settings to point to the existing blog or blog library.

Procedure

1. Create a project or access an existing project, and open the project in edit mode.
2. If you did not already, create a page for the existing blog or blog library or open an existing page.

   Note: Blog and Blog Library require a certain amount of horizontal screen space to render properly, and are therefore not intended for small mobile devices such as smartphones. For some tablet and desktop situations with limited horizontal screen space, use the portlets in a one-column layout.
3. Click the **Content** tab.
4. Click **All** in the list of categories, and search for the portlet **Web Content Viewer**.
5. Add the web content viewer to your page by clicking the plus sign (+) or dragging the viewer onto the page.
6. Click **Save**.
7. Click the drop-down menu that is in the web content viewer title bar, and select **Edit Shared Settings**.
8. Under Content Type, select **Content Item**.
9. Under Content, click **Edit**.
10. Click **Libraries**.
11. Click the library that contains the blog or blog library you want to add.
12. Expand the selected library to see the blog or blog library you want to use.
13. Select the blog or blog library and click **OK**.
14. Click **Apply** to save your changes to the web content viewer.
15. Click **Advanced Options**.
16. Ensure the **Broadcast links to** option is set to **None** or **This Portal page**.
17. Ensure the **Receive links from** option is set to **This portlet only** or **Other portlets and this portlet**.
18. Click **OK**.

What to do next

Note: Tagged web content that is displayed in the web content viewer is only available when there is a single instance of this portlet on the page. When you click a tag result, the Tag Center broadcasts the information on what content displays in the viewer with a public render parameter. If you have multiple instances of web content that is displayed in the web content viewer, then these instances display the content that you tagged rather than display the original content of these instances.

Assigning blog access to users

The Portal administrator can assign Editor access to you if you need to create and manage blogs within the site. If you are given Editor access to a blog, you can create or modify posts in that blog. If you are given Editor rights to a blog library, you can create and modify blogs and you can create and modify posts in that blog library. If you have Manager rights to a blog, you can create, modify, and delete posts and delete comments in that blog. If you are given Manager rights to a blog library, you can create, modify, and delete blogs. You can also create, modify, and delete posts and delete comments within the blogs.

About this task

For an example of assigning access to users, refer to the following instructions for adding users to the Editor role:

Procedure

1. Click the **Administration menu** icon. Then, click **Portal Content** > **Web Content Libraries**.
2. Go to the library that contains the blog you want to manage and click **Set permissions**.
3. Click **Edit Role** for the Editor role.
4. Click **Add** to assign users or groups to the Editor role. Search for the users or groups that belong to this role.

Example

For example, the settings in the following table allow all authenticated users to create new blogs, but restricts them to posting only in blogs they created. Use the **Set Permissions** and **Library Resources** options from the Web Content Library page to assign the following settings:

Table 317. An example of access settings for a blog

Access settings	Group	Access level	Other settings
Presentation Templates	All Authenticated Portal Users	Contributor	Do not select "Allow propagation"
Presentation Templates	All Authenticated Portal Users	User	
Content	All Authenticated Portal Users	Editor	

Viewing blogs and blog posts

You can view the blogs that are created in a specific library and you can view the posts in a blog by descending order. If you are a contributor, you can view blog pages. By default, all portal users can view content in a blog or blog post once it has been created.

Procedure

1. Navigate to the page that contains the blog or blog library.
2. **(Blog library only)** click **List of Blogs** to view all of the blogs that were created in a specific library. Blogs are listed in descending order from first to last and display the name of the person who created the blog (Editor) and the date and time when the blog was created.
   a. Move through the list of blogs by using the page controls.
   b. Specify how many blogs to list on a page by choosing to **Show** *10*, *20*, *50* *100*.
   c. If you have Editor access, you can click **Create Blog**.
   d. Click the title of a blog to open the blog.
3. **(Blog or Blog Library)** View the posts, which are listed in descending order from first to last and display the name of the person who created the post (User) and the date and time when the post was created.
   a. Click **Latest Blog Posts** to view the most recent blog content.
   b. Move through the list of posts by using the page controls.
   c. Specify how many posts to list on a page by choosing to **Show** *10*, *20*, *50*, or *100*.
   d. If you have Editor access, you can **Create Post**. You can edit and delete posts that you own.
   e. If you have User access, you can comment on a post.

What to do next

After you view posts, you can add comments to share your thoughts. To add a comment, click **Add a Comment**.

Deleting blogs or blog libraries

If you are the owner of a blog site, you have Manager access and can delete any blog, pages, comments, or posts on that site.

About this task

To delete an individual blog, open the project that contains the blog library in a project. To delete an entire blog library, use the Portal Administration page.

Procedure

1. To delete one blog from a library, go to the page that contains the blog library.
2. Click **List of Blogs**.
3. Go to the blog library that contains the blog you want to delete.
4. Click **Delete** and **OK**.
5. You can delete an entire blog library. Click the **Administration menu** icon. Then, click **Portal Content** > **Web Content Libraries**.
6. Go to the library you want to delete and click **Delete library** and **OK**.

Wikis

Use wikis to share community content on a variety of subjects pertinent to your intranet and extranet sites. Wikis typically combine text with graphics and links to other wikis and web sites. You can monitor and manage your own wiki articles.

About this task

"Learn about the template libraries used by wikis"
Wikis use the template libraries provided by IBM Web Content Manager. Wikis use the Web Resources v70 library and the Wiki v70 library. The page hierarchy that is provided for wikis is the common one defined by the Web Content Manager template libraries.

"Adding a wiki to a page" on page 2006
With Editor access to the portal or the portal page, you can add a wiki to a page to quickly create and edit content in-line.

"Adding existing wikis to a page" on page 2007
If you created a wiki for another page and now want to use it again, add a web content viewer to the new page and edit its settings to point to the existing wiki.

"Assigning wiki access to users" on page 2008
If you are the administrator, you have Manager access and can assign Editor access to other users who need to create and manage wikis within the site. If you are the owner of a wiki site, you have Manager access. As wiki site Manager, you can also delete any wiki page or wiki. If you have Editor access, you can create and edit any wiki site and wiki pages. All wiki editors can modify all pages of a wiki site.

"Deleting wikis" on page 2008
If you are the owner of a wiki site, you have Manager access and can delete any wiki or wiki pages on that site.

"Purging deleted wiki pages" on page 2008
When you delete a wiki post, it is removed from the site, but remains in the delete view in IBM Web Content Manager. Users with Administrator access to libraries can purge these deleted wiki pages, which optimizes performance of the wiki library. Purging the deleted wiki pages removes all occurrences, including all versions. You cannot restore deleted items after you purge them.

Related concepts:

"Tagging and rating portal content" on page 34
Users can tag or rate portal content and view the tags and ratings. Tagging and rating allow users to better organize, categorize, and find portal content. This task includes Web Content Manager, Connections, and custom content. For example, users can tag or rate books in an online bookstore.

Learn about the template libraries used by wikis

Wikis use the template libraries provided by IBM Web Content Manager. Wikis use the Web Resources v70 library and the Wiki v70 library. The page hierarchy that is provided for wikis is the common one defined by the Web Content Manager template libraries.

Note: The Web Content Manager libraries described in this topic only work with the theme customizer used in the default theme. Customization to these libraries affects all wikis on the site.

Web Resources v70 library

The Web Resources v70 library provides the authoring and presentation templates for wikis. All wikis on the site use this single shared library. Changes to the main shared library affects all wikis.

Wiki template

When you add a wiki to a page, a copy of the Wiki v70 library is created using the name that you provided when adding the wiki to a page. The copy of the wiki v70 library is used to store wiki page content when new pages are added to this wiki.

Adding a wiki to a page

With Editor access to the portal or the portal page, you can add a wiki to a page to quickly create and edit content in-line.

Before you begin

Refer to the WebSphere Portal family Wiki for an example of how you can modify your custom theme to enable the capability of adding blogs and wikis to a page.

Procedure

1. Create a project or access an existing project, and open the project in edit mode.
2. If you did not already, create a page for the wiki or open an existing page.

   Note: Wiki requires a certain amount of horizontal screen space to render properly, and is not supported for mobile devices. For desktops applications with limited horizontal screen space, use the portlet in a one-column layout.
3. Click the **Content** tab, then click **Communications**.
4. Click **Wiki**.
5. Click **Add to page**.
6. Provide a name. Then, click **Add**.

   The name is used for the portlet title, library display title, and library name. It can contain alphanumeric characters, spaces, and the following characters: `$ - _ . ! ( ) ,`. If you use other characters, they are replaced by characters that are supported by the IBM Web Content Manager library.
7. Click **Save Draft**.

   This action copies the appropriate Web Content Manager wiki template library and places an instance of the web content viewer on the page.

What to do next

After you publish the site, click **New Page** to add content or click **Edit** to modify content. To delete a wiki page, click **Delete**.

Note: Tagged web content that is displayed in the web content viewer is only available when there is a single instance of this portlet on the page. When you click a tag result, the Tag Center broadcasts the information on what content displays in the viewer with a public render parameter. If you have multiple instances of web content that is displayed in the web content viewer, then these instances display the content that you tagged rather than display the original content of these instances.

Related concepts:

"Web Content Viewer best practices and limitations" on page 2086
View some best practices and limitations for using Web Content Viewers.

Adding existing wikis to a page

If you created a wiki for another page and now want to use it again, add a web content viewer to the new page and edit its settings to point to the existing wiki.

About this task

Refer to the following steps only when you are adding a web content viewer portlet from the **Page Properties** menu and configuring this web content viewer to point to an existing wiki. To create a new wiki, use the Customize link to add a wiki to a page.

Procedure

1. Create a project or access an existing project, and open the project in edit mode.
2. If you did not already, create a page for the wiki or open an existing page.

   Note: Wiki requires a certain amount of horizontal screen space to render properly, and is not intended for small mobile devices such as smartphones. For some tablet and desktop situations with limited horizontal screen space, use the portlet in a one-column layout.
3. Click the **Content** tab.
4. Click **All** in the list of categories, and search for the portlet **Web Content Viewer**.
5. Add the web content viewer to your page by clicking the plus sign (+) or dragging the viewer onto the page.
6. Click **Save**.
7. Click the drop-down menu that is in the web content viewer title bar, and select **Edit Shared Settings**.
8. Under Content Type, select **Content Item**.
9. Under Content, click **Edit**.
10. Click **Libraries**.
11. Click the library that contains the wiki you want to add.
12. Expand the selected library to find the wiki you want to use.
13. Select the wiki and click **OK**.
14. Click **Apply** to save your changes to the web content viewer.
15. Click **Advanced Options**.
16. Ensure the **Broadcast links to** option is set to **None** or **This Portal page**.
17. Ensure the **Receive links from** option is set to **This portlet only** or **Other portlets and this portlet**.
18. Click **OK**.

What to do next

Note: Tagged web content that is displayed in the web content viewer is only available when there is a single instance of this portlet on the page. When you click a tag result, the Tag Center broadcasts the information on what content displays in the viewer with a public render parameter. If you have multiple

instances of web content that is displayed in the web content viewer, then these instances display the content that you tagged rather than display the original content of these instances.

Related concepts:

"Web Content Viewer best practices and limitations" on page 2086
View some best practices and limitations for using Web Content Viewers.

Assigning wiki access to users

If you are the administrator, you have Manager access and can assign Editor access to other users who need to create and manage wikis within the site. If you are the owner of a wiki site, you have Manager access. As wiki site Manager, you can also delete any wiki page or wiki. If you have Editor access, you can create and edit any wiki site and wiki pages. All wiki editors can modify all pages of a wiki site.

About this task

If you are a contributor, you can view wiki pages. By default, all portal users can view content in a wiki after it is created.

For an example of assigning access to users, refer to the following instructions for adding users to the Editor role:

Procedure

1. Click the **Administration menu** icon. Then, click **Portal Content** > **Web Content Libraries**.
2. Go to the library that contains the wiki you want to manage and click **Set Permissions**.
3. Click **Edit Role** for the **Editor** role.
4. Click **Add** to assign users or groups to the Editor role. Search for the users or groups that belong to this role.

Deleting wikis

If you are the owner of a wiki site, you have Manager access and can delete any wiki or wiki pages on that site.

About this task

You can delete a wiki and all its pages by deleting the wiki from a project.

Procedure

1. Create a project or access an existing project, and open the project in edit mode.
2. Open the page that contains the wiki.
3. From the site toolbar, click **More** > **Delete Page**.
4. Click **OK**.
5. Publish the project.

Purging deleted wiki pages

When you delete a wiki post, it is removed from the site, but remains in the delete view in IBM Web Content Manager. Users with Administrator access to libraries can purge these deleted wiki pages, which optimizes performance of the wiki library. Purging the deleted wiki pages removes all occurrences, including all versions. You cannot restore deleted items after you purge them.

Procedure

1. Select **Applications** > **Content** > **Web Content Management**.
2. In the Library Explorer, select the wiki library that contains the pages that you want to purge.

   Tip: If you do not see your library, click **Preferences** > **Configure** > **Library Selection** to add the library to the selection list.
3. Select **Item Views** > **Deleted Items**.
4. Select the appropriate wiki page and click **Purge**. Click **OK**.

Installed portlets

Learn about the portlets that are provided with WebSphere Portal Express.

You find all portlets that are provided with the portal under the directory *PortalServer_root*. Portlets that are already deployed by the WebSphere Portal Express installation are in the directory *wp_profile_root*/installedApps directory.

In addition to the portlets listed here, WebSphere Portal Express also provides several portlets that are used for administering the portal server. You find these portlets with the Administration portlets.

For the most up-to-date information on portlets, including the portlets available for download, go to the IBM Collaboration Solutions Catalog.

"Business portlets"
Business portlets are automatically installed during the WebSphere Portal Express installation. You can place and display them on a page after you install and start the portal.

Related information:

IBM WebSphere Portal Business Solutions Catalog

Business portlets

Business portlets are automatically installed during the WebSphere Portal Express installation. You can place and display them on a page after you install and start the portal.

All business portlets that are provided with WebSphere Portal Express are in the following directory: *PortalServer_root*/bp. To make the portlets available, click the **Administration menu** icon. Then, click **Portlet Management** > **Portlets**.

Most of the business portlets are also available on the IBM Collaboration Solutions Catalog.

IBM i: Use the **Install and Configure** option of the remote GUI installation to install the portlets.

Table 318. Business portlets available for immediate use

Application	Description
Welcome	Shows version and copyright for WebSphere Portal Express installed.

Table 318. Business portlets available for immediate use (continued)

Application	Description
Banner Ad	Banner Ad enables users to insert and view an image on a portal page. This image can be hyperlinked to a web page. This web page can be specified to open either in the same or a new browser window. **Note:** The Banner Ad portlet verifies that you entered a URL for a particular image. It does not verify the validity of the URL.
Microsoft Exchange 2010 portlet application	The Microsoft Exchange 2010 portlet application connects to an Exchange server. This portlet application contains the following portlets: • MS Exchange 2010 Mail Portlet • MS Exchange 2010 Calendar Portlet
IBM Syndicated Feed Portlet for WebSphere Portal	Use this portlet to integrate, view, and manage RSS and Atom feeds from your portal pages. You can organize the feeds into new and existing feed categories and extensively customize the presentation style for these feeds. The portlet is supported in a proxied environment and enables dynamic modification of the portlet window title. As an Administrator, you can configure several administrative settings from within the portlet interface and apply selective locks on a user's ability to customize the portlet. The portlet complies with the Outline Processor Markup Language (OPML) standards for subscription management. It supports the export of feeds to an OPML-compliant file. It supports the import of multiple feeds into the portlet from an OPML URL.

Note: RSS portlet and IBM Feed Reader portlet are no longer included with WebSphere Portal Express. If you used these portlets, use IBM Syndicated Feed Portlet for WebSphere Portal. The Web2Bookmarks portlet is no longer available for immediate use. You can download the RSS, IBM Feed Reader, and Web2Bookmarks portlets from the IBM WebSphere Portal Business Solutions Catalog.

"Emailing with the Microsoft Exchange 2010 portlet application" on page 2011
By using the IBM portlet application for Microsoft Exchange 2010, users can perform routine mail-related tasks, such as composing, reading, and deleting emails. This application enables users to view, edit, and delete items in their mail and other personal Information Management applications. Administration users set up the portlets and the users who are registered with Exchange can personalize them.

Related information:

IBM WebSphere Portal Business Solutions Catalog

Catalog documentation for IBM Portlet Application for Microsoft Exchange

Emailing with the Microsoft Exchange 2010 portlet application:

By using the IBM portlet application for Microsoft Exchange 2010, users can perform routine mail-related tasks, such as composing, reading, and deleting emails. This application enables users to view, edit, and delete items in their mail and other personal Information Management applications. Administration users set up the portlets and the users who are registered with Exchange can personalize them.

About this task

The IBM portlet application for Microsoft Exchange connects to an Exchange 2010 server via Exchange Web Services.

The following portlets are provided in the portlet application that works with Microsoft Exchange 2010 server:

The IBM Microsoft Exchange Mail 2010 portlet
With Microsoft Exchange2010 Mail, you can connect to an Exchange 2010 Server. You can perform standard mail functions, such as compose, reply, forward, and delete mail.

The IBM Microsoft Exchange Calendar 2010 portlet
With Microsoft Exchange2010 Calendar, you can retrieve calendar information from the Exchange 2010 server and display it on your page. You can perform standard calendar functions, such as creating, editing, deleting, and viewing appointments.

Integration with Exchange OWA mode is not supported for the Exchange 2010 version.

"Setting the credential slot for the Microsoft Exchange 2010 portlet application"
For users to be able to use the IBM portlet application for Microsoft Exchange 2010 in WebSphere Portal Express, administrators must set up a vault slot for the portlets to use.

"Configuring your portal to accept the Exchange server SSL certificate" on page 2012
To use the secure socket layer (SSL) features, you must configure the JVM for the WebSphere Portal Express server to accept the SSL certificate of the Exchange server.

"Customizing the Exchange 2010 portlet" on page 2012
Each user that is registered with Exchange can customize an Exchange 2010 portlet by using the **Personalize** menu. Use the following list to set the appropriate configuration parameters.

Setting the credential slot for the Microsoft Exchange 2010 portlet application:

For users to be able to use the IBM portlet application for Microsoft Exchange 2010 in WebSphere Portal Express, administrators must set up a vault slot for the portlets to use.

Procedure

1. Go to the Credential Vault administration portlet.
2. Create a slot for use by the IBM portlet application for Microsoft Exchange 2010. This slot might not be a shared system credential.

3. Configure each portlet in the web module by placing the portlets on a page and click the configure icon. The configuration window shows a list of possible credential vault slots that you can use.

   Note: The **Configure** icon is only available to users who have Manage permission for the portlet.
4. Select the vault slot that you created for the Microsoft Exchange 2010 portlet application.
5. Click **OK**.
6. Repeat these steps for each portlet. If you select the same slot for each portlet, the users have the same credentials for each portlet.

Configuring your portal to accept the Exchange server SSL certificate:

To use the secure socket layer (SSL) features, you must configure the JVM for the WebSphere Portal Express server to accept the SSL certificate of the Exchange server.

About this task

You must obtain the certificate from the Exchange server. There are different methods to obtain the certificate. The best way is to ask the Exchange administrator to send it to you.

To make the Exchange Server SSL certificate available to the portal server, add the base 64-encoded ASCII data certificate by using the WebSphere Integrated Solutions Console.

Procedure

1. Open the WebSphere Integrated Solutions Console.
2. Click **Security** > **SSL certificates and key management**.
3. Under Related Items, click **Key Stores and Certificates**.
4. Click **NodeDefaultTrustStore**.
5. Under Additional Properties, click **Signer Certificates**.
6. Click **Add**.
7. In the **Alias** field, enter a name.
8. In the **File name** field, enter the path and name of the certificate file.
9. For the **Data type**, select **Bas64-encoded ASCII data** from the pull-down list.
10. Click **Apply**, then **OK**.
11. Restart server1 and your WebSphere Portal Express.

Customizing the Exchange 2010 portlet:

Each user that is registered with Exchange can customize an Exchange 2010 portlet by using the **Personalize** menu. Use the following list to set the appropriate configuration parameters.

About this task

Select version
Select the version of Exchange 2010 that you want to use with Microsoft Mail.

Select your time zone
From the drop-down menu, select the time zone in which you are located.

Server name
Enter the name of the Exchange 2010 Server.

Use a Secure Connection
Select this option to enable a secure connection.

User name
Enter the user name for Exchange 2010. For example, enter `jsmith`.

Password
Enter the password for the Exchange 2010 user name.

Mailbox alias
Enter your primary email address. For example, enter `jsmith@example.com`.

Domain
Enter the domain of your Exchange 2010 account that you received from your Exchange 2010 administrator.

Exchange Email address
Enter your Exchange 2010 email address.

Enter displayed date format
Enter the format in which you want to have the date display.

Number of attachments per message
From the drop-down menu, select the maximum number of attachments that a user can add to a message.

Messages per page
Set the number of messages you want to display.

Renditions

Renditions are different versions of an image component or element. Renditions can be thumbnails or smaller versions of an image formatted for mobile devices.

Renditions make it easy to reformat an image for publishing different sizes on different platforms. Users can choose the size of image they want with the user interface.

The image rendition exists in addition to the original image that is stored in the image component or element as binary code. The image rendition is part of the encapsulating image, which has the same UUID, but can be addressed by a unique URL.

The user interface supports uploading three renditions in addition to the original image. The renditions are named `Desktop`, `Tablet`, and `Smartphone`. You can expose the renditions through the user interface, or through a public API.

CF04 "Setting rendition properties on images" on page 2014
Renditions automatically render a version of your image that is best suited for the device connecting to the site.

CF04 "Disabling renditions" on page 2014
Renditions are enabled by default. If you want to disable them, you must run a ConfigEngine task.

CF04 "Enabling renditions" on page 2015
Renditions are enabled by default. If you disable them, you must run a ConfigEngine task to enable them.

Setting rendition properties on images

Renditions automatically render a version of your image that is best suited for the device connecting to the site.

Procedure

1. Select an image and open the rendition section in the image form.
2. To upload a version of the image that is sized for your purpose, select the rendition, such as **desktop**. You can choose from **desktop**, **tablet**, **smartphone**, or **none**. You can upload a main image, but it is not required if you have an image that is uploaded into the default rendition spot.
3. You can replace a rendition to scale and crop the image for each device size.
4. In the authoring template for image elements, you can disable renditions for an element or set preferred pixel or file sizes in the properties section.
5. When you are finished, click **Place Rendition**.

Results

For component and element tags that reference an image the attribute `rendition="auto"` renders the rendition based on the current device class, such as tablet. If that rendition is empty it renders the default rendition desktop per default, which is defined with `default.rendition.name` in the Web Content Manager Config Service Resource Environment Provider. If that is empty, it renders the main image.

For component and element tags that reference an image with the rendition attribute set to either **desktop**, **tablet**, or **smartphone** the specified rendition is rendered and no fallback is performed. Specifying `rendition="none"` renders the main image.

CF04 If you do not specify the rendition attribute, the main image renders.

CF06 If you do not specify the rendition attribute, `rendition="auto"` is the default.

Disabling renditions

Renditions are enabled by default. If you want to disable them, you must run a ConfigEngine task.

Procedure

1. Go to the *wp_profile_root*`/ConfigEngine` directory.
2. Run the following command:
   - Linux : `./ConfigEngine.sh disable-renditions -DPortalAdminPwd=`*password* `-DWasPassword=`*password*
   - IBM i: `ConfigEngine.sh disable-renditions -DPortalAdminPwd=`*password* `-DWasPassword=`*password*
   - Windows: `ConfigEngine.bat disable-renditions -DPortalAdminPwd=`*password* `-DWasPassword=`*password*
3. Restart the WebSphere Portal Express server.

Enabling renditions

Renditions are enabled by default. If you disable them, you must run a ConfigEngine task to enable them.

About this task

Running this task sets **renditions.enabled=true** as a global property for all virtual portals in the Web Content Manager Config Resource Environment Provider Service.

Procedure

1. Go to the *wp_profile_root*/ConfigEngine directory.
2. Run the following command:
   - Linux : `./ConfigEngine.sh enable-renditions -DPortalAdminPwd=`*password* `-DWasPassword=`*password*
   - IBM i: `ConfigEngine.sh enable-renditions -DPortalAdminPwd=`*password* `-DWasPassword=`*password*
   - Windows: `ConfigEngine.bat enable-renditions -DPortalAdminPwd=`*password* `-DWasPassword=`*password*
3. Restart the WebSphere Portal Express server.

Video start and end points

File components or elements that have a video file that is stored in an HTML5 video format, which includes MP4, webM, or ogg, are stored in Brightcove and have a Brightcove ID.

CF04

The video can be previewed in the content form and you can set starting and ending points for the video.

The following code is an HTML5 video player sample:

```
<video class="videoPlayer" controls="" preload="metadata" src='[Element context="current"
type="content" key="file" format="url"]#t=[Element context="current" type="content" key="file"
format="startTime_sec"],[Element context="current" type="content" key="file" format="endTime_sec"]
         Your browser does not support the video tag.
   </video>
```

The starting and ending points can be retrieved with the component or element tag and the parameter **format="startTime"** or **format="endTime"**, which return the time in seconds. **startTime_msec** and **endTime_msec** return the time in milliseconds.

The starting point frame is also used to generate the thumbnail of the video. The thumbnail can be retrieved with the parameter **rendition="thumbnail"**.

Limitation

The interactive cue point preview player in the content authoring form uses HTML 5 Media Fragment Identifiers for setting the start and end points of the video in the preview frame. Internet Explorer version 11 and earlier does not support these identifiers.

Delivering web content

The type of delivery method you use to deliver web content to your viewers depend on the type of content that is being delivered, and the type of viewers your website is intended for.

"Delivering web content on a portal page"
Using tools like Web Content Viewer Portlets, content associations, and web content page templates, you can build portal pages and display web content. You can also combine web content with other portlet-based content. Content associations tie viewers and portal pages to the site structure of your web content libraries.

"Access web content by using a servlet" on page 2087
Users can access content that is displayed by using the Web Content Manager servlet by connecting to a URL. A servlet delivered website is used when you don't need to use any WebSphere Portal based features such as authoring tools.

"Pre-rendered delivery" on page 2089
You can pre-render a complete IBM Web Content Manager site into HTML and save it to disk. The pre-rendered site can then be used as your live site and displayed to users that use either Web Content Manager or a web server. You deploy a pre-rendered site when you are not using any WebSphere Portal features and your content is static and is only updated periodically.

CF07 "Rendering modes for web content" on page 2093
Different presentation templates are created to render web content in different modes to display content in different contexts, such as a web content viewer portlet, or mobile devices.

Delivering web content on a portal page

Using tools like Web Content Viewer Portlets, content associations, and web content page templates, you can build portal pages and display web content. You can also combine web content with other portlet-based content. Content associations tie viewers and portal pages to the site structure of your web content libraries.

About this task

You can customize your portal delivery as well. Examples include:

- Creating your own web content page templates
- Deploying pre-configured Web Content Viewers
- Providing convenience features like customized error messages and friendly URLs

"Getting started with delivering web content on a portal page" on page 2017
The building blocks for delivering web content in a portal are web content viewers, web content page templates, and content mappings. These pieces provide a flexible framework that you can use to quickly assemble pages.

"Displaying content with Web Content Viewers" on page 2034
Display content from your web content system by adding a Web Content Viewer to the server where you want the content to show.

"Customizing web content delivery" on page 2036
Although web content viewers and page templates provide the basis for delivering web content, you can customize the environment to provide users with a better experience.

"Enabling remote rendering with WSRP and the Web Content Viewer" on page 2069

To display web content on a portal that does not include IBM Web Content Manager, you can use the Web Content Viewer and the WSRP support in the portal. The Web Content Viewer can then retrieve and display content from a web content system on a different server.

"Advanced administrative examples" on page 2074
You can undertake advanced administration for web content artifacts, such as web content pages and content associations.

"Web Content Viewer best practices and limitations" on page 2086
View some best practices and limitations for using Web Content Viewers.

Getting started with delivering web content on a portal page

The building blocks for delivering web content in a portal are web content viewers, web content page templates, and content mappings. These pieces provide a flexible framework that you can use to quickly assemble pages.

About this task

To help you get started, sample web content is also included in preinstalled libraries. The sample content demonstrates how the pieces work, and you can also adapt the sample content for your own use.

"Web Content Viewers"
Web Content Viewers are portlets that render content from a web content library as part of a portal page. If your presentation is simple, a single viewer can be sufficient. To provide a richer experience for your users, use multiple viewers to aggregate content from different libraries.

"Web content pages and templates" on page 2021
Web content pages are portal pages that are associated with content that is managed in IBM Web Content Manager. Similar to web content pages, web content templates are page templates that are associated with content in Web Content Manager.

"Web content associations" on page 2023
Web content associations are used to combine portal pages and associated web content items that are managed by IBM Web Content Manager so they can be managed and rendered consistently. Web content associations map portal pages to the site structure in the IBM Web Content Manager system.

"Creating content with sample web content template items" on page 2026
To illustrate how page templates, web content viewers, and content associations work together, IBM Web Content Manager provides sample web content. The sample content includes examples of web content template pages and predefined portlets that you can add to pages to render content.

"Link examples for Web Content Viewers" on page 2032
Web Content Viewers can broadcast and receive links to communicate with other viewers. Depending on the link settings that you use with the viewers, the behavior of the viewers can be different. These examples demonstrate how different broadcast and receive settings can affect what a viewer renders.

Web Content Viewers:

Web Content Viewers are portlets that render content from a web content library as part of a portal page. If your presentation is simple, a single viewer can be sufficient. To provide a richer experience for your users, use multiple viewers to aggregate content from different libraries.

How viewers locate content

When you add a Web Content Viewer to a web content page, the viewer locates the content to be rendered by evaluating several pieces of information:

- The portlet configuration settings for the viewer can identify the default content to be rendered when a user browses to a page that contains the viewer.
- If the viewer does not specify default content, it determines whether a default content association is defined for the page. If a content association exists, the viewer renders the default content of the referenced site area in IBM Web Content Manager.
- If the current request contains a public render parameter (**path-info** or **context**), the viewer renders the content that is identified by the render parameter. This setting overrides the content setting from the portlet configuration or from the content association on the page. An example of a case where a render parameter might be involved is when users click a link to a content item.

When content is rendered, a Web Content Viewer checks first for a render parameter on the request. As shown in Table 1, if no parameter exists, the viewer evaluates its own portlet configuration and any content association on the page that contains the viewer.

Table 319. How Web Content Viewers determine which content to render

Content reference in portlet configuration?	Content association on web content page?	Content that is rendered by viewer
Yes	No	Content that is identified by portlet configuration of the viewer.
No	Yes	Content that is identified by association on page.
Yes	Yes	Content that is identified by portlet configuration of the viewer. The content that is specified in the portlet configuration can use a relative path. In this case, the complete path to the target content is derived by combining the associations on the page and the viewer.

References to content can be direct or relative:

Direct path to the target content

A Web Content Viewer can reference either a site area or a specific item in the library. This example is a reference to the item `Article` in the site area `Articles` of the library `Web Content`:

```
Web Content/Articles/Article
```

When you configure the viewer with a direct path to content, the content association on the page that contains the viewer is ignored.

To configure a direct path to content, use the **Select content and path** setting in the **Content behavior** settings of the Web Content Viewer.

Relative path to the target content

When a content path based on a relative path is derived, the viewer appends the content path in its configuration to the content association on the page. For example:

- Content association on web content page: `Web Content/Articles`
- Content that is referenced in portlet configuration: `Article`
- Resolved content path: `Web Content/Articles/Article`

To configure a relative path to content, use the **Select content and use the content association of current page** setting in the **Content behavior** settings of the Web Content Viewer.

Depending on how you reference content, you can create Web Content Viewers and web content pages that are as specific or generic as you require.

Reference no content

You can define a content association on a web content page and then add a generic viewer that references no content. The viewer detects the site area that is defined by the content association on the page and renders content from the site area. No additional configuration of the Web Content Viewer is required.

Reference specific content

You can configure a viewer to point to a specific piece of content. You can then add this viewer to any web content page to render the mapped content, regardless of any content association on the page.

Reference content with a relative path

If you are using a consistent site structure with your web content libraries, you can take advantage of the relative path capability for referencing content. For example, you can create a reusable viewer that can render different content depending on the web content page where the viewer is deployed. By defining a viewer with a content association that uses a relative path, you can add instances of that viewer to different pages. The viewers then render different content, according to the content associations on the pages.

You can use one content association on a page and then add multiple Web Content Viewers on the page. In this case, the Web Content Viewers are configured to use a relative path into different site areas in the library. By changing only the content association on the page, you can then redirect the viewers to another library or other set of site area content.

Create content with Web Content Viewers

When added to a page, a viewer can create a copy of the content that is identified in the portlet configuration. This feature provides several advantages:

- You can create content items quickly and easily, within the scope of the page where they are used.
- You can modify the individual copies of the content independently of each other.
- Typically, there is no need to further adjust the configuration of the viewer after you add it to the page.

Web Content Viewers that are configured to create content can be used multiple times, either on the same page or on different pages. Each instance of the viewer references a separate copy of the content item that is referenced in the portlet configuration.

Copied content is stored in the site area that is identified by the default content association of the current page. In addition, the portlet configuration of the newly added viewer is automatically updated to specify a relative path to the copied content.

Link Web Content Viewers

Many Web Content Viewers can be added to a single portal page or a series of pages. Sometimes it is necessary for different Web Content Viewers to interact with each other. For example, a menu component might be placed in one viewer and a content item in another viewer. If you want the content item to change when a different link is clicked in the menu, you must link the two viewers.

Web Content Viewers can broadcast or receive links:

Broadcasting links
The state or context of a Web Content Viewer is not sent directly from one portlet to another. You can configure viewers to broadcast their current state or context to other viewers on the same page or to viewers on other pages. Any information broadcast by a Web Content Viewer is received only by viewers that are configured to receive this information.

Receiving links
A Web Content Viewer can receive the following information:

- Information about the state or context of the current content item or component that is being rendered by the viewer.
- Information from content items or components that are rendered by other viewers that are broadcasting links.

For examples of the different ways that you can use linking with Web Content Viewers, see *Link examples for Web Content Viewers.*

Web Content Viewers and remote servers

To display web content on a portal that does not include Web Content Manager, you can use the Web Content Viewer and the WSRP support in the portal. The Web Content Viewer can then retrieve and display content from a web content system on a different server.

Related tasks:

"Enabling remote rendering with WSRP and the Web Content Viewer" on page 2069
To display web content on a portal that does not include IBM Web Content Manager, you can use the Web Content Viewer and the WSRP support in the portal. The Web Content Viewer can then retrieve and display content from a web content system on a different server.

Related reference:

"Link examples for Web Content Viewers" on page 2032
Web Content Viewers can broadcast and receive links to communicate with other viewers. Depending on the link settings that you use with the viewers, the behavior of the viewers can be different. These examples demonstrate how different broadcast and receive settings can affect what a viewer renders.

Web content pages and templates:

Web content pages are portal pages that are associated with content that is managed in IBM Web Content Manager. Similar to web content pages, web content templates are page templates that are associated with content in Web Content Manager.

Pages and page templates are tied to web content by content associations. These associations are defined in the page properties and can be specified with the Associations window when you edit page properties. You can associate a page with one or more site areas in one or more web content libraries.

If managed pages are enabled, all managed pages are automatically associated with a corresponding page site area in the Portal Site library. This type of association is called a system content association and enables changes to the page to be managed by Web Content Manager. System content associations are managed by the portal and cannot be deleted or changed manually.

When there are multiple content associations for a page, one of the associations is designated as the default association. Typically, the default association is the system content association for the page. However, you can use the Associations window to specify a different content association as the default association.

Web content pages

With web content pages, you can take advantage of the following benefits when you render web content:

- If you add an unconfigured web content viewer to the page, the viewer automatically renders the default content of the site area that is specified by the default content association.
- Dynamic page selection determines the best web content page to use to render a content item when you click a link to the content item. For example, if you click a link to a content item in a search result, the portal evaluates the following set of content associations:
  - Associations that exist for the site area that directly contains the target content item.
  - Associations for any site areas that are ancestors of the site area that contains the target content item.

  The portal identifies the page that is mapped to the site area that is closest to the target content item. The viewer then renders the content item on that page.
- You can extend friendly URLs to reference content items that are rendered on a web content page. Friendly URLs for web content are composed by combining the friendly URL of the current page and the content path of the rendered content item.

You can create a web content page in two ways:

- Create the page from a web content page template.
- Add a content association to an existing portal page.

Web content page templates

To create a web content page template, create a page under the **Page Templates** label. This label is the root label for all page templates in the portal. You can access the **Page Templates** label by clicking the **Administration menu** icon and then by clicking either of the following locations:

- **Portal User Interface** > **Page Templates**
- **Portal User Interface** > **Manage Pages** > **Content Root** > **Hidden Pages**

When you create a web content page template, you define the layout, style, and contents of any web content page that is created from the template. Web content page templates have all the same flexibility and customization features as a standard portal page or portal page template. You can do common tasks like adding content with portlets, changing the style of the page, or changing the layout of objects on the page. By using viewers with other portlets in a web content page template, you can create pages that support a wide range of user goals. Likewise, you can rely on only viewers and create a website that is primarily composed of information in your web content system.

As with standard page templates, you can create web content pages from a web content page template:

- Manage Pages administration portlet
- Create Page tab in the site toolbar

When users create a web content page from a template, content that is associated with the page template is copied with the page itself. If you create multiple pages from the same page template, each page results in a separate copy of that content.

When you create a page from a template, page titles in any language are not copied. The following elements are copied to the new page:

- Portlet entities, including portlet preferences
- Page layout and style
- Theme and skin settings
- Portlet wires for communication with other portlets
- Page parameters
- Page description (all languages)

In addition, the following changes take place automatically, depending on the individual web content associations that exist on the page.

- If the page template is a managed page, with a system content association that references the Portal site library, the following changes apply:
  - A portal page site area is created in the Portal Site library, with the title of the site area that is derived from the title of the new page. The hierarchy structure of the portal page site area is automatically synchronized with the page hierarchy in the portal.
  - All authoring template mappings and all nested content are copied over into the new portal page site area. However, any nested portal page site areas are not copied.
- If a default content association references a library other than the Portal Site library, the following changes apply. These changes apply regardless of whether the page template is a managed page.
  - A site area is created, with the title of the site area that is derived from the following elements:

- The title of the new page.
- The name of the site area that is being derived from the friendly URL name of the page.

– The site area is created as a child of the site area that is defined as the default content association of the parent page of the new page. This support requires that the parent page is associated with a site area outside the Portal Site library.
– All site area properties and all nested content are copied over to the new site area.
– The default content association on the new page is modified to reference the newly created site area.

Web content viewers on the page template can be configured to reference content that is copied when a page is created from the template. When the page is created, the viewer configuration is automatically adjusted to point to the new content that is created during page instantiation.

Note: Managed pages must be enabled to support page templates that store their associated web content in the Portal Site library. If you disable managed pages, the content that is associated with a template is no longer copied during page instantiation. In addition, the corresponding preferences of any web content viewers that are on the page are not adjusted.

Hierarchical page templates

It is possible to create a hierarchy of templates, for examples, a parent template page with a child page. If the page metadata `ibm.portal.instantiation.page.include.descendants` is set to `true` on the parent page, then creating a page from the parent template not only creates a single page but the complete hierarchy that includes the children.

Related tasks:

"Creating a web content page" on page 2035
A web content page is a page that is associated to one or more site areas in IBM Web Content Manager. You can create a web content page from a web content template page, or you can convert an existing portal page into a web content page.

Web content associations:

Web content associations are used to combine portal pages and associated web content items that are managed by IBM Web Content Manager so they can be managed and rendered consistently. Web content associations map portal pages to the site structure in the IBM Web Content Manager system.

You can define a default content association and multiple other associations, which are used for dynamic page resolution. For each page, a system content association maps the page to its corresponding portal page site area in the Portal Site library. You also have the option of using page template instantiation to trigger the copying of web content into the Portal Site library or another library. This option is set by a page template parameter.

Each web content association consists of a reference to a portal page and a reference to a site area in a web content library. When a page contains a web content association, web content viewers added to the page can automatically render the content that is provided by the associated site area. In addition, a web

content page template that contains an association can create copies of associated content when you create a page by using the template.

When multiple associations are defined for the same web content page, one of those associations is identified as the default content association. When you create a page and Managed Pages are enabled on the system, a system content association is automatically created to the corresponding portal page site area in the Portal Site library. This system content association is designated as the default association, but you can change that setting later as needed.

The default content association has several uses:

- When you add a web content viewer to the page without configuring the viewer to reference content, it renders the content indicated by the default association.
- When you create a page from a web content page template, the default content association of the page template indicates the site area to be copied during page instantiation.
- If you are building a friendly URL to content on the page, the default association indicates the path to the rendered content. This content path fragment is appended to the friendly URL of the current page to generate the complete friendly URL.

All content associations for a page are used for dynamic page resolution. The portal uses dynamic page resolution to determine the best matching page for rendering a specific content item.

You can also configure each web content association to enable or disable page-based access control when you render content from the mapped site area. With this feature, users who are authorized to view the page are also assumed to have view access for content under the associated site area.

Web content associations are managed by the content mapping service of the portal. You can manipulate web content associations with the following methods:

- The Page Associations window available in the site toolbar through **Page** > **General** > **Details** > **Default site area**.
- The page properties available from the Manage Pages portlet
- XML configuration interface, by using the **xmlaccess** command
- Portal Scripting Interface
- REST API for content associations
- Public Java API for the content mapping service

Page context and user context

The content associations of a page can define the initial web content context of the page. This context is used for rendering when users first access a page. The context of the page can change when users interact with the content on the page. Each web content viewer on the page can be configured with an explicit context that overrides the rendering page context. The context of the portlet can also change if it is configured to receive links. When users click a link within the viewer that is configured to broadcast its links, the page context is updated. This new context is maintained until users click another link, or until users start a new session. When users start a new session, the original page context is used.

Content copying triggered by page template instantiation

When the page template parameter called `ibm.portal.instantiation.content.dynamic.copy.target.selection` is set to **true**, the default content association that references a library other than the Portal Site library is copied into the Portal Site library during page instantiation. This allows you to store associated web content outside of the Portal Site library and have this content that is copied into the Portal Site library during page template instantiation. Using a separate library for your page templates is useful if you want to update the library independently of the Portal Site library, either through importing and exporting or syndication. A separate library is also helpful if you use the same content items on several page templates.

This parameter does not affect web content that is associated with page templates through system content associations, such as web content that is stored in the Portal Site library. Such content is always copied to the new page site area associated with the new page.

If the parent page of the new page has both a system content association with the Portal Site library and a default content association that references another library, you can use the `ibm.portal.instantiation.content.preferred.copy.target` parameter to identify the copy target location.

`ibm.portal.instantiation.content.preferred.copy.target=internal` copies the content into the Portal Site library.

`ibm.portal.instantiation.content.preferred.copy.target=template` copies the content to the other library associated with the parent page. This is the default value.

Set page parameters on your web content page templates by using **Page Properties** > **Advanced** in the Manage Pages portlet or by using **Page** > **General** > **Edit Page Properties** > **Advanced** in the site toolbar.

Related concepts:

"Web content pages and templates" on page 2021
Web content pages are portal pages that are associated with content that is managed in IBM Web Content Manager. Similar to web content pages, web content templates are page templates that are associated with content in Web Content Manager.

Related tasks:

"Enabling page-based access control for web content pages" on page 2059
Typically, when you render content items in a web content viewer, access control enforcement on those content items is handled by IBM Web Content Manager. However, you can use page-based access control to delegate access control enforcement to the web content page that is used to display the content.

Related reference:

"Content associations reference" on page 2076
Content associations are used to associate web content pages with your web content site structure. When you select a folder to associate with a web content page, a content association is created and maintained within the portal.

"XML configuration interface and content associations" on page 2076
With the XML configuration interface (**xmlaccess** command), you can perform batch updates of content associations or export associations to import into another portal. Content association information is represented in the XML configuration

schema by content-mapping-info elements.

"Portal Scripting Interface and content associations" on page 2078
With the Portal Scripting Interface, you can create scripts to automate the management of content associations. Using the ContentMapping bean with the Portal Scripting Interface, you can add, modify, and remove content associations.

"REST API and content associations" on page 2080
If you are creating or extending an application and want to manage content associations with that application, you can use portal remote APIs. These APIs retrieve a list of content associations and then create, update, or delete associations.

Related information:

Java API for Content Mapping Service

Creating content with sample web content template items:

To illustrate how page templates, web content viewers, and content associations work together, IBM Web Content Manager provides sample web content. The sample content includes examples of web content template pages and predefined portlets that you can add to pages to render content.

About this task

In addition to examining how these rendering pieces are defined, you can also explore the web content libraries that contain the sample content.

Note: If you want to customize the sample web content that IBM WebSphere Portal Express provides, create copies of the sample web content libraries and customize those copies.

"Adding sample content with the site toolbar" on page 2027
The sample content that is provided with the portal includes four pre-configured content items that you can add to a page from the site toolbar. Dragging one of these items into a page automatically creates a copy of the item and adds the copy to the page. The predefined content items are available in the site toolbar in the **Web Content** category of the **Create** > **Content** tab.

"Creating content with the Articles template page" on page 2029
The sample content that is provided with IBM Web Content Manager includes a web content template page that is called **Articles**. This template page demonstrates how you can build pages with predefined content.

CF03 "Adding the Articles template page to a virtual portal" on page 2030
By default, virtual portals do not contain the Articles template page. To use the Articles template page in a virtual portal, follow the procedure given here.

"CSS styles used by the sample web content template items" on page 2031
The markup that is generated by the sample web content template items is primarily controlled by presentation templates that are stored in the Web Content Templates 3.0 library. These templates rely on the availability of several CSS class definitions in the `wp_oob_sample_styles` theme module.

"Adding the sample web content libraries in the authoring portlet" on page 2032
The templating sample content that is provided with IBM WebSphere Portal Express is delivered in two web content libraries: `Template Page Content 3.0` and `Web Content Templates 3.0`. You can use these libraries and their content as a starting point for working with web content page templates and developing your own templates.

Adding sample content with the site toolbar:

The sample content that is provided with the portal includes four pre-configured content items that you can add to a page from the site toolbar. Dragging one of these items into a page automatically creates a copy of the item and adds the copy to the page. The predefined content items are available in the site toolbar in the **Web Content** category of the **Create** > **Content** tab.

About this task

- The Article presents a simple article that is composed of a title, a short description, and a rich-text body. The default presentation template that is used to render individual articles includes the following elements:
  - Lightweight inline widgets to support tagging and rating
  - Extra site analytics data that is added to the generated markup
- The List of Articles presents a list of all articles that exist in the context of the current page. The viewer also enables users to create articles.
- The Rich Text presents a simple rich text element.
- The Image presents a simple image.

Procedure

To use these sample web content viewers, complete the following steps:

1. Go to the page where you want to add the viewers.
2. Edit the page, and select the **Create** > **Content** tab of the site toolbar.
3. Click **Web Content**. Now the **Create** > **Content** tab shows the Article, List of Articles, Rich Text, and Image content items.
4. Add the content items to your page by clicking the plus sign (+) or dragging the items onto the page. You can also adjust the arrangement of the portlets.
5. Click **Save & Exit**.

Results

When you add one of these content items to a page, new content is created. The content item is copied into the site area that is specified by the default content association of the page. A web content viewer instance is added to the page and automatically adjusted to reference the copied content item.

To help you better understand what these components are doing, here is a more detailed look.

Article

When added to a page, the Article renders a sample article from a web content library. The article also includes Edit and Delete icons to modify the article content or to delete the article from the web content library.

The **drag and drop** configuration that is mapped to the Article content item defines the preferences for the corresponding content viewer portlet. It has the following features:

- The viewer uses a customized portlet title (`Article`) instead of the default title to provide contextual information about the content that is rendered by the viewer. When you are assembling pages and adding web content viewers, taking advantage of details like customized titles is important for orienting users.

- To ensure that the content item to be rendered can be passed in by other viewers, the viewer is configured to receive links with the setting **Other portlet and this portlet**. For example, if you click a link in an instance of the List of Articles viewer on the same page, this viewer renders the linked content.
- The Article viewer also defines preferences that enable the generation of extra metadata for the page, which is based on the currently rendered article instance. This metadata includes description, keywords, and author.
- The default presentation template that is used to render articles (`Web Content Templates 3.0/Article`) includes an authoring tools component. The component enables users to modify the content of individual articles with inline editing. The authoring tools component is hidden automatically when the page is not in edit mode.

List of Articles

When added to a page, the List of Articles renders a list of sample articles from a web content library. It references a menu in web content manager that renders a list of all articles that are associated with the current page. The content also includes a **button** to create an article. The **drag and drop** configuration that is mapped to the List of Articles content item defines the preferences for the corresponding content viewer portlet. It has the following features:

- The List of Articles viewer does not contribute page metadata or a page title. This behavior occurs because the viewer renders a list of items instead of specific content that would determine the semantic content of the page.
- Because the viewer is rendering a list of items, the viewer is not listening to links that are broadcast from other portlets.
- The default presentation template that is used to render lists of articles (`Web Content Templates 3.0/List of Article`) includes an authoring tools component. The component enables users to create an article with inline editing. The authoring tools component is hidden automatically when the page is not in edit mode.
- The viewer is configured to broadcast links with the **Dynamically select a web content page** setting. This configuration results in the List of Articles viewer to determine the content that is rendered by other viewers on the same page, such as the Articles viewer.

Rich Text

The Rich Text content item consists of a name, which is not rendered by default, and a rich text body. The default presentation template that is used to render rich text (`Web Content Templates 3.0/Rich Text`) includes an authoring tools component. The component enables users to modify the content of the rich text item with inline editing. The authoring tools component is hidden automatically when the page is not in edit mode.

Image

The Image viewer content item consists of a name, which is not rendered by default, and a reference to an image. The default presentation template that is used to render images (`Web Content Templates 3.0/Image`) includes an authoring tools component. The component enables users to modify the image reference with inline editing. The authoring tools component is hidden automatically when the page is not in edit mode.

Related tasks:

"Creating a web content page" on page 2035
A web content page is a page that is associated to one or more site areas in IBM Web Content Manager. You can create a web content page from a web content template page, or you can convert an existing portal page into a web content page.

Creating content with the Articles template page:

The sample content that is provided with IBM Web Content Manager includes a web content template page that is called **Articles**. This template page demonstrates how you can build pages with predefined content.

About this task

The Articles template page is a managed page that contains two web content viewers. These viewers are configured to render content items that are stored in the site area for the page in the Portal Site library.

Procedure

To create a page that is based on the Articles template, complete the following steps:

1. Go to the page where you want to add the new page.
2. Edit the page and select **Create** > **Page** in the site toolbar.
3. Use the Create Page tab to create the page. Select the **Articles** template for the new page.

Results

The new page contains the List of Articles web content viewer and the Article viewer:

After you create a page with the Articles template, the new page contains two web content viewers. The List of Articles viewer shows a list of any articles in the site area that are associated with the page. The Article viewer shows a sample article that was created.

The Articles template demonstrates several key features of web content page templates:

- The Articles template page contains a system content association to the site area for the template page in the Portal Site library (`Portal Site/Content/Content Root/Hidden Pages/Page Templates/Articles`. This site area contains three content items that are used as the initial content for pages that are created from the page template. To see this association, edit the page, and then in the site toolbar, click **Page** > **General** > **Details** > **Default site area**.
- There are two web content viewers included with the Articles template:
  - The List of Articles viewer renders links to all articles stored in page site area.
  - The Article viewer renders a specific article that is selected in the List of Articles viewer. The viewer also contributes page metadata to the HTML head section of the rendered portal page.
- The content items that are provided by default require CSS styles that are defined in the `wp_oob_sample_styles` theme module. To make these styles available to the default content items, you must ensure that the page that

contains the items uses a theme profile that includes the `wp_oob_sample_styles` theme module. The deferred profile (`profile_deferred.json`) includes this theme module.

For more information about theme modules and theme profiles, see *The module framework.*

When you create a page that is based on the Articles template, several things happen:

- A page is created based on the page template. The new page has the same layout and style as the template and contains copies of the two web content viewers.
- The new page has a system content association to its corresponding portal page site area.
- The content items that are contained in the site area for the page template are copied into the site area that is associated with the new page.
- The Articles viewer and List of Articles viewer on the new page automatically render the newly created content that is associated with the new page. The rendering occurs because of the viewer configuration:
  - The configuration of the List of Articles viewer is updated during instantiation to point to the newly generated `List of Articles` content item.
  - The Article viewer is configured to directly retrieve its context from the content association of the containing page.

Note: The Articles page template is a managed page that stores its associated web content in the Portal Site library. If you disable managed pages, the content that is associated with this template is no longer copied during page instantiation. In addition, the corresponding preferences of the Articles viewer and List of Articles viewer are not adjusted.

Related concepts:

"The module framework" on page 2526
The module framework allows extensions to contribute to different areas of a page to provide flexibility, enhance the user experience, and maximize performance.

Adding the Articles template page to a virtual portal: CF03

By default, virtual portals do not contain the Articles template page. To use the Articles template page in a virtual portal, follow the procedure given here.

Procedure

1. Syndicate the two libraries Web Content Templates 3.0 and Template Page Content 3.0 from the main portal installation to the virtual portal.
2. To add the template page to the virtual portal, start the following portal configuration engine task:
   - Linux :

     ```
     ./ConfigEngine.sh action-init-content-templating-pages
                       -DVirtualPortalContext=virtual_portal_context_url
     ```

   - IBM i:

     ```
     ConfigEngine.sh   action-init-content-templating-pages
                       -DVirtualPortalContext=virtual_portal_context_url
     ```

   - Windows:

     ```
     ConfigEngine.bat  action-init-content-templating-pages
                       -DVirtualPortalContext=virtual_portal_context_url
     ```

3. To move the referenced content items into the PortalSite library, start the following portal configuration engine task:
   - Linux :

     ```
     ./ConfigEngine.sh action-internalize-content-mappings-vp
                       -DVirtualPortalContext=virtual_portal_context_url
     ```

   - IBM i:

     ```
     ConfigEngine.sh   action-internalize-content-mappings-vp
                       -DVirtualPortalContext=virtual_portal_context_url
     ```

   - Windows:

     ```
     ConfigEngine.bat  action-internalize-content-mappings-vp
                       -DVirtualPortalContext=virtual_portal_context_url
     ```

Results

After you complete these steps, the Articles template page is ready for you to use in the virtual portal.

CSS styles used by the sample web content template items:

The markup that is generated by the sample web content template items is primarily controlled by presentation templates that are stored in the Web Content Templates 3.0 library. These templates rely on the availability of several CSS class definitions in the `wp_oob_sample_styles` theme module.

Any pages that render the sample items must use a theme profile that includes the `wp_oob_sample_styles` theme module or another module that contains the same CSS class definitions. An example of such a theme profile is the full theme profile that is installed by default.

The `wp_oob_sample_styles` theme module includes a single CSS file that is named `oob_samples.css`. This stylesheet is loaded from the path *`common_resources_root`*`/ibm/css/samples`. The *common_resources_root* node is specified in the WebSphere Integrated Solutions Console by the **`resources.commonResourcesRootURI`** property of the WP GlobalThemeConfig resource environment provider. The default value for this property is `dav:fs-type1/common-resources`. This value causes the CSS styles to be loaded from the portal file store with the following path: `/common-resources/ibm/css/samples/oob_samples.css`. For more information about setting the *common_resources_root* node, see *Adapting the list of required runtime configuration changes for your theme.*

Stylesheet note: The `Rich Text` presentation template that is used as the default presentation for Rich Text content items uses CSS styles that render differently depending on the layout container that contains the Rich Text web content viewer. Different classification CSS classes are assigned to these layout containers. As a result, the font sizes of headings and body elements are reduced when a rich text item is moved from the center column into a more narrow side column.

Related tasks:

"Adapting the list of required runtime configuration changes for your theme" on page 2819
You must adapt the list of required runtime configuration changes for your theme.

Adding the sample web content libraries in the authoring portlet:

The templating sample content that is provided with IBM WebSphere Portal Express is delivered in two web content libraries: `Template Page Content 3.0` and `Web Content Templates 3.0`. You can use these libraries and their content as a starting point for working with web content page templates and developing your own templates.

About this task

Note: If you want to customize the sample web content that WebSphere Portal Express provides, create copies of the sample web content libraries and customize those copies.

Web Content Templates 3.0 library
: This library contains the shared components, authoring templates, and presentation templates that are used by the content items that are contained in the `Template Page Content 3.0` library. The library also contains the sample content items that are used with the "Articles" page template.

Template Page Content 3.0 library
: This library contains the content items that are used by web content viewer clones that create content when added to a page. These content items represent the base content that is copied when you add content from the **Web Content** category of the site toolbars **Create** > **Content** tab to a page. Initially, this library contains two content items. The **Article** content item represents a simple article that is created from the "Articles" authoring template in the Web Content Templates 3.0 library. The **List of Articles** content item is an instance of the "List of items" authoring template in the Web Content Templates 3.0 library. The item represents a list of all articles that exist in the context of the currently rendered page.

Virtual portal note: If you want to use the sample content with a specific virtual portal, you must syndicate these web content libraries to the virtual portal. If you fail to syndicate these libraries, an error is displayed when you add the sample content to a page.

Procedure

To work with the sample web content libraries, complete the following steps:

1. Open the authoring portlet for Web Content Manager by clicking **Applications** > **Content** > **Web Content Management**.
2. In the authoring portlet, click **Preferences** > **Configure**.
3. In the **Library Selection** section, add the `Template Page Content 3.0` and `Web Content Templates 3.0` libraries to the list of selected libraries.

Link examples for Web Content Viewers:

Web Content Viewers can broadcast and receive links to communicate with other viewers. Depending on the link settings that you use with the viewers, the behavior of the viewers can be different. These examples demonstrate how different broadcast and receive settings can affect what a viewer renders.

Example 1: Configuring links for a single portlet

In this scenario, a single Web Content Viewer is configured to show a dynamic component, such as a menu. The menu contains links to content in a site area. When you click a link, the viewer continues to render the component, but the component uses the target content to set the web content context. Based on the new context, the component generates appropriate markup. In this example of a menu, the component shows links to content of the site area where the target content is located.

Table 320. Example 1: Configuring links for a single Web Content Viewer

Web Content Viewer content	Broadcast links to	Receive links from
Component	None	This Web Content Viewer

When you display a component with a Web Content Viewer, you can also select an alternative presentation template in the content view of the content section. When a link is clicked in the component, the content that is then rendered uses the alternative presentation template.

Example 2: Configuring links for a menu and content

In this scenario, one Web Content Viewer contains a Web Content Manager menu and another Web Content Viewer contains Web Content Manager content. Links must be created between the two viewers to enable the rendered content to change when different links in the menu are selected.

Table 321. Example 2: Configuring links for a menu and content

Web Content Viewer content	Broadcast links to	Receive links from
Menu	This page	None
Content	None	Other Web Content Viewers and this viewer

Example 3: Configuring links for a navigator and content

In this scenario, one Web Content Viewer contains an item views navigator and another viewer contains some Web Content Manager content. Links must be created between the two viewers to enable the rendered content to change when different links in the navigator are selected. The navigator also changes to reflect the current state of the content that is rendered.

Table 322. Example 3: Configuring links for a navigator and content

Web Content Viewer content	Broadcast links to	Receive links from
Navigator	This page	Other Web Content Viewers and this viewer
Content	This page	Other Web Content Viewers and this viewer

Example 4: Configuring dynamic links for a navigator and web content pages

In this scenario, one Web Content Viewer contains a site views navigator, and several web content pages that are associated with different site areas. Each web

content page contains a viewer and is configured to show the default content item of the site area that is associated with the page.

Instead of manually creating links between the different viewers, you can use dynamic link broadcasting with the viewer. Dynamic link broadcasting automatically determines which web content page is used as the target for the links to the site area in the navigator.

Table 323. Example 4: Configuring dynamic links for a navigator and web content pages

Web Content Viewer content	Broadcast links to	Receive links from
Navigator	Dynamically select a web content page	None
Content on web content page	This page	Other Web Content Viewers and this viewer

Related concepts:

"Web Content Viewers" on page 2017
Web Content Viewers are portlets that render content from a web content library as part of a portal page. If your presentation is simple, a single viewer can be sufficient. To provide a richer experience for your users, use multiple viewers to aggregate content from different libraries.

Displaying content with Web Content Viewers

Display content from your web content system by adding a Web Content Viewer to the server where you want the content to show.

About this task

If your presentation is simple, a single Web Content Viewer can be sufficient. You can also use multiple Web Content Viewers to provide a richer experience for your users.

Depending on how you decide to deploy the servers in your environment, there are different ways to render content:

- You can install a Web Content Viewer locally on the same portal server where Web Content Manager is installed.
- You can install the viewer remotely on a different portal server.

  "Adding a Web Content Viewer Portlet"
  Add a Web Content Viewer Portlet to a page with the site toolbar.

  "Creating a web content page" on page 2035
  A web content page is a page that is associated to one or more site areas in IBM Web Content Manager. You can create a web content page from a web content template page, or you can convert an existing portal page into a web content page.

Adding a Web Content Viewer Portlet:

Add a Web Content Viewer Portlet to a page with the site toolbar.

Procedure

1. Go to the page that you want to add the new page.
2. Edit the page, and select **Create** in the site toolbar.
3. Click **Applications**. Now **Create** > **Applications** shows the available applications.

4. Search for the portlet Web Content Viewer.
5. Click **Add application to the page** that is indicated by the plus sign icon, or drag the items to the page to add the Web Content Viewer to your page
6. You can modify settings for the Web Content Viewer, such as a content association or portlet title.

Creating a web content page:

A web content page is a page that is associated to one or more site areas in IBM Web Content Manager. You can create a web content page from a web content template page, or you can convert an existing portal page into a web content page.

About this task

A web content page always restricts the sharing scope for public rendering parameters to the page itself. The corresponding page parameters are automatically added to the page when a page is associated to a site area.

Caching note: When you are using web content pages, you cannot use advanced web content caching but instead can use only the portlet fragment cache.

Procedure

1. There are two ways to create a web content page.
   - Follow these steps to create a web content page from a web content template page:
     a. Go to the page where you want to add the new page, and edit the page.
     b. Open the site toolbar, select **Create** > **Page** use the Create Page tab to create the page. Select a web content page template for the new page.

        Note: If managed pages are enabled, all page templates result in web content pages.
     c. Click **Create Page** to create a web content page from the selected template.

        If the page template has a default content association, the new page is automatically associated with a new copy of the referenced site area. Any other content associations that are not designated as the default association are copied without changes.
   - Follow these steps to convert a portal page to a web content page:
     a. Go to the page where you want to add the new page, and edit the page.
     b. Open the site toolbar and select **Page** > **General** > **Details** > **Default site area**
     c. In the Page Associations window, click **Add web content**.
     d. Select one or more site areas that you want to associate with the page, and click **OK**.
     e. Specify the default association by selecting the association in the **Default** column.
2. After you create the web content page, you can add web content viewers to the page from the **Create Applications** tab of the site toolbar.

   You can add a standard Web Content Viewer portlet, or you can add any predefined web content viewers that you create.

- If you add a standard web content viewer, the viewer renders any content from the site area that is indicated by the default content association of the page.
- If you add a predefined viewer that is configured to create content with the **Create content (based on selection)** setting, the following things happen:
  - The base content that is referenced by the viewer is copied into the site area that is referenced by the default content association of the page.
  - The new instance of the viewer renders the copied content in the site area that is referenced by the default content association of the page.

  An example of this type of predefined viewer is the Rich Text viewer that is available from the **Web Content** category of the **Content** tab.

Related concepts:

"Web content pages and templates" on page 2021
Web content pages are portal pages that are associated with content that is managed in IBM Web Content Manager. Similar to web content pages, web content templates are page templates that are associated with content in Web Content Manager.

Related tasks:

"Enabling page-based access control for web content pages" on page 2059
Typically, when you render content items in a web content viewer, access control enforcement on those content items is handled by IBM Web Content Manager. However, you can use page-based access control to delegate access control enforcement to the web content page that is used to display the content.

Customizing web content delivery

Although web content viewers and page templates provide the basis for delivering web content, you can customize the environment to provide users with a better experience.

About this task

For example, you can create custom objects like page templates and web content viewers. You can also provide users with conveniences like customized error messages and friendly URLs that reference web content.

"Creating web content page templates" on page 2037
Create web content page templates to quickly deploy new pages that contain web content. With a template, you can define the layout and presentation of the page, including adding web content viewers that are configured to render web content.

"Creating web content when you add a web content viewer to a page" on page 2038
To make it easier for users to find web content, the site toolbar provides palettes. You use these palettes to organize components that you can add, such as portlets, iWidgets, and web content.

"Customizing error messages for Web Content Viewers" on page 2040
If an error occurs during rendering, the Web Content Viewer shows an error screen. You can customize the default error screen, and you can create your own custom JSP file that is used to display error messages.

"Friendly URLs and Web Content Viewers" on page 2040
Friendly URLs provide a way for you to define a custom address for a portal page that is easy to remember and share. The Web Content Viewer expands on friendly URL support so you can specify extra path information in the friendly URL.

"Setting up a web content fallback page" on page 2057
Set up a web content fallback page to be used when a web content viewer cannot determine which page to use to display a content item. The fallback page can also be used when users do not have sufficient privileges to view the page originally associated with the content item.

"Enabling page-based access control for web content pages" on page 2059
Typically, when you render content items in a web content viewer, access control enforcement on those content items is handled by IBM Web Content Manager. However, you can use page-based access control to delegate access control enforcement to the web content page that is used to display the content.

"Previewing content on web content pages" on page 2060
When working with content in the authoring portlet, you can preview content items in a portal environment as part of a web content page. To preview content items, there must be a web content viewer on the associated web content page, and the viewer must be configured to receive links.

"Adding HTML meta tags for Search Engine Optimization" on page 2060
Search engine optimization (SEO) focuses on improving the visibility of a page or website in search engine results. A basic technique of SEO is adding HTML title and meta tags to your page source. These meta tags are used to define description information and other metadata that search web engines and crawlers can use when they create search indexes and collections. When you include content in a page with a web content viewer, you can improve the search engine optimization of the page by adding title and meta tags with values derived from the web content itself.

CF06 "Improving page loading performance with asynchronous web content rendering" on page 2065
You can increase page loading performance by separating portal page content delivery from web content rendering. To do so, you use the asynchronous web content rendering feature. If you enable asynchronous web content rendering, the portal serves the page immediately, but only with placeholder content. The Web Content Viewer then dynamically inserts the configured content into the portal page after the content has been completely rendered.

Related concepts:

"Preparing the site toolbar" on page 1900
The site toolbar enables content authors to work with content from the website instead of using the authoring portlet. Customize the site toolbars to improve the authoring experience for content authors.

Creating web content page templates:

Create web content page templates to quickly deploy new pages that contain web content. With a template, you can define the layout and presentation of the page, including adding web content viewers that are configured to render web content.

About this task

When a page is created from the template, the site structure that is required in the web content library is also created automatically. For example, if managed pages are enabled, a site area that corresponds to the page hierarchy is created in the Portal Site library. If managed pages are disabled, a site area is created as a child of the site area that is associated with the parent page of the new page. You can access the resulting site structure in the Web Content Manager authoring portlet.

Procedure

1. Using the administration interface, browse to the page templates location. You can access this location from either of the following paths:
   - Click **Portal User Interface** > **Page Templates**.

     Note: This path uses the site toolbar that is included with the Portal 8.0 theme.
   - Click **Portal User Interface** > **Manage Pages** > **Content Root** > **Hidden Pages** > **Page Templates**.
2. Create a page. When you create the page, use the **Basic** template and then define a content association for the page. The content that is contained in the mapped site area is copied to any page that is created by using the page template.
3. Edit the layout of the page and add any web content viewers or other portlets that you want to include in the template.

   If you add a viewer that is intended to render content that is copied when the page is created, then configure the viewer:
   a. Select **Edit Shared Settings**.
   b. Select **Select content and use the content association of current page** from the **Content behavior** section. This setting causes the viewer to reference content from the site area that is defined in the content association on the page that contains the viewer.

      Note: If you did not add a content association to the page template, this option is not available.

Results

After you create the template page, the new template is displayed in the list of available templates when you create a page.

Creating web content when you add a web content viewer to a page:

To make it easier for users to find web content, the site toolbar provides palettes. You use these palettes to organize components that you can add, such as portlets, iWidgets, and web content.

About this task

By default, the Content palette includes four sample content items. You can add your own content to the Content palette. For more details about how to do so, read *Customizing the Content palette*.

To add web content, you can use the preferred drag-and-drop configurations. You can also configure individual copies of the Web Content Viewer portlet to copy web content when you add these portlet copies to a page. Copies of the web content viewer appear in the Applications palette of the site toolbar.

To include a web content item in the Applications palette of the site toolbar, follow the procedure that is given here.

Procedure

1. Make a copy of the portlet called Web Content Viewer. To do so, use the portal administration interface. Give the copy a name that indicates the content item that the viewer represents.
2. Add the new web content viewer to a page. You need this page only temporarily to provide a way of configuring the web content viewer.
3. Open the **Configure** mode for the portlet.
4. From the **Content Type** section, select the content item that you want to add.
5. From the **Content Behavior** section, select **Select content and path**.
6. From the section that is named **When this Portlet is added to the page**, select **Create content (based on selection)**.

   Note: You cannot add portlets that use this setting to a page that does not have a default content association.
7. In the **Content** section, specify the content item or site area that this web content viewer represents.

Results

After you complete these steps, the copy of the web content viewer that represents your web content is displayed in the Applications palette of the site toolbar. Whenever you add this portlet copy to a page, the associated content item or site area is copied into the site area that is associated to the page by the default content association.

Notes for the XML configuration interface: To create the copies of the web content viewer, you can also use the portal XML configuration interface. To specify the behavior of the setting Create content (based on selection) by using with the XML configuration interface, specify the following portlet preference in the XML import file:

Preference:

```
com.ibm.portal.wcm.copy.contents
```

Value:

```
true
```

You can also specify the target site area that you want to be used when the portal copies the web content. To define a target site area, use the following portlet preference in the XML import file:

Preference:

```
WCM_COPY_CONTENT_RELATIVE_PATH
```

Value:

```
site_area_name
```

When you add the web content viewer that specifies this `WCM_COPY_CONTENT_RELATIVE_PATH` preference to a page, the portal first creates the specified site area under the default content association of the page. Afterward, the portal copies the web content that is associated with the web content viewer to the specified site area rather than to the default content association.

Related concepts:

"Customizing the Content palette" on page 1904
The Content palette, which is accessed from the site toolbar, contains the content items a content author can add to a page.

Customizing error messages for Web Content Viewers:

If an error occurs during rendering, the Web Content Viewer shows an error screen. You can customize the default error screen, and you can create your own custom JSP file that is used to display error messages.

About this task

The default error screen provides a standard error message, which is shown on every type of error, and a more detailed error message. The detailed error message provides information about the cause of the error and is displayed when you click the **View details** link.

Procedure

1. Create a customized error JSP file.
   a. Copy the original `error.jsp` file from *wp_profile_root*`/installedApps/`*node_name*`/PA_WCMLRingPortJSR286.ear/ilwwcm⌂localrende.war/jsp/html` directory to create your custom error JSP file.

      Almost everything in the original JSP file can be changed according to your requirements. If you want to show the cause of the error, you must retain this part of the original file in your error JSP file:

      ```
      <%-- use errorbean from request --%>
      <jsp:useBean id="errorbean" scope="request"
          type="com.ibm.portal.portletui.messages.StatusMessageBean" />
      <% String msg = errorbean.getMessage(); %>
      ```

      The variable *msg* contains the message text of the error. In the original `error.jsp` file, this message is only shown in a separate window if a user selects the **View details** link.
2. Configure the Web Content Viewer to use the customized error JSP file.
   a. Log in to the portal as an administrator.
   b. Click the **Administration menu** icon. Then, click **Portlet Management** > **Portlets**.
   c. Locate the `Web Content Viewer` portlet.
   d. Click **Configure Portlet**.
   e. Edit the value of the **`ERROR_JSP`** parameter, and set the path to your customized error JSP file as the parameter value.

      Storing JSP files: JSP files are stored within a web application that runs on the portal. To reference a JSP file in another web application, use the following path: `contextPath;jspPath`. For example: `/wps/customapplication;/jsp/jspFilename.jsp`.

      A dynamic context path value can be defined by adding a token to the context path that corresponds to a key and value pair to the Web Content Manager configuration service environment provider. When this key is used as the token in the jsp value field, it is replaced dynamically at render time. For example: `[my.custom.key];myfile` where *my.custom.key* is a constant within the Web Content Manager configuration service.

Friendly URLs and Web Content Viewers:

Friendly URLs provide a way for you to define a custom address for a portal page that is easy to remember and share. The Web Content Viewer expands on friendly URL support so you can specify extra path information in the friendly URL.

By defining friendly URLs for your portal pages, users can browse the portal by using more concise URLs that better reflect the page structure. The additional path information in a friendly URL for web content points to a content item to be displayed in the Web Content Viewer.

"About friendly URLs for web content"
With friendly URLs for web content, you can construct URLs to content items that are clear and concise. Although you can construct friendly URLs that reference web content items, IBM Web Content Manager itself does not generate friendly URLs by default. However, to cause the web content viewer to generate friendly URLs, you can create a plug-in that implements a content URL generation filter.

 "How to validate friendly URLs for web content" on page 2045
Learn about the validation and customization of friendly URLs for web content, including use cases and customization options.

"Friendly URL for web content example" on page 2052
This example demonstrates how friendly URLs for web content work with multiple web content viewers on a single portal page. The example describes the portal page structure referenced by the friendly URLs and explains the underlying structure of the content in an IBM Web Content Manager site framework.

Related tasks:

"Creating a content URL generation filter class" on page 3237
A content URL generation filter is used to customize the URLs that are generated by a web content viewer. By creating a plug-in that implements a content URL generation filter, you can tailor the URLs to content items.

About friendly URLs for web content:

With friendly URLs for web content, you can construct URLs to content items that are clear and concise. Although you can construct friendly URLs that reference web content items, IBM Web Content Manager itself does not generate friendly URLs by default. However, to cause the web content viewer to generate friendly URLs, you can create a plug-in that implements a content URL generation filter.

These URLs are easier for users to remember and share and are a convenient way for users to create bookmarks to content items. External applications can also use friendly URLs to provide links directly to content items in the portal. To create effective friendly URLs for web content, you must understand how friendly URLs for portal pages are constructed and how friendly URLs for web content extend those URLs.

How friendly URLs for pages are constructed

For a page to be referenced as part of a friendly URL, you must assign a friendly URL name for the page. You can assign a friendly URL when you create the page, or you can edit the page properties after the page is created.

Friendly URLs take the following general form:

`http://host_name:port_number/context_root/portal/page_id/[!ut/p/encoded_suffix]`

The *page_id* portion of the friendly URL is made up of the friendly URL names of each page in the page structure. The page structure begins at the content root and ends with the currently selected page.

For example, you might have a portal page called Products with a friendly URL name of products. Under the Products page is another page called Appliances with a friendly URL name of appliances. When referenced as a complete friendly URL, you would enter the following URL to access the Appliances page:

```
http://www.example.com:10039/wps/portal/products/appliances
```

For friendly URLs to work for a specific page, you must define a friendly URL name for each page or label in the page structure. If you want to suppress a friendly URL name from showing in the friendly URL, you can specify a friendly URL name of com.ibm.portal.friendly.wildcard for the page. For example, if the Products page has a friendly URL name of com.ibm.portal.friendly.wildcard, the friendly URL in the previous example for the Appliances page is abbreviated:

```
http://www.example.com:10039/wps/portal/appliances
```

Note: When the portal displays a page using a friendly URL, the URL can include an encoded suffix at the end of the URL with the form !ut/p/*base_codec*/*rich_state*. This suffix contains information about the portals state that the portal might use when displaying the page. However, when bookmarking or sharing friendly URLs, it is not necessary to include the suffix.

How friendly URLs for web content are constructed

Friendly URLs for web content are constructed just as friendly URLs for pages but include additional information that identifies the path to a content item. When the portal decodes a friendly URL, it decodes the URL from beginning to end. Each path segment of the URL is matched with the friendly URL names of portal pages until no more matches can be located. The remainder of the URL is then considered to be path information to a content item.

This path information is mapped to a shared public render parameter that is scoped to the portal page identified by the URL. The fully qualified name of this **path-info** parameter is http://www.ibm.com/xmlns/prod/websphere/portal/publicparams:path-info. The **path-info** parameter can contain multiple values, with the individual values representing segments of a content path. The segments are concatenated using a forward slash (/) as a path separator.

Friendly URLs for web content take the following general form:

```
http://host_name:port_number/context_root/portal/page_id/path_to_content/[!ut/p/encoded_suffix]
```

When you add a web content viewer to a portal page, the web content viewer reads the **path-info** parameter. The viewer assembles the path to the content to be rendered by appending the path information to the path of the default content mapping defined for the current page. For example, you might have the following friendly URL for web content:

```
http://www.example.com:10039/wps/portal/products/appliances/welcome
```

Several conditions contribute to this URL:

- The portal page Products has a friendly URL name of products, and underneath the Products page is another page called Appliances with a friendly URL name of appliances.
- A web content library contains a site area called Appliances, which contains a content item called welcome. For this example, the web content library is called Web Content.
- The portal page Appliances has a default content mapping to the Web Content/Appliances site area.

When a web content viewer is added to the `Appliances` page, the web content viewer interprets the **path-info** information from the friendly URL. The viewer identifies `welcome` as path information that represents content in a web content library. By examining the default content mapping on the page, the web content viewer locates the `Web Content/Appliances` site area and then displays the `welcome` content item.

The *page_id* portion of the friendly URL is always evaluated first. Because of this priority, ensure that your naming schemes do not overlap when setting up your portal page hierarchy and your web content hierarchy. In particular, the *path_to_content* information cannot begin with segments that could be part of the `page_id` portion of the friendly URL. If the first segment of the *path_to_content* information matches the friendly URL name of a portal page at that point in the page hierarchy, the friendly URL could reference the wrong page.

Considerations for the path-info parameter:

- For a web content viewer to process the **path-info** parameter, the web content viewer must be configured to receive links. If it is configured to receive links, the web content viewer gives precedence to the **path-info** parameter over the **context** public render parameter. When you click links displayed by the web content viewer, the link automatically incorporates the path information for the linked item.
- Clicking **Clear page context** when editing the settings of a web content viewer also clears the **path-info** parameter.
- If a friendly URL includes an encoded suffix, it takes this form: `!ut/p/`*base_codec*`/`*rich_state*. Because this information is encoded, it is not intended to be read by people. However, the portal itself might act on the information, which can sometimes cause the wrong page to be displayed.

  If the **path-info** public shared render parameter is encoded in the *rich_state* portion of the suffix, the **path-info** contents overwrites the *path_to_content* portion of the friendly URL. It is also possible that there could be a mismatch between the **path-info** contents and the path information encoded in the *rich_state* section. If such a mismatch occurs, the portal replaces the *path_to_content* portion of the friendly URL with the *rich_state* information and directs the user to that page.

  The following tables demonstrate how the presence of *rich_state* information affects the page that is shown:

Table 324. Example of rich_state information affecting displayed page

Description	URL
The user navigates to URL in the portal.	`http://www.example.com:10039/wps/portal/home/content_item_1/!ut/p/b1/dY07Do...`
The user modifies the URL in the browsers address bar to go to `content_item_2`.	`http://www.example.com:10039/wps/portal/home/content_item_2/!ut/p/b1/dY07Do...`
Resulting URL.	`http://www.example.com:10039/wps/portal/home/content_item_1/!ut/p/b1/dY07Do...` Because the *rich_state* portion of the URL still contains path information pointing to `content_item_1`, the portal overwrites the *path_to_content* portion of the URL. The user remains on the same page instead of being directed to the page where `content_item_2` is displayed.

Table 325. Example of friendly URL for web content without rich_state information

Description	URL
The user navigates to URL in the portal.	`http://www.example.com:10039/wps/portal/home/content_item_1/!ut/p/b1/dY07Do...`
The user modifies the URL in the browsers address bar to go to `content_item_2`.	`http://www.example.com:10039/wps/portal/home/content_item_2`
Resulting URL.	`http://www.example.com:10039/wps/portal/home/content_item_2` Because the user removed the *`rich_state`* portion of the URL when modifying the URL, the *`path_to_content`* portion of the URL is evaluated. The user is directed to the page where `content_item_2` is displayed.

Content URL generation filters and friendly URLs

A content URL generation filter is used to customize the URLs that are generated by a web content viewer. The web content viewer generates a content URL whenever there is a URL to web content within content that the viewer is displaying. By creating a plug-in that implements a content URL generation filter, you can tailor the URLs to content items. For details, see *Creating a content URL generation filter class.*

Troubleshooting friendly URLs for web content

If you are seeing unexpected behavior when using friendly URLs for web content, review these issues to help identify why the friendly URL is not working. Friendly URLs for web content take the following general form:

`http://`*`host_name`*`:`*`port_number`*`/`*`context_root`*`/portal/`*`page_id`*`/`*`path_to_content`*`/[!ut/p/`*`encoded_suffix`*`]`

- Support for friendly URLs for web content is only enabled when the configuration properties `friendly.enabled` and `friendly.pathinfo.enabled` both have a value of `true` in the portal Configuration Service.
- The web content viewer displays a warning message in the following situations:
  - The friendly URL for web content references a content item that cannot be located.
  - The user does not have sufficient access rights to view the referenced content item.
- The portal page specified in the friendly URL for web content must have a default content mapping to an existing web content site area. If there is no default content mapping on the page, any web content viewers on the page display a warning message about the missing page context.
- If the target page does not contain a web content viewer that is configured to receive links, the content item specified in the friendly URL for web content is not displayed.
- If a web content viewer is not configured to broadcast links, links rendered by the viewer do not affect the friendly URL for web content.
- The default portal page selection does not show the path of the default content item in the friendly URL. The *`path_to_content`* portion of the URL includes the content path information only after users browse web content using links displayed by the viewer.

- Friendly URLs for web content are URL-encoded. When using friendly URLs for web content, special characters that show in any segment of the URL must be URL-encoded. For example, a space character in the URL would be replaced by its URL-encoded equivalent: `%20`. Some web browsers perform automatic decoding of the URL. In this case, you might see unencoded characters in the URL, but the portal always works with an encoded version of the URL.
- The segments of a friendly URL for web content are not localized for multiple languages. The *`path_to_content`* portion of a friendly URL for web content is composed of the unlocalized names of web content folders, site areas, and content items. For example, if you name these items with English terms, the friendly URL for web content is constructed of these English terms, even if the portal language is not English.

How to validate friendly URLs for web content:

Learn about the validation and customization of friendly URLs for web content, including use cases and customization options.

Friendly URLs for web content provide human-readable URLs that are easy for users to remember. A friendly URL points to a specific content item that is displayed on a specific portal page. By default, WebSphere Portal Express does not check whether a friendly URL addresses an existing content item or not. However, the following scenarios would benefit from validation and customized handling of invalid friendly URLs:

- A user mistypes a friendly URL.
- A user clicks a saved bookmark that points to a portal page or a content item that no longer exists.
- A search engine browses portal pages to update a search index.

If you use friendly URLs for web content, you can configure WebSphere Portal Express to validate the friendly URL associated with a request. If you enable friendly URL validation, portal assembles a content path by appending the path information to the path of the default content mapping that is defined for the resolved page. Afterward, portal checks whether that content path identifies a content item that is available in the current user context. The friendly URL for web content is valid if the portal finds a content item. Otherwise, portal sends an error response. Using the WebSphere Portal Express Configuration Service and individual page parameters, you can customize error responses in the following ways:

- Return an HTTP error code along with a localized message.
- Return an HTTP error code and display the resolved page.
- Return an HTTP error code and display the resolved page and render a specific content item.
- Return an HTTP error code and display a specific page.
- Return an HTTP error code and display a specific page that renders a specific content item.

CF03 "How to enable the validation of friendly URLs for web content" on page 2046
Learn about the properties and values that are required in WebSphere Portal Express Configuration Service Resource Environment Provider to validate friendly URLs for web content.

CF03 "Configuring the validation of friendly URLs for web content" on page 2046

After you enable the validation of friendly URLs for web content, you can choose from various configuration options. These options enable you to specify how the portal responds to friendly URLs that contain path information that does not identify an available content item. Learn about the parameter combinations you can specify and how the portal response varies based on these combinations.

CF03 "How to prevent friendly URL redirects for invalid friendly URLs for web content" on page 2051
If the validation of friendly URLs for web content is enabled and the path information of an incoming friendly URL is not valid, portal responds with the HTTP status code as defined by the portal Configuration Service property and page parameter **friendly.pathinfo.validation.errorCode**. However, depending on the configuration, portal does not always send the configured HTTP status code. Portal can identify conditions that require a different HTTP status code.

How to enable the validation of friendly URLs for web content:

Learn about the properties and values that are required in WebSphere Portal Express Configuration Service Resource Environment Provider to validate friendly URLs for web content.

Before you enable the validation of friendly URLs, verify that your portal server supports friendly URLs that contain path information to a content item as part of the URI. Verify the following properties and values in the WebSphere Portal Express Configuration Service:

- **friendly.enabled** is set to `true`.
- **friendly.pathinfo.enabled** is set to `true`.
- **friendly.pathinfo.invalid** is set to `false`.

If the previous properties are not set to the specified values, your portal server does not use friendly URLs for web content and validation of friendly URLs is not possible.

To enable the validation of friendly URLs for web content, set the **friendly.pathinfo.validation.enabled** property to `true` in the WebSphere Portal Express Configuration Service Resource Environment Provider. For more information about the **friendly.pathinfo.validation.enabled** property and the properties that are required to support friendly URLs, see *Configuration Service* in the related links.

Related reference:

"Configuration Service" on page 297
The portal Configuration Service is responsible for collecting the most essential configuration data of the IBM WebSphere Portal Express engine.

Configuring the validation of friendly URLs for web content:

After you enable the validation of friendly URLs for web content, you can choose from various configuration options. These options enable you to specify how the portal responds to friendly URLs that contain path information that does not identify an available content item. Learn about the parameter combinations you can specify and how the portal response varies based on these combinations.

You can specify the following parameters for the overall portal as properties and key value pairs in the portal Configuration Service or for individual pages as page

parameters. Page parameters that you set for a specific page override any global settings that are specified in the portal Configuration Service. Page parameters are inherited by all child pages. For more information, see *Configuration Service* in the related links.

friendly.pathinfo.validation.errorCode = (404)

This key specifies the HTTP status code that the portal returns if the path information of a friendly URL cannot be resolved to a content item for the requested page. You can specify one of the following values:

404 The default value. This HTTP status code tells a caller, such as a search crawler or web browser, that no content is found for the friendly URL. The missing content might be temporarily or permanently missing.

410 This HTTP status code informs a caller, such as a search crawler or a web browser, that the resource for the friendly URL is no longer available. This missing resource is permanently gone.

Portal can identify conditions that require a different HTTP status code than the one you configure by using **friendly.pathinfo.validation.errorCode**. For example, friendly URL redirects require the HTTP status code 302. To support the most common use cases, see the topic *Preventing friendly URL redirects for invalid friendly URLs for web content*.

friendly.pathinfo.validation.errorTextProvider

This key specifies the text provider of the localized HTTP status message to send as well as the configured HTTP status code. If you configure a text provider and a request URL has invalid path information, portal responds with a blank page that displays only the HTTP status code and the corresponding localized message that is specified by the text provider. The value of this parameter must be the ID of an implementation of the `com.ibm.workplace.wcm.api.plugin.textprovider.TextProvider` interface. To use the default messages of WebSphere Portal Express, specify the text provider with the ID `PathInfoValidationTextProvider`. If you implement a custom text provider, make sure that it supports message keys that are composed of the prefix `HTTP_STATUS_MESSAGE_` and the configured HTTP status code, for example: `HTTP_STATUS_MESSAGE_404`.

Important: Portal ignores this setting if you also specify the **friendly.pathinfo.validation.errorURI** property or page parameter.

friendly.pathinfo.validation.errorResourceBundle

This key specifies a Java resource bundle as an alternative to implementing a custom text provider. If you configure a Java resource bundle and a request URL has invalid path information, portal responds with a blank page displays only the HTTP status code and the corresponding localized message from the Java resource bundle. The value of this setting must be the fully qualified name of the Java resource bundle. If you provide a custom Java resource bundle, make sure that it contains message keys that are composed of the prefix `HTTP_STATUS_MESSAGE_` and the configured HTTP status code, for example: `HTTP_STATUS_MESSAGE_404`.

Important: Portal ignores this setting if you also specify the **friendly.pathinfo.validation.errorURI** property or page parameter. Portal also ignores this setting if you set the value of the

friendly.pathinfo.validation.errorTextProvider property or page parameter to a custom text provider ID.

friendly.pathinfo.validation.errorURI
This key specifies the piece of content URI that portal resolves if the request URL has invalid path information. The value of this parameter must be a piece of content URI that portal can resolve, for example:

nm:oid:*unique_page_name*
This navigation model URI redirects the request to a specific portal page based on the unique name of the target page.

custom:resolutionserviceuri
This custom implementation of the `com.ibm.portal.resolver.ResolutionService` interface resolves invalid path information to a dynamically determined navigational state. When portal resolves the piece of content URI, the content path that failed the portlet validation is passed to the resolution service as the **wcmContentPath** parameter.

friendly.pathinfo.validation.errorContentPath
This key specifies the full content path that portal sets as public Web Content Manager context of the resolved page if the request URL has invalid path information. Web Content Viewer portlets on the resolved page that are configured to listen to other portlets can then render the content with the specified path. The value of this setting must be the path of a content item that is available to users, for example: `/Web Content/home/human_resources/health/topic_not_found`.

Table 326. Portal responses to invalid friendly URLs for web content when parameters are set to specific values

Portal response to invalid friendly URLs for web content			**Properties and values**			
HTTP status code	**Page**	**Web content**	**friendly.pathinfo.validation.errorTextProvider**	**friendly.pathinfo.validation.errorURI**	**friendly.pathinfo.validation.errorContentPath**	**friendly.pathinfo.validation.errorTextProvider** or **friendly.pathinfo.validation.errorURI** or **friendly.pathinfo.validation.errorContentPath**
404	The page that is specified in the requested URL.	Web content is rendered according to the private or public Web Content Manager context, but not based on the path information. The path information was invalid and removed from the state.	N/A	N/A	N/A	N/A

Table 326. Portal responses to invalid friendly URLs for web content when parameters are set to specific values (continued)

Portal response to invalid friendly URLs for web content			**Properties and values**			
HTTP status code	**Page**	**Web content**	**friendly.pathinfo.validation**	**friendly.pathinfo.validation.errorTextProvider or friendly.pathinfo.validation.errorTextProvider.bundleURI**		
410	The page that is specified in the requested URL.	Web content is rendered according to the private or public Web Content Manager context, but not based on the path information. The path information was invalid and removed from the state.	410	N/A	N/A	N/A
404	The page displays the following text: `Error 404: The requested content does not exist. Please verify that the URL is correct.`	N/A	N/A	**PathInfoValidationTextProvider** HTTP_STATUS_MESSAGE_404=The requested content does not exist. Please verify that the URL is correct.	N/A	N/A

Table 326. Portal responses to invalid friendly URLs for web content when parameters are set to specific values (continued)

Portal response to invalid friendly URLs for web content			Properties and values			
HTTP status code	**Page**	**Web content**	**friendly.pathinfo.validation.errorTextProvider**	**friendly.pathinfo.validation.errorTextProvider or friendly.pathinfo.validation.errorTextProvider**	**friendly.pathinfo.validation.errorTextProvider**	**friendly.pathinfo.validation.errorTextProvider**
404	The page with the unique name `missing.content` displays.	Web content is rendered according to the private or public Web Content Manager context, but not based on the path information. The path information was invalid and removed from the state.	N/A	N/A	`nm:oid:missing.content`	N/A
404	The page that is specified in the requested URL.	Web content is rendered according to the private Web Content Manager context of the Web Content Viewer portlet or `/Web Content/ Home/ Missing Content`.	N/A	N/A	N/A	`/Web Content/ Home/ Missing Content`

Table 326. Portal responses to invalid friendly URLs for web content when parameters are set to specific values (continued)

Portal response to invalid friendly URLs for web content			Properties and values			
HTTP status code	**Page**	**Web content**	**friendly.pathinfo.validation.errorCode**	**friendly.pathinfo.validation.errorCode or friendly.pathinfo.validation.errorResourceBundle**	**friendly.pathinfo.validation.errorTextProvider**	**friendly.pathinfo.validation.errorTextProvider or friendly.pathinfo.validation.errorResourceBundle**
410	The page with the unique name `missing.content` displays.	Web content is rendered according to the private Web Content Manager context of the Web Content Viewer portlet or `/Web Content/ Home/ Missing Content`.	410	N/A	`nm:oid:missing.content`	`/Web Content/ Home/ Missing Content`

Related tasks:

"Creating a Text Provider class" on page 3213
A text provider is used to provide localized text that can be used within web content item forms. For example, a text provider can be used to localize the field labels or help text within an authoring template so that each user sees the labels or help text in their own language.

Related reference:

"Configuration Service" on page 297
The portal Configuration Service is responsible for collecting the most essential configuration data of the IBM WebSphere Portal Express engine.

Related information:

Advanced options

Editing pages, labels, and URLs

How to prevent friendly URL redirects for invalid friendly URLs for web content:

If the validation of friendly URLs for web content is enabled and the path information of an incoming friendly URL is not valid, portal responds with the HTTP status code as defined by the portal Configuration Service property and page parameter **friendly.pathinfo.validation.errorCode**. However, depending on the configuration, portal does not always send the configured HTTP status code. Portal can identify conditions that require a different HTTP status code.

If you use the **friendly.pathinfo.validation.errorTextProvider** or the **friendly.pathinfo.validation.errorResourceBundle** settings, you can skip this information. With either of those settings, portal interrupts the request processing and sends an error response that contains only the configured HTTP status code and a localized status message if a friendly URL contains invalid path information.

If you do not use those settings to configure the friendly URL validation and if you require the portal to send the configured HTTP status code instead of 302, you need to limit the friendly URL redirects to valid friendly URLs for web content. By default, portal enforces a redirect by using the HTTP status code 302 if the incoming URL does not contain the friendly URL prefix of the addressed page or if the path information contained in the friendly URL does not match the path information from the portal state. Because the friendly URL validation component removes the path information from the state if it is not valid, the portal sends a 302 status code instead of the configured HTTP status code. Use the following combination of portal Configuration Service properties to prevent those friendly URL redirects when a friendly URL for web content is not valid:

- **friendly.redirect.enabled**=false
- **friendly.pathinfo.validation.redirect.onsuccess.enabled**=true

For more information about these properties, see *Configuration Service* in the related links.

If you limit friendly URL redirects, set the portal theme parameter **com.ibm.portal.theme.hasBaseURL** to true. If you have a custom theme, ensure that the theme writes the HTML **base** tag. Update the theme parameter by using the XML configuration interface as the following sample XML script illustrates:

```
<?xml version="1.0" encoding="UTG-8"?>

<request
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
type="update">

 <!-- This sample sets the hasBaseURL parameter in the Portal 8.5 Theme. -->
 <portal action="locate">
  <theme action="update" uniquename="ibm.portal.85Theme" >
   <parameter name="com.ibm.portal.theme.hasBaseURL"
    type="string" update="set">true</parameter>
  </theme>
 </portal>
</request>
```

Related reference:

"Configuration Service" on page 297
The portal Configuration Service is responsible for collecting the most essential configuration data of the IBM WebSphere Portal Express engine.

Friendly URL for web content example:

This example demonstrates how friendly URLs for web content work with multiple web content viewers on a single portal page. The example describes the portal page structure referenced by the friendly URLs and explains the underlying structure of the content in an IBM Web Content Manager site framework.

The example also describes the configuration of the web content viewers.

CF03 Starting with the combined cumulative fix readme 03 (CF03), the example also describes the use of the function to validate friendly URLs for web content. This example assumes that WebSphere Portal Express is configured to support friendly URLs for web content and their validation.

Elements in the example

The example is composed of the following elements:

Portal page hierarchy

The portal page hierarchy in this example is:

```
Content Root
     > Home
          > Human Resources
     > Missing Content
```

Note: CF03 The page Missing Content applies only to the combined cumulative fix readme 03 (CF03).

The page Home has a friendly URL name of home, and the page Human Resources has a friendly URL name of hr. The pages can be accessed directly using the following friendly URLs for pages:

- http://www.example.com:10039/wps/portal/home
- http://www.example.com:10039/wps/portal/home/hr

CF03 The following page parameters apply to the following pages in the portal page hierarchy:

- The page Missing Content has the unique name missing.content and is hidden from the navigation by setting the **com.ibm.portal.Hidden** page parameter to true.
- The page Content Root has the **friendly.pathinfo.validation.errorURI** page parameter set to nm:oid:missing.content.
- The page Human Resources has the **friendly.pathinfo.validation.errorContentPath** page parameter set to /Web Content/Home/Human Resources/HR Default.

Web Content Manager site framework

The Web Content Manager site framework resembles the portal page hierarchy:

```
Web Content (Web Content Library)
     > Home (site area)
          > Human Resources (site area)
               > HR Welcome (content item)
               > Health (site area)
                    > Workplace Safety (content item)
                    > Personal Wellness (content item)
               > HR Default (content item)
     > Missing Content (site area)
          > Content Not Found (content item)
     > HR Menu (menu component)
```

Note: CF03 The pages HR Default, Missing Content, and Content Not Found apply only to the combined cumulative fix readme 03 (CF03).

The content items in the Home site area can be referenced by the following content paths:

- Web Content/home/human resources/hr welcome
- Web Content/home/human resources/health/workplace safety
- Web Content/home/human resources/health/personal wellness
- CF03 Web Content/home/human resources/hr default

CF03 The site area Missing Content stores a default content item that is displayed on the corresponding page by default.

The menu component HR Menu is defined to display content from the human resources site area and the health site area.

Content association

The portal page Human Resources has the default content association to the Web Content/home/human resources site area.

CF03 The portal page Missing Content has the default content association to the Web Content/missing content site area.

Web content viewers

The page Human Resources contains two instances of the web content viewer, Web Content Viewer A and Web Content Viewer B.

- The Web Content Viewer A viewer renders the menu component HR Menu and is configured to broadcast links to this portal page.
- The Web Content Viewer B viewer inherits the content to display from the content association defined for the page Human Resources. The viewer is configured to receive links from other portlets and from itself.

CF03 The page Missing Content contains one instance of the web content viewer. The viewer inherits the content to display from the content association defined for the page Missing Content. The viewer is configured to receive links from other portlets and from itself.

Browsing the example content

With the portal page and web content site framework defined, browsing the content demonstrates how the different elements interact:

1. Navigate to Human Resources page for the first time.
   - The URL that is displayed in the address bar of the browser is http://www.example.com:10039/wps/portal/home/hr/!ut/p/b1/...
   - The URL reflects the friendly URL names of the portal pages Home and Human Resources.
   - The Web Content Viewer A viewer renders the menu component and displays links to the content items HR Welcome, Workplace Safety, and Personal Wellness.
   - The Web Content Viewer B viewer shows the default content item HR Welcome from the site area Human Resources, because of the content association defined on the portal page.

     Note: When the portal page is first displayed, the path of the default content item is not included in the friendly URL.
2. Click **Workplace Safety** from the list of content items.
   - The URL that is displayed in the address bar in the browser is http://www.example.com:10039/wps/portal/home/hr/health/workplace%20safety/!ut/p/b1/...
   - The Web Content Viewer B viewer displays content item Workplace Safety.
   - The URL is adjusted so that the path to the content item (health/workplace%20safety) becomes part of the URL.
3. Click **HR Welcome** from the list of content items.
   - The URL that is displayed in the address bar of the browser is http://www.example.com:10039/wps/portal/home/hr/hr%20welcome/!ut/p/b1/...
   - The Web Content Viewer B viewer displays the content item HR Welcome again, giving the same result as when the portal page was viewed for the first time.

- Because the Web Content Viewer A viewer is broadcasting the link to the content item, the URL that is displayed in the browser is updated to reference the path to the content item (hr%20welcome).

Content item references with friendly URLs for web content

The URL displayed in the web browser can sometimes include the content item path when you browse pages and content with web content viewers. However, you can also reference content items directly in friendly URLs for web content.

For example, to reference the content items HR Welcome, Workplace Safety, and Personal Wellness in the context of the Human Resources page, you would use the following friendly URLs for web content:

- http://www.example.com:10039/wps/portal/home/hr/hr%20welcome
- http://www.example.com:10039/wps/portal/home/hr/health/workplace%20safety
- http://www.example.com:10039/wps/portal/home/hr/health/personal%20wellness

Note: These friendly URLs for web content include URL-encoded space characters (%20) instead of unencoded space characters. Your web browser might accept unencoded space characters when specifying content item names in friendly URLs for web content. However, to ensure consistent behavior from the portal, use the URL-encoded value.

CF03

Testing invalid friendly URLs for web content

Sometimes, the friendly URL entered by a user or requested by a search crawler can be wrong. The following table shows friendly URLs that contain path information that does not identify an available content item and an explanation of the portal response when an invalid friendly URL is requested.

Invalid friendly URL for web content: http://www.example.com:10039/wps/portal/home/products/appliances

Portal response:

HTTP Status code
404

Page Missing Content

Web content
Content Not Found

Explanation:

Resolving the page
The last path segment of the friendly URL that portal can match to a friendly URL name is home. Therefore, the resolved page is Home and /products/appliances becomes the path information of the request.

Validating the path information
When validating the friendly URL for web content, portal assembles the content path to validate by appending the path information to the path of the default content association of the

resolved page. As the page Home does not have a default content association to construct the content path, the friendly URL for web content is considered invalid.

Handling the invalid path information
The HTTP status code of the response is set to the default value (`404`) because **`friendly.pathinfo.validation.errorCode`** is not set. The resolved page (Home) inherits the **`friendly.pathinfo.validation.errorURI`** setting from Content Root. Therefore, the portal resolves the URI `nm:oid:missing.content` that addresses the page Missing Content. The Web Content Viewer portlet on the page Missing Content uses the default content mapping of the page to determine the web content to render (`Content Not Found`).

Invalid friendly URL for web content: `http://www.example.com:10039/wps/portal/home/hr/group incentives`
Portal response:

HTTP Status code
`404`

Page Missing Content

Web content
`HR Default`

Explanation:

Resolving the page
The last path segment of the friendly URL that portal can match to a friendly URL name is `hr`. Therefore, the resolved page is Human Resources and `/group incentives` becomes the path information of the request.

Validating the path information
When validating the friendly URL for web content, portal assembles the content path to validate by appending the path information to the path of the default content association of the resolved page. The result for the page Human Resources is `/Web Content/home/human resources/group incentives`. As there is no content item for the computed content path, the friendly URL for web content is considered invalid.

Handling the invalid path information
The HTTP status code of the response is set to the default value (`404`), because **`friendly.pathinfo.validation.errorCode`** is not set. The resolved page (Human Resources) inherits the **`friendly.pathinfo.validation.errorURI`** setting from Content Root. Therefore, portal resolves the URI `nm:oid:missing.content` that addresses the page Missing Content. Additionally, portal sets the public Web Content Manager context as defined by the page parameter **`friendly.pathinfo.validation.errorContentPath`** of the resolved page (`/Web Content/Home/Human Resources/HR Default`).

Invalid friendly URL for web content: `http://www.example.com:10039/wps/portal/home/hr/health/medical prevention and rehabilitation`
Portal response:

HTTP Status code
`404`

Page Missing Content

Web content
`HR Default`

Explanation:

Resolving the page
The last path segment of the friendly URL that portal can match to a friendly URL name is `hr`. Therefore, the resolved page is Human Resources and `/health/medical prevention and rehabilitation` becomes the path information of the request.

Validating the path information
When validating the friendly URL for web content, portal assembles the content path to validate by appending the path information to the path of the default content association of the resolved page. The result for the page Human Resources is `/Web Content/home/human resources/health/medical prevention and rehabilitation`. As there is no content item for the computed content path, the friendly URL for web content is considered invalid.

Handling the invalid path information
The HTTP status code of the response is set to the default value (`404`) because **`friendly.pathinfo.validation.errorCode`** is not set. The resolved page (Human Resources) inherits the **`friendly.pathinfo.validation.errorURI`** setting from Content Root. Therefore, the portal resolves the URI `nm:oid:missing.content` that addresses the page Missing Content. Additionally, the portal sets the public Web Content Manager context as defined by the page parameter **`friendly.pathinfo.validation.errorContentPath`** of the resolved page (`/Web Content/Home/Human Resources/HR Default`).

Related tasks:

"Hiding and displaying pages in the navigation" on page 1803
By default, pages that you create are displayed in the navigation of the portal site. If you do not want a page that you create to appear in the navigation, you can hide the page by setting the **`com.ibm.portal.Hidden`** page parameter to `true`. While this parameter does not affect your portal access control settings for the page, it is hidden from the navigation.

Related information:

Advanced options

Managing custom unique names

Setting up a web content fallback page:

Set up a web content fallback page to be used when a web content viewer cannot determine which page to use to display a content item. The fallback page can also be used when users do not have sufficient privileges to view the page originally associated with the content item.

About this task

Note: Although a fallback page is one way of handling failed page resolution, you can also use a content page resolution filter to perform more advanced resolution.

With these filters, you can tailor how the viewer behaves if no page can be found for a content item. For more information, see *Creating a content page resolution filter class.*

Procedure

1. Create the portal page to be used as the web content fallback page.
   - Specify a unique name for the page so that you can reference the page later.
   - Assign any access rights required for users. For example, if the fallback page is available in the public part of the portal, ensure that anonymous users have view access to the page.
2. Add a web content viewer to the fallback page.
   a. Click the **Edit Page Layout** icon (small pencil) for the new page.
   b. Click **Add portlets** and select **Web Content Viewer** from the list of portlets.
3. Update the WebSphere Portal Express configuration service to enable the fallback page.
   a. Log in to the WebSphere Integrated Solutions Console..
   b. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
   c. Click **WP ConfigService**.
   d. Under **Additional Properties**, click **Custom Properties**.
   e. Click **New**, and enter the property name `wcm.fallback.page`. Set the string value to the unique name or object ID of the portal page that you created as the fallback page.
   f. Save the changes to the master configuration.
4. Restart the portal.

Results

The web content fallback page is displayed when a viewer is configured with a link broadcast setting of **Dynamically select a web content page** and one of the following conditions occurs:

- The web content viewer cannot determine which page to use to display the linked content item.
- The web content viewer identifies the page associated with the content item, but the user does not have sufficient privileges to view that page.

Related concepts:

"Dynamic web content page selection" on page 2085
The web content viewer provides a link broadcasting feature that dynamically determines the best web content page to use when rendering the linked content item. To perform this page selection, the viewer uses the content associations on the web content page.

Related tasks:

Creating a page from the site toolbar
A page displays content, such as portlets and other pages. Pages organize your site information. As you create pages, you also create new navigational elements to the site. You can create a page under an existing page or you can create a page that is a peer to an existing page. When you create a page you can also reference an existing page, apply a layout, and select supported markups.

"Creating a content page resolution filter class" on page 3231
A content page resolution filter is used to customize the behavior of the content page resolution filter chain. This method is used to tailor the response to a web content request in several ways, including overriding the content item that is displayed or the portal page that is used to display a content item in the web content viewer.

Enabling page-based access control for web content pages:

Typically, when you render content items in a web content viewer, access control enforcement on those content items is handled by IBM Web Content Manager. However, you can use page-based access control to delegate access control enforcement to the web content page that is used to display the content.

About this task

Page-based access control can be specified for individual content associations of a page. When page-based access control is enabled for a specific association, Web Content Manager does not perform additional access control checks when the associated site area contains the rendered content. This content includes not only direct children of the associated site area but also any nested site areas and their related content. Instead, the access control enforcement layer assumes that the same level of access that is granted on the page also applies to the rendered content. The affected content also includes any content items or components that are loaded during the rendering of the target content item. As a result, rendering performance is improved. If the viewer renders content that is not contained in the associated site area, normal access control enforcement is performed.

Page-based access control affects only the access control enforcement that is triggered by the web content viewer. Other access paths, such as the access control that is used by the search crawler, are not affected. In addition, Web Content Manager resources like images that are loaded directly by the browser are also not affected.

Context processor plug-ins note: Context processor plug-ins that are used with the web content viewer run in the access control context of the initial page request. Changes made to the rendering context that might affect whether page-based access control is enabled are evaluated after all other context processor plug-ins complete their configuration.

Procedure

1. To enable page-based access control, ensure that the web content page is associated with a site area in the web content system. You can manage content associations using the site toolbar or the Page Properties portlet.
2. For each content association for which you want to enable page-based access control, select **Use Portal Page Security** for the association.

   Note: If the user creating the page does not have sufficient permissions to enable page-based access control, the check box is disabled. The following access rights must be defined:
   - Administrator @ *wcm_library*, where *wcm_library* represents the library containing the content that is associated to the web content page.
   - Administrator @ CONTENT_MAPPINGS
   - Editor @ *wcm_page*, where *wcm_page* represents the web content page for which you want to enable page-based access.

Related tasks:

"Creating a web content page" on page 2035
A web content page is a page that is associated to one or more site areas in IBM Web Content Manager. You can create a web content page from a web content template page, or you can convert an existing portal page into a web content page.

Previewing content on web content pages:

When working with content in the authoring portlet, you can preview content items in a portal environment as part of a web content page. To preview content items, there must be a web content viewer on the associated web content page, and the viewer must be configured to receive links.

About this task

This preview shows the content item in a more accurate context, as it would be displayed to users. There are two ways to preview content items in a portal:

- You can use dynamic page selection. In this case, the portal determines the best match for the web content page to use for previewing the content item. The page selection is based on the content associations between the site area that contains the content item and the available web content pages.
- You can specify individual pages for previewing content.

Procedure

1. Enable the preview function in the authoring portlet. This step must be done only once.
   a. Select either **Edit Shared Settings** or **Configure** from the menu in the title bar of the authoring portlet.
   b. In the **Previewing Options** section, indicate the previewing method that you want to use.
      - To use dynamic page selection, select **Allow authors to preview content dynamically in a Web Content portal page**.
      - To preview content items on specific pages, select the pages to use for previewing from the list with the following heading: **Allow authors to preview content in the local portal pages**.
2. Preview the content as a web content page.
   a. Select a content item in the authoring portlet.
   b. Click **Preview**, and then select **Preview as a Web Content portal page**. A separate browser window opens to show the content item on its associated web content page.

Adding HTML meta tags for Search Engine Optimization:

Search engine optimization (SEO) focuses on improving the visibility of a page or website in search engine results. A basic technique of SEO is adding HTML title and meta tags to your page source. These meta tags are used to define description information and other metadata that search web engines and crawlers can use when they create search indexes and collections. When you include content in a page with a web content viewer, you can improve the search engine optimization of the page by adding title and meta tags with values derived from the web content itself.

About this task

Note: This support is available with combined cumulative fix 12 for IBM Web Content Manager Version 7.

By default, the HTML title for a page is defined by the page title in the portal. However, when you add a web content viewer to a page to render web content, you can override the value that is used for the HTML title. For metadata tags to be included as portlet preferences, you must set a custom HTML title so that only one portlet can contribute the metadata to the head section.

With the **Page Display Title** field in the portlet settings for the viewer, you can define an HTML title that better reflects the content on the page. You can even have the viewer pull the title directly from the rendered content.

Note: Although multiple web content viewers on the same page can set meta tag values, this practice does not necessarily result in improved SEO. This issue can be further complicated when multiple viewers set different values for the same meta tag name. When you have multiple viewers on the same page, select the viewer whose content best represents what the page is about. You can then use that viewer to define a new HTML title and any meta tags.

Procedure

To override the HTML title for a page and set meta tags, complete the following steps.

1. Select one web content viewer to be the primary viewer on the page. Click **Edit Shared Settings**, and select a value for the **Page Display Title** field in the portlet settings for the viewer.

   To override the HTML title, you must select a value other than **Use default title**. If you want the title value to come directly from the web content that is rendered by the viewer, select **Select from content**. This setting uses the value of the **Display title** field for the content item in Web Content Manager.

   After you save the changes, the page header is updated with the new title value. For example:

   ```
   <head>
      <title>Display title of the rendered web content</title>
   </head>
   ```

2. Create portlet preferences for each meta tag that you want to add to the page header. Each meta tag is defined by a pair of portlet preferences:

   - `meta.tag.name.`*`suffix`* identifies the name of the meta tag (for example, `keywords`).
   - `meta.tag.content.`*`suffix`* identifies the value of the meta tag.

   You can also define a specific attribute for the meta tag with the following portal preference: `meta.tag.attribute.`*`suffix`*`.`

   The *suffix* portion of each preference is used to associate a name preference with its related value preference. The suffix can be any value while it is unique across the preferences.

   There are two ways you can add portlet preferences:

   - The **Manage Portlets** portlet of the administration interface. Locate the instance of the web content viewer you want to modify, and select the **Configure portlet** icon.

- The XML configuration interface. Export the page that contains the instance of the web content viewer you want to modify. Edit the exported XML file with the meta tags you want to add, and update the page by using the XML file along with the **xmlaccess** command.

If you do not set a portlet preference for the attribute name, the attribute name "name" is used by default.

a. Specify the portal preference for the name of the meta tag. The meta tag name takes the following format:

   `meta.tag.name.suffix=name`

   If you want to specify an attribute other than the name attribute, you can define the attribute name with the following format:

   `meta.tag.attribute.suffix=attribute_name`

   For example, to add the following meta tag with the name `keywords`:

   ```
   <meta name="keywords" content=""/>
   ```

   Specify the following preference:

   ```
   meta.tag.name.1=keywords
   ```

   To add the following meta tag with the `http-equiv` attribute:

   ```
   <meta http-equiv="content-language" content="en-US"/>
   ```

   Specify the following preference:

   ```
   meta.tag.attribute.1=http-equiv
   ```

b. Specify the portal preference for the value of the meta tag. The value of the meta tag can be specified in three ways:
   - You can explicitly enter text for the meta tag value.
   - The meta tag value can be derived from the value of a text element in the rendered web content.
   - The meta tag value can be derived from properties that contain information about the rendered web content.

   Depending on how you want to specify the meta tag value, different portlet preferences are required. Only one value can be specified per suffix.

   Use preset text

   The meta tag value takes the following format:

   `meta.tag.content.text.suffix=text`

   The *suffix* portion must match the *suffix* value of the associated `meta.tag.name.suffix` preference. The *text* portion indicates the text to use for the meta tag value.

   Use the value of an element

   The meta tag value takes the following format:

   `meta.tag.content.element.suffix=name_of_element`

   The *suffix* portion must match the *suffix* value of the associated `meta.tag.name.suffix` preference. The *name_of_element* portion indicates the name of the element from the web content that is rendered.

Table 327. Elements for populating meta tag values

Element	Meta tag value
Text component	Text of the element
Date component	Date of the element
Image component	URL of the image
File component	URL of the file

Use a property

The meta tag value takes the following format:

`meta.tag.content.property.`*`suffix=property`*

The *suffix* portion must match the *suffix* value of the associated `meta.tag.name.`*`suffix`* preference. The *property* portion indicates the property that contains information about the web content that is rendered. The properties are associated with fields on the rendered content.

Table 328. Properties for populating meta tag values

Property	Meta tag value
`AdditionalViewers`	Name of additional viewers
`Authors`	Display names of the authors of the rendered content
`authtemplatename`	Name of the authoring template of the rendered content
`authtemplatetitle`	Display title of the authoring template of the rendered content
`Categories`	Titles of any categories associated with the rendered content
`CreationDate`	Creation date of the rendered content
`Creator`	Display name of the user who created the rendered content
`CurrentStage`	Name of the current workflow stage of the rendered content
`Description`	Localized description of the rendered content
`ExpiryDate`	Expiration date of the rendered content
`ID`	ID of the rendered content
`GeneralDateOne`	Date from the general date one field
`GeneralDateTwo`	Date from the general date two field
`Keywords`	Keywords associated with the rendered content
`LastModifiedDate`	Date that the rendered content was last modified
`LastModifier`	Display name of the user who made the last change to the rendered content
`Name`	Name of the rendered content
`Owners`	Display names of the owners of the rendered content

Table 328. Properties for populating meta tag values (continued)

Property	Meta tag value
`PublishDate`	Date the rendered content was published
`SitePath`	Site path of the rendered content
`Status`	Workflow status of the rendered content
`Title`	Localized title of the rendered content
`Workflow`	Name of the workflow of the rendered content

For several of the most common meta tags, default values are predefined. For these meta tags, you can create the portlet preference for only the meta tag name. The meta tag value is provided automatically, without the need for a corresponding name preference. The following meta tags have default values:

`Author` The default value is a list of the authors of the rendered content.

`Keywords`
The default value is list of any keywords that are associated with the rendered content.

`Description`
The default value is the localized description of the rendered content.

If you do not want to use the default value, you can set the value using one of the methods previously described.

c. Optional: If the value of the meta tag requires a scheme attribute, specify the scheme attribute with the `meta.tag.scheme.`*`suffix`* preference. The meta tag scheme attribute takes the following format:

```
meta.tag.scheme.suffix=attribute_value
```

For example, to add the following scheme attribute with the value `W3CDTF`:

```
<meta name="DC.date" content="2000-01-01T12:00+00:00" scheme="W3CDTF"/>
```

Specify the following preference:

```
meta.tag.scheme.1=W3CDTF
```

The format and scheme that are used to write date elements and content properties that are related to date and time information, such as the `LastModifiedDate` property, depends on the meta tag attribute name. By default, all date and time information is formatted according to the date format defined by the HTTP specification. The format that is used to write date and time information in other meta tags is the data and time format that is recommended by the World Wide Web Consortium (W3C) under the scheme named W3CDTF.

Examples

The following examples demonstrate the different ways of specifying portlet preferences and the resulting meta tags in the output.

- Setting the meta tag value with the user who created the rendered content:

```
meta.tag.name.1=DC.creator
meta.tag.content.property.1=Creator
```

Result:

```
<meta name="DC.creator" content="content admin"/>
```

- Setting the meta tag value with preset text:

  ```
  meta.tag.name.1=DC.publisher
  meta.tag.content.text.1=IBM
  ```

  Result:

  ```
  <meta name="DC.publisher" content="IBM"/>
  ```

- Setting multiple meta tag values with the default value for the author and the value of the text element `descelement` in the rendered content:

  ```
  meta.tag.name.1=author
  meta.tag.name.2=description
  meta.tag.content.element.2=descelement
  ```

  Result:

  ```
  <meta name="author" content="content author"/>
  <meta name="description" content="Information about IBM"/>
  ```

- Setting the meta tag with an `http-equiv` attribute and a value of the date that the rendered content was last modified.

  ```
  meta.tag.name.1=last-modified
  meta.tag.attribute.1=http-equiv
  meta.tag.content.property.1=LastModifiedDate
  ```

  Result:

  ```
  <meta http-equiv="last-modified" content="Mon, 01 Aug 2011 13:45:57 GMT"/>
  ```

- Setting the meta tag and with a scheme attribute and a value of the date that the rendered content was published.

  ```
  meta.tag.name.1=DC.date
  meta.tag.scheme.1=W3CDTF
  meta.tag.content.property.1=PublishDate
  ```

  Result:

  ```
  <meta name="DC.date" content="2011-08-01T08:15:30+02:00" scheme="W3CDTF"/>
  ```

Improving page loading performance with asynchronous web content rendering: CF06

You can increase page loading performance by separating portal page content delivery from web content rendering. To do so, you use the asynchronous web content rendering feature. If you enable asynchronous web content rendering, the portal serves the page immediately, but only with placeholder content. The Web Content Viewer then dynamically inserts the configured content into the portal page after the content has been completely rendered.

About this task

Asynchronous web content rendering separates the web content delivery into two steps:

1. First, IBM WebSphere Portal Express serves the page with a predefined static markup instead of the web content. This so called bootstrap markup then asynchronously triggers the second stage by using a resource URL.
2. In a second step, the Web Content Viewer portlet renders the requested web content, which then is inserted into the correct placeholder of the current portal page. For this step, the Web Content Viewer portlet uses the `XmlHttpRequest` function of the web browser and JSR 286 compliant serving of resources.

Asynchronous web content rendering is intended to be used for web content that needs a long time to be created. Examples are web content that is remotely served or content that is created by extensive back-end operations. Such content increases the total time for loading a portal page because the portal needs to aggregate all portlets on a page. Therefore, the user experience improves by selectively enabling asynchronous web content rendering for Web Content Viewer portlets that take a long time to render. Whether asynchronous web content rendering is useful depends on the page composition and the web content. The site administrator needs to evaluate it for each individual portlet.

Notes:

- By default, asynchronous web content rendering is disabled for all instances of the Web Content Viewer portlet. You enable it by using the Edit Shared Settings or Configure mode.
- If you enable asynchronous rendering, the Web Content Viewer portlet uses it only while the portal is in View mode.
- The portal does not use asynchronous web content rendering for requests by search crawlers. This way, the portal makes sure that web content is correctly analyzed and used by search engines.

  CF06 "Enabling asynchronous web content rendering"
  You enable asynchronous web content rendering by using the Edit Shared Settings or Configure mode.

  CF06 "Customizing loading markup for asynchronous web content rendering" on page 2067
  To adjust the loading markup to the user experience, you can customize it. For example, you can use a company-specific loading icon or text.

  CF06 "Using onPageLoad scenarios with asynchronous web content rendering" on page 2068
  Separating the page delivery from the web content delivery can increase the page loading time. However, all JavaScript functions that rely on the `onPageLoad` functions can access only the bootstrapping markup, but not the rendered web content markup. The web content markup is injected into the page as soon as it is ready.

  CF06 "Asynchronous web content rendering and caching" on page 2069
  The Web Content Viewer portlet supports setting specific portlet fragment caching configuration values. This caching also works with asynchronous web content rendering. However, the respective content for the fragment cache is different.

Enabling asynchronous web content rendering: CF06

You enable asynchronous web content rendering by using the Edit Shared Settings or Configure mode.

Before you begin

To enable asynchronous web content rendering, a user needs sufficient access rights: At least either Manager role privileges on the page with the Web Content Viewer or Manager role privileges on the Web Content Viewer portlet itself.

About this task

To enable asynchronous web content rendering for a single portlet instance, use the following procedure.

Procedure

1. Go to the page that contains the Web Content Viewer portlet for which you want to enable asynchronous web content rendering.
2. Change to **Edit** mode.
3. Open the display menu of the portlet.
4. Select **Edit Shared Settings** to enter the configuration mode.
5. Expand the **Advanced Options** section if it is not expanded yet.
6. Scroll down to the **Asynchronous Web Content Rendering** section.
7. To enable asynchronous web content rendering for this portlet, select the check box in the Asynchronous Web Content Rendering subsection.
8. Click **OK** to save and leave the Edit Shared Settings mode.

Optionally, you can also set a timeout for the loading icon for asynchronous web content rendering. If you enable this timeout, the portal marks web content as unavailable, if it takes too long to render. To enable the timeout, proceed as follows:

9. Select **Edit Shared Settings** to enter the configuration mode.
10. Expand the **Advanced Options** section if it is not expanded yet.
11. Scroll down to the **Asynchronous Web Content Rendering** section.
12. Specify the timeout in milliseconds for the asynchronous request. Increase or decrease the timeout value to adjust it to the use case and to the back-end turn around times. To disable the timeout check completely, set the timeout to the value zero (`0`). The asynchronous request then waits until a page redirect occurs or the browser window is closed.
13. Change from Edit mode to View mode.

Results

The web content now renders asynchronously. Until the web content is available, a loading icon and corresponding text appears.

What to do next

Note: Be aware of the following possible error scenarios with asynchronous web content rendering:

- If a timeout occurs, the placeholder area of the Web Content Viewer removes the waiting icon and displays an error text instead.
- If the request fails on the server side inside the `serveResource` method, an HTTP error code is displayed in the placeholder area with a corresponding IBM Web Content Manager error message.
- Issues in the web browser such as missing libraries or theme profiles are displayed in a similar way to indicate the problem.

Customizing loading markup for asynchronous web content rendering: CF06

To adjust the loading markup to the user experience, you can customize it. For example, you can use a company-specific loading icon or text.

About this task

To do so, use the following instructions.

Procedure

1. Create an IBM Web Content Manager HTML component in a Web Content Manager library. The body of this newly created Web Content Manager HTML component serves as the custom loader markup and is delivered by the Web Content Viewer portlet.
2. To notify the portlet to use a custom markup rather than the default content, set the page parameter `wcm.custom.asynchronous.rendering.loader`. As the value, specify the path to the HTML component. All Web Content Viewer portlets on that page then use the custom markup from the HTML component.
3. To address the configured web content, create the JSR 286 compliant Resource URL. To do so, use the `PortletResourceURL` rendering plug-in. This resource URL triggers the `serveResource` method of the Web Content Viewer portlet and returns the rendered web content.
4. Use this markup to update the HTML document object model. The `PortletResourceURL` supports the attributes that you can use with resource URLs, such as the resource identifier`id` or `cacheability`. For the custom asynchronous web content rendering scenario, you do not need to define any attributes. As more than one instance of this custom loading markup on a page is possible, use the `namespace` attribute of the portlet rendering plug-in to create unique identifiers.

Example

The following content is an example of a custom loader markup:

```
<div>
   <a href="[Plugin:PortletResourceURL]" id="[Plugin:Portlet key="namespace"]customAsyncRenderingHook"></a>
   <div style="width: 100%; text-align: center; padding-top: 5em;">
      <span>Loading web content...</span>
   </div>
</div>

<script type="text/javascript">
(function() {
   var hook = document.getElementById("[Plugin:Portlet key="namespace"]customAsyncRenderingHook");
   var xhr = new XMLHttpRequest();
   xhr.onreadystatechange = function() {
      if (xhr.readyState==this.DONE) {
         if (xhr.status==200) {
            hook.parentNode.innerHTML = xhr.responseText;
         } else if (xhr.status==0) {
            // implement timeout handling here ...
         } else {
            // implement error handling here
         }
      }
   };

   xhr.open("GET", hook.href);
   xhr.timeout = 5000;
   xhr.ontimeout = function() { console.error('Asynchronous rendering timed out.'); };
   xhr.send();
}());
</script>
```

Using onPageLoad scenarios with asynchronous web content rendering: CF06

Separating the page delivery from the web content delivery can increase the page loading time. However, all JavaScript functions that rely on the `onPageLoad` functions can access only the bootstrapping markup, but not the rendered web content markup. The web content markup is injected into the page as soon as it is ready.

About this task

In some scenarios, custom JavaScript code needs to run after the asynchronously loaded markup is ready. For such cases, IBM WebSphere Portal Express provides a

global object called wp_wcm_async. This object becomes available with the wp_wcm_async theme modules. To register custom functions to different scenarios, you can use either of the two following two functions:

- To register a function that is called after all asynchronous items on a page are rendered, use addOnPageLoad.
- To register a function that is called after the asynchronous item that is identified by the key is loaded, use addOnAsyncSpotLoad.

To uniquely identify the asynchronous item, use the portlet namespace for the key. Custom theme profiles need to add a moduleID dependency to the new asynchronous web content rendering theme module. In a default WebSphere Portal Express V 8.5 CF06 installation, the asynchronous rendering theme module is included in the Basic Content and Basic Content with Dojo profiles.

Asynchronous web content rendering and caching:

The Web Content Viewer portlet supports setting specific portlet fragment caching configuration values. This caching also works with asynchronous web content rendering. However, the respective content for the fragment cache is different.

If servlet fragment caching, portlet fragment caching, Web Content Viewer caching, and asynchronous web content rendering are all enabled, the Web Content Viewer portlet adds the default or custom loader markup to the fragment cache. It does not add the asynchronously rendered web content to this fragment cache. This way, it increases the possibility to cache an elaborated page setup even with user-specific web content. The reason is mainly because the user-specific content is added asynchronously. The initial page markup contains only the placeholder loading markup, which is static and can therefore be cached easily.

Enabling remote rendering with WSRP and the Web Content Viewer

To display web content on a portal that does not include IBM Web Content Manager, you can use the Web Content Viewer and the WSRP support in the portal. The Web Content Viewer can then retrieve and display content from a web content system on a different server.

About this task

When to use remote rendering: The preferred way to render content on one server from another server is to syndicate the content to the delivery server. On the delivery server, you can then locally render the content with a Web Content Viewer. However, remote rendering with WSRP is appropriate for service-oriented architecture (SOA) scenarios where you want to incorporate specific pieces of content into your website.

When you use the Web Content Viewer for remote rendering with WSRP, the following conditions apply:

- The remote web content server acts as the WSRP Producer.
- The portal with the Web Content Viewer acts as the WSRP Consumer.

Note: Remote rendering with WSRP is supported only when you render content from the default virtual portal.

Procedure

1. Set up the WSRP environment between the Producer portal and the Consumer portal, as described in *WSRP Services*. If you plan to use the **Edit Shared Settings** mode or the **Configure** mode in the portlet with WSRP, configure web service security between the Producer and the Consumer portals.
2. You might have basic authentication challenges when you configure the portlet on the Consumer. To avoid these challenges, customize the WSRP resource proxy for LTPA token forwarding. Go to *Customizing the WSRP resource proxy for LTPA token forwarding* for information.
3. Provide the Web Content Viewer Portlet as a WSRP service hosted on the remote web content server acting as the WSRP Producer.
4. Consume the remote Web Content Viewer that is provided as a WSRP service on the portal acting as the WSRP Consumer.
5. Configure the Web Content Viewer to display content, just as you would configure a local Web Content Viewer.

   When you use the viewer with WSRP, settings for selecting content from a web content library show content from the remote web content system.

   Note: Depending on the configuration of the Web Content Viewer, resources like resource bundle files or content processor plug-ins might be required. In such cases, the resources must be available on the remote web content server acting as the WSRP Producer.

   Limitations when using WSRP with the Web Content Viewer:

   - Because the concept of pages and web content pages does not exist in WSRP, you cannot use the dynamic link broadcasting feature with web content pages. When you specify how to broadcast links, do not select **Dynamically select a web content page** in the **Broadcast links to** field. Selecting this option has the same effect as broadcasting links to the current page.
   - Web content inline editing for your web content is not supported with WSRP.
   - The use of remote authoring action URLs in your web content is not supported with WSRP.
   - Tagging and rating for web content is not supported with WSRP.
   - Personalization elements are not supported with WSRP. Therefore, you cannot use features that require personalization rules. Examples: content targeting, federated documents, social rendering, and IBM Digital Data Connector (DDC) for WebSphere Portal Express.

   Limitations when using WSRP with the Web Content Viewer with other non-IBM WebSphere Portal WSRP Consumers: The link broadcasting feature of the viewer is not supported for the WSRP Consumers of other vendors. This limitation is because the concept of pages and web content pages does not exist in WSRP. When you specify how to broadcast links, you can select only the option **None**.

   "CORS and remote web content rendering with WSRP and the Web Content Viewer" on page 2071
   Cross-origin resource sharing (CORS) describes a mechanism for supporting requests that a web page sends to a server that is not in the same domain as the originating web page. The CORS concept must be supported by both the web browser and the server.

   "OpenAjax security and remote web content rendering with WSRP and the Web Content Viewer" on page 2073
   The Enabler from the Mashups 3.0.0.1 component that is included in IBM

WebSphere Portal Express as a theme module implements some features that are specified by the OpenAjax Alliance. One of them is a generic override for Dojo XMLHttpRequests

Related concepts:

"Web Content Viewers" on page 2017
Web Content Viewers are portlets that render content from a web content library as part of a portal page. If your presentation is simple, a single viewer can be sufficient. To provide a richer experience for your users, use multiple viewers to aggregate content from different libraries.

"WSRP services" on page 1433
IBM WebSphere Portal Express supports the Web Services for Remote Portlets (WSRP) standard. By using this standard, portals can provide portlets, applications, and content as WSRP services. Other portals can then integrate the WSRP services as remote portlets for their users.

Related tasks:

"Customizing the WSRP resource proxy for LTPA token forwarding" on page 1488
The WSRP resource proxy can forward single sign-on cookies (LTPA, LTPA2) from the client requests to resources in the same single sign-on domain.

"Providing WSRP services as a Producer" on page 1454
After you prepared your portal as a Producer portal, you can provide your portlets through WSRP. Providing portlets makes them available to Consumers. They can integrate them into their Consumer portals and use them as remote portlets. You can also withdraw portlets from being provided through WSRP. Consumer portals can then no longer use them.

"Configuring security on the Consumer portal" on page 1461
You can configure security for the WSRP Consumer. If you enable security, the WSRP Consumer sends a security token as part of the WSRP request message to the WSRP producer. The security token represents the identity of the user who is logged in to the Consumer Portal. The WSRP Producer uses the security token to process the WSRP requests under the user identity that is represented by the security token.

CORS and remote web content rendering with WSRP and the Web Content Viewer:

Cross-origin resource sharing (CORS) describes a mechanism for supporting requests that a web page sends to a server that is not in the same domain as the originating web page. The CORS concept must be supported by both the web browser and the server.

For more information about the CORS support in IBM WebSphere Portal Express, read *Manage CORS in WebSphere Portal Express*.

If you use remote rendering with WSRP and the Web Content Viewer portlet as your web content delivery model, you must make yourself familiar with CORS. In the **Edit shared settings** and **Configure** modes, a consumed Web Content Viewer portlet uses XMLHttpRequests to load information from the remote web content portal. CORS can prevent this remote connection from being successful. Usually, the remote web content portal that acts as the WSRP Producer is in a different domain than the portal with the Web Content Viewer portlet that acts as the WSRP Consumer. Therefore, the Producer portal can reject XMLHttpRequests when you try to configure the consumed Web Content Viewer portlet on your Consumer portal.

Web browsers can implement CORS in different ways or not at all. Therefore, you might experience issues only when you use a specific web browser. In case of such issues, the JavaScript console shows that requests made by the Web Content Viewer portlet result in an error with HTTP status code 403 (Forbidden). Example:
`PROPFIND http://`*`WSRP_CONSUMER_HOSTNAME`*`:`*`WSRP_CONSUMER_PORT`*`/`
`WSRP_CONSUMER_CONTEXT_ROOT``/WsrpProxyPortlet/ResourceProxy/.../`
`WSRP_PRODUCER_CONTEXT_ROOT``/mycontenthandler/dav/content/libraries/ 403 (Forbidden)`

If you experience issues when you use the **Edit shared settings** or **Configure** mode of the consumed Web Content Viewer portlet as described earlier, you can choose one of the following options:

- The best solution is to add the WSRP consumer as a trusted origin to the whitelist of the WSRP Producer. For more information, read *Manage CORS in WebSphere Portal Express*.

  Note: If you choose this option, be aware that you might need to repeat this configuration after you upgrade or migrate your WebSphere Portal Express to a newer version.
- Configure the WSRP resource proxy of the WSRP consumer to prevent it from forwarding the `Origin` HTTP header that CORS uses. If the requests do not contain the header, the remote web content portal does not reject the requests. For more information, read *Customizing the WSRP resource proxy HTTP header forwarding behavior*.

  Note: If you choose this option, make sure that you fully understand the implications of removing the `Origin` HTTP header. The target server treats all requests that are made through the WSRP resource proxy as same-origin requests, even if the target server supports CORS and normally rejects requests from that domain.
- Disable the CORS support of the WSRP Producer portal. To disable CORS, set the property `com.ibm.portal.csrf.enabled` of the portal WP Configuration Service resource environment provider to `false`. Then, restart your portal for the changes to take effect. For details, about how to set portal service configuration properties, read *Setting service configuration properties*.

  Note: If you choose this option, make sure that you fully understand the implications of disabling the CORS support. With disabled CORS support, the portal accepts all cross-origin requests that it rejects if the CORS support is enabled.

Related tasks:

"Customizing the WSRP resource proxy HTTP header forwarding behavior" on page 1490
By default, the WSRP resource proxy forwards all HTTP headers from the client request except for the host header and cookie headers. You can define further headers that you do not want to be forwarded.

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Related information:

Manage CORS in WebSphere Portal

OpenAjax security and remote web content rendering with WSRP and the Web Content Viewer:

The Enabler from the Mashups 3.0.0.1 component that is included in IBM WebSphere Portal Express as a theme module implements some features that are specified by the OpenAjax Alliance. One of them is a generic override for Dojo XMLHttpRequests

It adds the following extra HTTP request headers:

com.ibm.lotus.openajax.virtualhost
This header specifies the virtual host name that the portal uses to create absolute URLs.

com.ibm.lotus.openajax.virtualport
This header specifies the virtual host port that the portal uses to create absolute URLs.

As a consumed Web Content Viewer portlet uses Dojo XMLHttpRequests in specific situations, those HTTP request headers can cause issues. For example, to configure the portlet to render a web content element remotely, the portlet dynamically loads the elements of the selected web content item from the remote web content portal, that is the WSRP Producer. The corresponding requests include the `com.ibm.lotus.openajax.*` HTTP request headers that are mentioned before. They identify the WSRP Consumer portal that renders the web content as a virtual host. The WSRP resource proxy then uses the virtual host as the target server. As a result, the WSRP resource proxy uses the web content delivery portal (the WSRP Consumer) as the target server instead of the remote web content portal. Eventually, the WSRP resource proxy requests fail with HTTP status code `404 (Not Found)`.

If you experience issues when you configure the consumed Web Content Viewer portlet as described earlier, you can choose one of the following options:

- As the Enabler component implements the override for Dojo XMLHttpRequests, check whether your portal really requires that component. If you find that you do not need any Enabler functions on your web content delivery portal, you can change your theme and theme modules to prevent the `mm_enabler` theme module from being loaded. For more information, read *The module framework*.
- Configure the WSRP resource proxy of the WSRP Consumer to prevent it from forwarding the `com.ibm.lotus.openajax.virtualhost` and `com.ibm.lotus.openajax.virtualport` HTTP headers that are set by the Enabler component. If the headers are not present in the Dojo XMLHttpRequests, the WSRP resource proxy addresses the remote web content portal correctly. For more information, read *Customizing the WSRP resource proxy HTTP header forwarding behavior*.

Related concepts:

"The module framework" on page 2526
The module framework allows extensions to contribute to different areas of a page to provide flexibility, enhance the user experience, and maximize performance.

Related tasks:

"Customizing the WSRP resource proxy HTTP header forwarding behavior" on page 1490
By default, the WSRP resource proxy forwards all HTTP headers from the client request except for the host header and cookie headers. You can define further headers that you do not want to be forwarded.

Advanced administrative examples

You can undertake advanced administration for web content artifacts, such as web content pages and content associations.

"Creating a web content page with the XML configuration interface"
As with other portal pages, you can create a web content page with the XML configuration interface (**xmlaccess** command). Page definition is similar to a standard portal page. However, there is an additional page parameter that specifies the site area that is associated with the web content page.

"Content associations reference" on page 2076
Content associations are used to associate web content pages with your web content site structure. When you select a folder to associate with a web content page, a content association is created and maintained within the portal.

"Dynamic web content page selection" on page 2085
The web content viewer provides a link broadcasting feature that dynamically determines the best web content page to use when rendering the linked content item. To perform this page selection, the viewer uses the content associations on the web content page.

Creating a web content page with the XML configuration interface:

As with other portal pages, you can create a web content page with the XML configuration interface (**xmlaccess** command). Page definition is similar to a standard portal page. However, there is an additional page parameter that specifies the site area that is associated with the web content page.

Procedure

1. When creating your **xmlaccess** command, specify your page parameters as you would for a standard portal page.

   The following block of code is an example of specified page parameters:

   ```
   <content-node action="update" content-parentref="parentOID" domain="rel" objectid="someOID"
   preserve-old-layout="true" type="page">
     <content-mapping-info>
       <content-mapping content-id="/mylibrary2/sitearea2" default="true"
   delegated-access-level="User"/>
       <content-mapping content-id="ddccb7ed-8485-48c8-b875-31d17d9da65b"
   default="false"/>
     </content-mapping-info>
   </content-node>
   ```

   Note: The value of the content-id attribute can be either the ID or the path to the web content item. If you are using the content path, the value must begin with the forward slash character (/) followed by the library name. When creating a web content page using the content path, you cannot build the path from the **Display title** fields of the items in the path. Instead you must use the **Name** fields of the items when specifying the path.

2. Because the web content viewer uses public render parameters to identify the content to render, include the page parameter **param.sharing.scope** when creating your **xmlaccess** command. Set the value for the parameter to `ibm.portal.sharing.scope.page`.

   The following block of code is an example parameter definition for web content pages when using the XML configuration interface:

```
<parameter name="com.ibm.portal.wcm.contentroot" type="string" update="set">
   <![CDATA[/mylib/mysite/mysitearea]]>
</parameter>
<parameter name="param.sharing.scope" type="string" update="set">
   <![CDATA[ibm.portal.sharing.scope.page]]>
</parameter>
```

3. When creating your **xmlaccess** command, add at least one web content viewer that is configured to listen to other portlets and make dynamic broadcasts. Adding the viewer ensures that content selected for this page is rendered correctly and that links between pages work properly.

   The following block of code is an example of how to add the web content viewer using the XML configuration interface:

```
<component action="update" active="true" deletable="undefined" domain="rel"
modifiable="undefined" objectid="7_U796BB1A0OS250IOS7F1BP3081" ordinal="100"
orientation="H" skinref="undefined" type="container" width="undefined">
  <component action="update" active="true" deletable="undefined" domain="rel"
  modifiable="undefined" objectid="7_U796BB1A0OS250IOS7F1BP3085" ordinal="100"
  orientation="V" skinref="undefined" type="container" width="undefined">
    <component action="update" active="true" deletable="undefined" domain="rel"
    modifiable="undefined" objectid="7_U796BB1A0OS250IOS7F1BP3087" ordinal="100"
    skinref="undefined" type="control" width="undefined">
      <portletinstance action="update" domain="rel" objectid="5_U796BB1A0OS250IOS7F1BP3083"
       portletref="3_U796BB1A008OD0IOS2ODAD28U4">
        <preferences name="WCM_BROADCASTS_TO" update="set">
          <value><![CDATA[WCM_LINKING_DYNAMIC]]></value>
        </preferences>
        <preferences name="WCM_LISTENS_TO" update="set">
          <value><![CDATA[WCM_LINKING_OTHER]]></value>
        </preferences>
      </portletinstance>
    </component>
  </component>
</component>
```

   Migration note: After Version 6.1.5, the format used by the XML configuration interface to represent content associations for a web content page has changed. Typically, the migration process automatically converts all existing web content pages to the updated format. However, if you create web content pages on the older portal after migration and then import the pages to the Version 8.5 portal, the page format is incompatible. In this case, you must manually run the `action-migrate-content-mappings` configuration task on the Version 8.5 portal to convert the new web content pages to the Version 8.5 format. To perform the conversion, run the following task from the *wp_profile_root*`/ConfigEngine` directory:

   - Windows: `ConfigEngine.bat action-migrate-content-mappings -DWasPassword=password -DPortalAdminPwd=password`
   - Linux: `./ConfigEngine.sh action-migrate-content-mappings -DWasPassword=password -DPortalAdminPwd=password`
   - IBM i: `ConfigEngine.sh action-migrate-content-mappings -DWasPassword=password -DPortalAdminPwd=password`

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

"Public render parameters" on page 3070
Public render parameters allow JSR 286 portlets to share navigational state information. They are specially useful for coordinating the multiple navigation or viewer portlets that display different information items that are all related to the same parameter name. The portal stores all portlet render parameters, including public render parameters, as an encoded part of the current portal URL. Therefore, public render parameters are correctly preserved by typical browser navigation actions such as the Back button and bookmarking.

Related reference:

"REST API and content associations" on page 2080
If you are creating or extending an application and want to manage content associations with that application, you can use portal remote APIs. These APIs retrieve a list of content associations and then create, update, or delete associations.

Related information:

 Package com.ibm.portal.services.contentmapping

Content associations reference:

Content associations are used to associate web content pages with your web content site structure. When you select a folder to associate with a web content page, a content association is created and maintained within the portal.

Typically, the management of content associations is automatically handled by the portal. However, for situations where you want to work directly with content associations, several mechanisms are available: the XML configuration interface, the Portal Scripting Interface, and the REST API for content associations.

"XML configuration interface and content associations"
With the XML configuration interface (**xmlaccess** command), you can perform batch updates of content associations or export associations to import into another portal. Content association information is represented in the XML configuration schema by content-mapping-info elements.

"Portal Scripting Interface and content associations" on page 2078
With the Portal Scripting Interface, you can create scripts to automate the management of content associations. Using the ContentMapping bean with the Portal Scripting Interface, you can add, modify, and remove content associations.

"REST API and content associations" on page 2080
If you are creating or extending an application and want to manage content associations with that application, you can use portal remote APIs. These APIs retrieve a list of content associations and then create, update, or delete associations.

Related concepts:

"Web content associations" on page 2023
Web content associations are used to combine portal pages and associated web content items that are managed by IBM Web Content Manager so they can be managed and rendered consistently. Web content associations map portal pages to the site structure in the IBM Web Content Manager system.

XML configuration interface and content associations:

With the XML configuration interface (**xmlaccess** command), you can perform batch updates of content associations or export associations to import into another

portal. Content association information is represented in the XML configuration schema by content-mapping-info elements.

Export content associations

The content associations for a web content page are represented in an XML export file as nested content-mapping-info elements.

The following example represents a web content page with two content associations:

```
<content-node action="update" content-parentref="6_00000000000000000000000000A0" domain="rel"
 objectid="someOID" preserve-old-layout="true" type="label">
  <content-mapping-info>
    <content-mapping content-id="74-11" default="false"/>
    <content-mapping content-id="007" default="true" delegated-access-level="User"/>
  </content-mapping-info>
</content-node>
```

Note: If no content-mapping-info elements are present in an XML export document, there are currently no content associations defined for the web content page.

Import content associations

When importing content associations with an XML import file, associations for a web content page are represented in the content-mapping-info element for the web content page. Any content associations that are already defined for the web content page are removed when you perform the import process and replaced with the new associations.

The following example updates a web content page to have two specific content associations:

```
<content-node action="update" content-parentref="6_00000000000000000000000000A0" domain="rel"
 objectid="someOID" preserve-old-layout="true" type="label">
  <content-mapping-info>
    <content-mapping content-id="74-11" default="false" />
    <content-mapping content-id="007" default="true" delegated-access-level="User"/>
  </content-mapping-info>
</content-node>
```

Note: If no content-mapping-info element is present in an XML import document, no changes are made to the content associations currently defined for the web content page.

Delete content associations

You can delete content associations by specifying an empty content-mapping-info element in the XML import file.

The following example updates a web content page to delete any defined content associations:

```
<content-node action="update" content-parentref="6_00000000000000000000000000A0" domain="rel"
 objectid="someOID" preserve-old-layout="true" type="label">
  <content-mapping-info/>
</content-node>
```

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Portal Scripting Interface and content associations:

With the Portal Scripting Interface, you can create scripts to automate the management of content associations. Using the ContentMapping bean with the Portal Scripting Interface, you can add, modify, and remove content associations.

Note: Before a script can work with the ContentMapping bean, you must establish a user session with the portal using the **login** command of the Portal bean. The user identity must have sufficient permissions to administer the web content pages and web content library folders referenced by the script.

Retrieve content associations

To retrieve content association information, use the select method to specify the object ID of the web content page. You can often derive the object ID for a resource from another bean and use that as input for the select method. For example, you might have a web content page with the unique name `my.test.page`. Using the find method of the Content bean, you can determine the object ID of the `my.test.page` page.

Jacl example:

```
$set the_page [$Content find page uniquename "my.test.page"]
$ContentMapping select $the_page
```

Jython example:

```
the_page = Content.find('page','uniquename','my.test.page')
ContentMapping.select(the_page)
```

After you have the object ID of the web content page, you can use the list method and the get methods to access the content associations. The list method returns a list of content association IDs. The IDs can identify either the resource ID of a folder or the content path of the folder, depending on how the page is mapped. You can use the content association IDs returned by the list method as arguments for the get method.

Jacl example:

```
$set the_page [$Content find page uniquename "my.test.page"]
$ContentMapping select $the_page
foreach mid [$ContentMapping list mappings] {
  puts "  Mapping $mid info:"
  puts "  content id: [$ContentMapping get content-id $mid]"
  puts "  default? [$ContentMapping get isdefault $mid]"
  puts "  scope: [$ContentMapping get scope $mid]"
}
```

Jython example:

```
var the_page = Content.find('page','uniquename','my.test.page')
ContentMapping.select(the_page)
for mid in ContentMapping.list('mappings').split():
  print "  Mapping "+mid+" info:"
  print "  content id: "+ContentMapping.get('content-id', mid)
  print "  default? "+ContentMapping.get('isdefault', mid)
  print "  scope: "+ContentMapping.get('scope', mid)
```

The get method can return the default association for the selected web content page. The list method can retrieve a list of scopes that are defined for the associations of the web content page.

Jacl example:

```
$set the_page [$Content find page uniquename "my.test.page"]
$ContentMapping select $the_page
puts "available scopes: [$ContentMapping list scopes]"
puts "default mapping: [$ContentMapping get defaultmapping]"
puts "portal resource OID: [$ContentMapping get oid]"
```

Jython example:

```
var the_page = Content.find('page','uniquename','my.test.page')
ContentMapping.select(the_page)
print "available scopes: "+ContentMapping.list('scopes')
print "default mapping: "+ContentMapping.get('defaultmapping')
print "portal resource OID: "+ContentMapping.get('oid')
```

Add content associations

Use the add method to add new content associations to a web content page. You can assign a content association by specifying the content path of the folder or the ID of folder. If you identify the folder by content path, the association is internally transformed to actually point to the ID of the folder. As a result, if you rename the folder later, the association still points to the same folder.

Jacl example:

```
$ContentMapping select [$ContentNode find page uniquename "my.sample.page"]
$ContentMapping add content-path "/test1/mapping"
set the_content_id  .... ## obtain ID of content to be mapped
$ContentMapping add id $the_content_id
```

Jython example:

```
ContentMapping.select(Content.find('all','un','my.sample.page'))
ContentMapping.add('content-path','/test1/mapping')
var the_content_id = ... ## obtain ID of content to be mapped
ContentMapping.add('id',the_content_id)
```

Remove content associations

The ContentMapping bean provides two methods you can use to remove content associations from a web content page:

remove
: Removes an individual content association, as specified either by the resource ID of the folder or the content path of the folder.

delete Removes all content associations for the web content page.

The following examples demonstrate how to remove the content associations for two web content pages. The content associations of the first page are removed individually with the remove method, and the content associations of the second page are removed with the delete method.

Jacl example:

```
$set the_first_page [$Content find page uniquename "my.test.page"]
$ContentMapping select $the_first_page
foreach mid [$ContentMapping list mappings] {
  $ContentMapping remove $mid
}

$set another_page [$Content find page uniquename "my.second.test.page"]
$ContentMapping select $another_page
$ContentMapping delete
```

Jython example:

```
var the_page = Content.find('page','uniquename','my.test.page')
ContentMapping.select(the_first_page)
for mid in ContentMapping.list('mappings').split():
  ContentMapping.remove(mid)

var another_page = Content.find('page','uniquename','my.second.test.page')
ContentMapping.select(another_page)
ContentMapping.delete()
```

Modify content associations

To modify content associations, use the set method of the ContentMapping bean. You can change the following attributes:

- Default flag
- Delegation mode
- Mapping scope

When calling the set method, pass in the ID of the content association that you want to update.

The following example updates two content associations for the web content page identified by the unique name `my.test.page`. Several settings are specified for the first content association:

- The default flag is set to make this content association the default content association for the web content page.
- The association scope is specified as `_scp_`.
- Page-based access control is turned off by setting the delegation mode to `false`.

For the second content association, the association scope is removed by specifying an empty string.

Jacl example:

```
$ContentMapping select [$ContentNode find page uniquename "my.sample.page"]
set first_m_id [lindex [$ContentMapping list mappings] 0]
$ContentMapping set scope $first_m_id "_scp_"
$ContentMapping set default $first_m_id true
$ContentMapping set delegation $first_m_id false
set second_m_id [lindex [$ContentMapping list mappings] 1]
$ContentMapping set scope $second_m_id ""
```

Jython example:

```
ContentMapping.select(Content.find('all','un','my.sample.page'))
var first_m_id = ContentMapping.list('mappings').split()[0]
ContentMapping.set('scope',first_m_id,'_scp_')
ContentMapping.set('default',first_m_id,'true')
ContentMapping.set('delegation',first_m_id,'false')
var second_m_id = ContentMapping.list('mappings').split()[1]
ContentMapping.set('scope',second_m_id,'')
```

Related reference:

"Portal Scripting Interface" on page 1113
You can use the Portal Scripting Interface to configure your portal by running scripts from a command line.

REST API and content associations:

If you are creating or extending an application and want to manage content associations with that application, you can use portal remote APIs. These APIs retrieve a list of content associations and then create, update, or delete associations.

Based on the Representational State Transfer (REST) architecture, the APIs represent information about content associations as Atom feeds. To perform actions, you send HTTP requests to specific URLs.

Retrieve all content associations

This request returns a feed containing all content associations available in the system.

URL `http://hostname:port/context_root/mypoc?uri=contentmapping:objecttype:CONTENT_NODE&mode=download`

HTTP method
GET

Links Link information is provided for each entry in the Atom feed, as identified by the rel attribute.

Table 329. Link information in the Atom feed for retrieving content associations

Link	Description
rel="self"	Link to this Atom entry.
rel="edit"	Link to this item that can be used for POST, PUT, and DELETE operations.
rel="related"	Link that can be used to view the web content page with which the content association is associated.
rel="first"	Link to the first feed fragment. This link is only served if a feed fragment was served.
rel="last"	Link to the last feed fragment. This link is only served if a feed fragment was served.
rel="previous"	Link to the feed fragment preceding the current feed fragment. This link is only served if a feed fragment was served that does not start at the beginning of the feed.
rel="next"	Link to the next feed fragment. This link is only served if a feed fragment was served that does not contain the last entry of the feed.

Supported URL parameters

Table 330. Supported parameters for requests to retrieve content associations

Parameter	Description
start-index	Identifies the start index of the feed fragment to be served. The default value is `0`.
max-results	Identifies the maximum number of entries to be served by this request, as determined by the configuration of the server.

Retrieve content associations for a specific web content page

This request returns a feed containing the content associations defined for a specific web content page.

URL The web content page is identified either by its object ID or by its unique name.

`http://hostname:port/context_root/mypoc?uri=contentmapping:oid:serialized_object_id`

`http://hostname:port/context_root/mypoc?uri=contentmapping:oid:unique_name`

HTTP method
GET

Links Link information is provided for each entry in the Atom feed, as identified by the `rel` attribute.

Table 331. Link information in the Atom feed for retrieving content associations for a specific web content page

Link	Description
rel="self"	Link to this Atom entry.
rel="edit"	Link to this item that can be used for POST, PUT, and DELETE operations.
rel="related"	Link that can be used to view the web content page with which the content association is associated.

Modify content associations

This request updates the content associations defined for a specific web content page.

URL The web content page is identified either by its object ID or by its unique name.

`http://hostname:port/context_root/mypoc?uri=contentmapping:oid:serialized_object_id`

`http://hostname:port/context_root/mypoc?uri=contentmapping:oid:unique_name`

HTTP method
PUT

Links Link information is provided for each entry in the Atom feed, as identified by the `rel` attribute.

Table 332. Link information in the Atom feed for modifying content associations

Link	Description
rel="self"	Link to this Atom entry.
rel="edit"	Link to this item that can be used for POST, PUT, and DELETE operations.
rel="related"	Link that can be used to view the web content page with which the content association is associated.

Supported URL parameters

Table 333. Supported parameters for requests for modifying content associations

Parameters	Description
update	The following values are supported for the **update** parameter: **merge** This mode merges the content associations in the request with the content associations on the server. The request updates existing content associations and adds new content associations, but the request does not delete other content associations already on the server. **replace** This mode replaces all current content associations on the server with the content associations specified in the request. The request updates existing content associations, adds new content associations, and deletes other content associations on the server that are not represented in the request. **delete** This mode deletes the content association specified in the request from the web content page.

Delete content associations

This request deletes either all content associations for a specific web content page or an individual content association to specific content item.

URL The web content page is identified either by its object ID or by its unique name.

`http://hostname:port/context_root/mypoc?uri=contentmapping:oid:serialized_object_id`

`http://hostname:port/context_root/mypoc?uri=contentmapping:oid:unique_name`

HTTP method
DELETE

Supported URL parameters

Table 334. Supported parameters for requests for deleting content associations for a web content page

Parameters	Description
content	Indicates the content associations to be deleted. If the **content** parameter is not specified, all content associations for the web content page are deleted. If the content ID for a content item is specified as the value of the **content** parameter, only one content association is deleted. The content association deleted is the one that maps the web content page to the specified content item. Any other content associations are unaffected.

Example Atom feed document

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:feed xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/" xmlns:app="http://www.w3.org/
2007/app" xmlns:xhtml="http://www.w3.org/1999/xhtml" xmlns:contentmappings="http://
www.ibm.com/xmlns/prod/content-mappings/v1.0"
xmlns:atom="http://www.w3.org/2005/Atom"
xml:base="http://www.example.com:10039/wps/mypoc/!ut/p/contentmapping/objecttype%3aCONTENT_NODE">
 <atom:title>Content Mappings Feed</atom:title>
 <atom:author>
  <atom:name>WebSphere Portal</atom:name>
 </atom:author>
 <atom:link href="/wps/mycontenthandler/!ut/p/contentmapping/objecttype%3aCONTENT_NODE?
max-results=2" rel="self" type="application/atom+xml"/>
 <atom:id>contentmapping:objecttype%3aCONTENT_NODE</atom:id>
 <atom:updated>2009-08-25T12:10:51.641Z</atom:updated>
 <atom:entry>
  <atom:id>contentmapping:oid6_2QC68B1A0GVJE0IOA0R0Q02040</atom:id>
  <atom:title>6_2QC68B1A0GVJE0IOA0R0Q02040</atom:title>
  <atom:link href="/wps/mycontenthandler/!ut/p/contentmapping/oid6_2QC68B1A0GVJE0IOA0R0Q02040"
rel="self" type="application/atom+xml"/>
  <atom:link href="/wps/mycontenthandler/!ut/p/contentmapping/oid6_2QC68B1A0GVJE0IOA0R0Q02040"
rel="edit" type="application/atom+xml"/>
  <atom:link href="/wps/myportal/!ut/p/c4/04_SB8K8xLLM9MSSzPy8xBz9CP0os3ijQGczCydDRwP3
MC9XA09_R4Mgg0ADIwMTA_2CbEdFAHZmm-M!/" type="text/html"/>
  <atom:updated>2009-08-25T12:10:51.805Z</atom:updated>
  <atom:content type="application/xml">
   <contentmappings:content-mapping-info id="6_2QC68B1A0GVJE0IOA0R0Q02040">
    <contentmappings:content-mapping content-id="f40429bb-0cb5-4c0f-8080-cfb66f0e9b91"
default="true"/>
   </contentmappings:content-mapping-info>
  </atom:content>
 </atom:entry>
 <atom:entry>
  <atom:id>contentmapping:oid6_2QC68B1A00VBC0IOSHU0A220O6</atom:id>
  <atom:title>cnCTFPortlet</atom:title>
  <atom:link href="/wps/mycontenthandler/!ut/p/contentmapping/oid6_2QC68B1A00VBC0IOSHU0A220O6"
rel="self" type="application/atom+xml"/>
  <atom:link href="/wps/mycontenthandler/!ut/p/contentmapping/oid6_2QC68B1A00VBC0IOSHU0A220O6"
rel="edit" type="application/atom+xml"/>
  <atom:link href="/wps/myportal/!ut/p/c4/04_SB8K8xLLM9MSSzPy8xBz9CP0os3ijQGczCydDRwOD
MCdnA0__YI9QA0cjIwN_M_2CbEdFAB2I97c!/" type="text/html"/>
  <atom:updated>2009-08-25T12:10:51.810Z</atom:updated>
  <atom:content type="application/xml">
   <contentmappings:content-mapping-info id="6_2QC68B1A00VBC0IOSHU0A220O6">
    <contentmappings:content-mapping content-id="6f0b7a804d426a9a8d53ede9170f1e3d"
```

```
default="true"/>
    <contentmappings:content-mapping content-id="291410004ce29075b879f90c4ec954a0"
default="false"/>
   </contentmappings:content-mapping-info>
  </atom:content>
 </atom:entry>
 <atom:link href="/wps/mypoc/!ut/p/contentmapping/objecttype%
3aCONTENT_NODE?mode=download&start-index=2&max-results=2" rel="next"
type="application/atom+xml"/>
</atom:feed>
```

Related tasks:

"Creating a web content page with the XML configuration interface" on page 2074
As with other portal pages, you can create a web content page with the XML configuration interface (**xmlaccess** command). Page definition is similar to a standard portal page. However, there is an additional page parameter that specifies the site area that is associated with the web content page.

Dynamic web content page selection:

The web content viewer provides a link broadcasting feature that dynamically determines the best web content page to use when rendering the linked content item. To perform this page selection, the viewer uses the content associations on the web content page.

Understanding dynamic page selection is useful when examining how multiple associations affect link resolution and when troubleshooting why a link is not targeting the expected page.

Dynamic page selection is enabled when the web content viewer is configured to use the **Dynamically select a web content page** link broadcasting option. To render a content item, the dynamically selected target page must contain at least one web content viewer. The viewer must be configured to receive links from other portlets.

When users select a link to a content item, the chain of content page resolution filters runs to determine which page renders the selected item. The same mechanism is used when previewing web content on pages or when selecting search results produced from the search seedlist 1.0 feature. You can control how the page is determined by adding a custom content page resolution filter into this filter chain. For more information, see *Creating a content page resolution filter class.*

The default filter plug-ins perform the following tests to determine a web content page.

1. An ordered list with all parent site areas of the selected content item is created. If the selected item itself is a site area, it is part of that list. The order in the list matches the order in the Web Content Manager content hierarchy. The topmost parent is the last item in the list, and the direct parent of the selected item (or the item itself) is the first item.
2. Based on this ordered list, a lookup is performed to find the web content page with the following criteria:
   - The current user has view access to the page.
   - The page is mapped to the first site area for which a page content association exists.

   If the content associations indicate that multiple pages map equally well, the web content viewer selects a single page using the following rules:

- If the current page is among the pages found, the current page takes precedence over the other results.
- If one of the pages that is indicated by the content associations is the default association for the page, that page takes precedence over other results.
- If the previous rules do not identify a page, a page is selected arbitrarily from the set of found pages.

3. The content item that gets rendered depends on the situation.
   - If a web content page can be determined in the previous test, the selected content item is rendered on this page.
   - If no page can be identified by the content associations, the web content fallback page is used to render the content item.
   - If no fallback page is configured or if the user does not have view rights on the fallback page, the Web Content Manager servlet renders the item.

Related tasks:

"Setting up a web content fallback page" on page 2057
Set up a web content fallback page to be used when a web content viewer cannot determine which page to use to display a content item. The fallback page can also be used when users do not have sufficient privileges to view the page originally associated with the content item.

"Creating a content page resolution filter class" on page 3231
A content page resolution filter is used to customize the behavior of the content page resolution filter chain. This method is used to tailor the response to a web content request in several ways, including overriding the content item that is displayed or the portal page that is used to display a content item in the web content viewer.

Web Content Viewer best practices and limitations

View some best practices and limitations for using Web Content Viewers.

User authentication

User authentication to Web Content Viewers is managed by IBM WebSphere Portal Express and IBM WebSphere Application Server.

Security and WSRP

When you display web content from remote servers by using WSRP and a Web Content Viewer, configure security and authentication between the servers:

- The portal server that acts as the WSRP Consumer
- The web content server that acts as the WSRP Producer

See *Security for WSRP services* for details.

User access to Web Content Manager content and components

Users are able to view only content and components that can be accessed by either a portal user ID or the user of the WSRP Consumer. This access must be defined in Web Content Manager. If a user ID or the user of the WSRP Consumer does not have sufficient rights to view a content or component, errors can occur.

Content and component limitations

Not all content or components that are built in a Web Content Manager solution are suitable for inclusion in a portal page:

- Content or components to be shown within a portal page must be self-contained and not rely on other content or components.
- When you create presentation templates or page styles to use when you display content within a portlet, reference only the content you want to show. Add components like menus and navigators in separate portlets, and link the components to other content portlets as required.
- JavaScript URLs are not supported.

Using JavaScript:

When a web page is rendered by Web Content Manager, some tags might be rewritten. Web Content Manager uses double quotation marks for attributes in HTML tags. If you use JavaScript to produce HTML tags, Web Content Manager does not recognize them if you use single quotation marks.

Other limitations

- The results of a POST operation in a form are only displayed within the portlet that sent the POST. You cannot send the result of a POST to another portlet.
- An anonymous portal user is also considered an anonymous Web Content Manager user, so overriding the log-in does not work for anonymous users.
- If a Web Content Manager proxy server is being used with Web Content Viewers, URLs rewritten by the proxy are not fully qualified. Instead, the URLs are relative to the server. To address this issue, redirect mappings can be created in the HTTP Server configuration that passes the URLs to the proxy server.
- Category selection trees cannot be used with the local Web Content Viewer.
- Only advanced caching can be used with a local Web Content Viewer.
- Tagged web content that is displayed in the Web Content Viewer is only available when there is a single instance of the portlet on the page. When you click a tag result, the Tag Center broadcasts the information about what content to display by using a public render parameter. If you are displaying multiple instances of content in the viewer, the instances show the content that you tagged rather than their original content.

Related tasks:

"Adding existing blogs or blog libraries to a page" on page 2002
If you created a blog or blog library for another page and now want to use it again, add a web content viewer to the new page and edit its settings to point to the existing blog or blog library.

"Adding a blog or blog library to a page" on page 2001
With Editor access to the portal or the portal page, you can add a blog or blog library to a page. Choose a blog to collaborate with your team on a single topic. Choose a blog library to collaborate with your team on multiple topics in a centralized view.

Access web content by using a servlet

Users can access content that is displayed by using the Web Content Manager servlet by connecting to a URL. A servlet delivered website is used when you don't need to use any WebSphere Portal based features such as authoring tools.

Accessing a web page by using a servlet

The following URL structure is used to connect to a web page:

```
http://[HOST]:[PORT]/wps/wcm/connect/[PATH]?srv=
```

Non-ascii characters:

Non-ascii characters cannot be used in the query string section of URLs. For this reason, it is best not to name Web Content Manager items using Non-ascii characters if you plan to use URLs to call Web Content Manager items.

- *[PATH]* can be the path to a site area or content item. This setting must be entered for all types of content, including components. In the case of components, this is the path to the site area or content item that the component is displayed with.
- *srv=* is either *cmpnt* or *page*.

Table 335. Service options

Service option	Details
`srv=cmpnt`	This option retrieves a component either from the component library or from a site area or content item. You must also specify the following: **`source=`** This determines where the component is being sourced from. This is either: • library • sitearea • content **`cmpntname=componentname`** This is the name of the component being retrieved.
`srv=page`	This retrieves a content item. As *srv=page* is returned as default, this can be omitted from the URL. The presentation template to use when displaying this content is specified by adding: `presentationtemplate=library/presentationtemplatename`

Examples:

URL to content:

```
http://[HOST]:[PORT]/wps/wcm/connect/[PATH]
```

Example: `http://host:10039/wps/wcm/connect/sitearea/content`

URL to content with a presentation template defined:

```
http://[HOST]:[PORT]/wps/wcm/connect/[PATH]
?presentationtemplate=[libraryname/presentationtemplatename]
```

Example: `http://host:10039/wps/wcm/connect/sitearea/content?presentationtemplate=library/presentationtemplate`

URL to a library component:

```
http://[HOST]:[PORT]/wps/wcm/connect/[PATH]
?srv=cmpnt&source=library&cmpntname=[componentname]
```

Example: `http://host:10039/wps/wcm/connect/sitearea/content?srv=cmpnt&source=library&cmpntname=component`

URL to a content component:

```
http://[HOST]:[PORT]/wps/wcm/connect/[PATH]
?srv=cmpnt&source=content&cmpntname=[componentname]
```

Example: `http://host:10039/wps/wcm/connect/sitearea/content?srv=cmpnt&source=content&cmpntname=component`

Applying Custom Caching and Expiring Parameters.

Like any other URL request that is made to a Web Content Manager Server, Custom Caching and Expiring parameters can be added to a request. See the topic, "Using Custom Caching" for further information.

Example:

`http://[HOST]:[PORT]/wps/wcm/connect/[PATH]?CACHE=SITE&EXPIRES=REL+9000s`

In this example, the content being retrieved by this URL is saved in the Basic Site Cache, and expired after 9000 seconds (two and half hours).

Pre-rendered delivery

You can pre-render a complete IBM Web Content Manager site into HTML and save it to disk. The pre-rendered site can then be used as your live site and displayed to users that use either Web Content Manager or a web server. You deploy a pre-rendered site when you are not using any WebSphere Portal features and your content is static and is only updated periodically.

Restrictions

- Site areas and content item names cannot contain characters that are considered invalid in file names by the operating system on which you are pre-rendering. For example, on a Windows server, these characters are invalid: `/ \ : * ? " < > |`.
- The path to the content item, including the directory path to which you are pre-rendering (for example, site area/content) cannot exceed the operating system's maximum path length:
  - Windows: 255 characters
  - IBM iLinux: 1024 characters
- The Search component cannot be used in pre-rendered sites.
- The Page navigation component cannot be used in pre-rendered sites.
- Personalization elements can be pre-rendered only if the personalization rule is configured for anonymous access.

Site security

Item security for different users set in an Web Content Manager environment is not transferred to pre-rendered sites. The security for the entire pre-rendered site is based on the `connect.moduleconfig.cacher.rendereruser` property as specified in the **WCM WCMConfigService** service by using the WebSphere Integrated Solutions Console.

"Pre-render methods" on page 2090
Pre-rendering can be configured to run automatically, or you can manually pre-render a website by using a URL.

"How to access the pre-rendered site" on page 2091
Pre-rendered sites are accessed either through IBM Web Content Manager, or through a web server.

Related concepts:

"Pre-rendering options" on page 425
You can enable pre-rendering so that content can be viewed either through a IBM Web Content Manager application or as a stand-alone site that is accessed through

a web server.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Pre-render methods

Pre-rendering can be configured to run automatically, or you can manually pre-render a website by using a URL.

Administrator access: To pre-render a website, you must have administrator access to the library that contains the site area.

Automatically pre-rendering a website

Pre-rendering can be run according to the cacher settings specified for the **WCM WCMConfigService** service as part of the pre-rendering configuration.

Manually pre-rendering a website

Pre-rendering can also be initiated through the URL interface. For example:

`http://host_name:port_number/wps/wcm/connect?MOD=Cacher&SRV=cacheSite&sitearea=sitearea_name&library=library_name`

Table 336. CacherModule options

Service	Required Parameters	Optional Parameters
SRV=cacheSite Initializes pre-rendering for the site area with a delay as given (in seconds).	`sitearea`=site area name	DELAY=<delay> LIBRARY=<library> **Note:** If no library is specified, the default library is used, as specified by the `defaultLibrary` property in the `WCM WCMConfigService` service.
SRV=flushSiteCache Clears (flushes) the site cache. Deletes all pre-rendered data.	`sitearea`=site area name	LIBRARY=<library> **Note:** If no library is specified, the default library is used, as specified by the `defaultLibrary` property in the `WCM WCMConfigService` service.
SRV=flushPageCache Flushes the page from the site cache. The site area and page are determined from the request URL.		
No SRV specified The CacherModule attempts to retrieve the page from the cache.		

Pre-rendering individual content items

You can also pre-render individual content items by using the following URL:

`http://host_name:port_number/wps/wcm/connect/library_name/site_area_name/content?MOD=Cacher`

Note:

- To pre-render individual content items, the site area that is specified in the URL must either be a site area set in **connect.moduleconfig.cacher.task.siteareas** in the **WCM WCMConfigService** service, or you must have previously manually pre-rendered the site area using "SRV=cacheSite" so that the site area already exists in the location you pre-render to.

Related concepts:

"Pre-rendering options" on page 425
You can enable pre-rendering so that content can be viewed either through a IBM Web Content Manager application or as a stand-alone site that is accessed through a web server.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

How to access the pre-rendered site

Pre-rendered sites are accessed either through IBM Web Content Manager, or through a web server.

Accessing the pre-rendered site through a Web Content Manager application

To enable users to access the pre-rendered site through a Web Content Manager application, specify the `connect.businesslogic.module.default.class` property in the `WCM WCMConfigService` service by using the WebSphere Integrated Solutions Console.

- Property name: `connect.businesslogic.module.default.class`
- Value: `com.aptrix.cacher.CacherModule`

Users can access the site through the following URL:

`http://host_name:port_number/wps/wcm/connect/library_name/sitearea_name`

The *library_name* parameter is optional. If no library is specified, the default library is used, as specified by the `defaultLibrary` property in the **WCM WCMConfigService** service.

Connect tags:

Connect tags are not processed by the CacherModule and are rendered intact. When the pre-rendered page is accessed by a user, only then are the connect tags processed.

Pre-rendering JSP components:

Extended cache support needs to be enabled to support pre-rendering JSP components. This setting is disabled by default. See the Knowledge Centre

topic that is called **Web Content Manager pre-rendering service** for details on how to enable the `prerender.extended.support.enabled` property.

Links to content not yet pre-rendered:
A component, such as a menu, that contains links to content not yet pre-rendered is retrieved by the CacherModule and added to the pre-rendered site. This only applies to content belonging to sites configured to be pre-rendered.

Custom expiring:
Custom caching parameters cannot be used in connect tags and URL requests in pre-rendered sites. "`EXPIRES=`" and "`CONNECTORCACHEEXPIRY=`" can be used to override your server's default basic and data cache settings.

Authoring portlet:
The authoring portlet can still be accessed when the default class is changed from `RendererModule` to `CacherModule` as long as it has not been added to the lists of sites to be pre-rendered.

CacherModule as default:
If using the URL interface when the `CacherModule` is the default, you do not need to specify `?mod=cacher`. Instead, enter the request as follows:

`http://`*`host_name`*`:`*`port_number`*`/wps/wcm/connect?SRV=cacheSite&library=`*`library_name`*`&sitearea=`*`sitearea_name`*

Accessing the pre-rendered site through a web server

If your web server is not used for WebSphere Portal Server, you can configure your web server to map to the following alias and Web Content Manager directories:

Table 337. Alias and directory details

Alias	Directory
/wps/wcm/connect	[ILWWCM_HOME]/ilwwcm/cacher

When using a web server to view a pre-rendered site, the previous directories require execute access.

Users access the site using the following URL:
`http://`*`host_name`*`:`*`port_number`*`/wps/wcm/connect/`*`library_name`*`/`*`sitearea_name`*

Use the following configuration properties in the `WCM WCMConfigService` to change the context of the URLs generated with pre-rendering:

- `connect.moduleconfig.cacher.task.cacherurl`
- `connect.moduleconfig.cacher.task.servletpath`

For example, to set a context of `/sales`, use the following properties:

Cacher URL

- Property name: `connect.moduleconfig.cacher.task.cacherurl`
- Value: `http://${WCM_HOST}:${WCM_PORT}/sales`

Servlet path

- Property name: `connect.moduleconfig.cacher.task.servletpath`
- Value: `/connect`

If your web server is used for both a WebSphere Portal server and accessing the pre-rendered site, you must change the context of the URLs. Any context that starts with the WebSphere Portal server context, `/wps` for example, is redirected to WebSphere Portal server.

Connect tags cannot be used:
Connect tags are not processed by the CacherModule and are rendered intact. When the page is viewed through a web server, connect tags cannot be processed. Therefore, connect tags cannot be used in sites that are to be viewed through a web server.

Pre-rendering JSP components:
The pre-rendering feature cannot be used to pre-render JSP components.

Dynamic elements:
When viewing a pre-rendered site through a web server, any dynamic elements such as menus and navigators are displayed as rendered by the configured CacherModule user, not by the user that is accessing the site. This means personalization cannot be used.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Rendering modes for web content

CF07

Different presentation templates are created to render web content in different modes to display content in different contexts, such as a web content viewer portlet, or mobile devices.

Render modes

These modes are supported when web content is rendered. You can create separate presentation templates for each type and select them when you create a site area template or content template:

- **Default Presentation Template**: This presentation template is used to render an item within a Web Content Viewer portlet.
- CF07 **Summary Presentation Template**: This presentation template is used when the summary render mode is used to render an item.
- CF07 **JSON Record Presentation Template**: This presentation template is used when the JSON render mode is used to render an item on mobile devices.
- CF07 **XML Document Presentation Template**: This presentation template is used when the XML render mode is used to render an item on mobile devices.
- CF07 **HTML Document Presentation Template**: This presentation template is used when the HTML render mode is used to render an item as a complete web page.

You then select a different presentation template for each rendering mode by clicking **Select Presentation Template** in the **Item Properties** section when you create a site area template or content template.

Defining rendering modes in a URL

You can specify a rendering mode in a URL by adding this parameter to the URL: `renderMode=`*`mode`*

These render modes are predefined: **`summary, json, xml, html`**. You can use the Web Content Manager API to define further render modes.

For example, to specify the URL to use the JSON rendering mode, you might use a URL like this:

`http://`*`myserver:port`*`/wps/wcm/myconnect/`*`mylibrary/mysitearea/mycontent`***`?renderMode=json`**

To specify more than one render mode, use a comma-separated list. For example, to specify the URL to use the JSON rendering mode, or if no JSON presentation template is specified, the XML rendering mode, you might use a URL like this:

`http://`*`myserver:port`*`/wps/wcm/myconnect/`*`mylibrary/mysitearea/mycontent`***`?renderMode=json,xml`**

Vanity URLs

You can associate vanity URLs with portal pages and labels. Vanity URLs are short URLs that people can easily remember. They are shorter than full WebSphere Portal Express URLs. They are sometimes also called marketing URLs. You can publish vanity URLs for marketing campaigns through different channels, such as email or print. This way, you can use vanity URLs to direct customers to a specific portal page or content item. Interested site visitors who want to view your campaign can then remember or copy the short vanity URL and type it into the browser address field.

Business users can create short and arbitrary URLs as vanity URLs. The vanity URLs can point to pages that marketing people can post in ads or magazines. To create vanity URLs, you use the IBM WebSphere Portal Express toolbar. Businesses can provide URLs with the following benefits to their customers and portal site visitors:

- Business users can assign vanity URLs to pages or to content on pages easily and change them frequently, for example, for promotion purposes.
- For site visitors, vanity URLs are self-speaking, short, and easy to remember. Site visitors can easily remember and type vanity URLs.

Technical and usage details about vanity URLs:

- Business and marketing users can easily define and manage vanity URLs by using the portal toolbar. Managing vanity URLs does not require administrator access and use of the portal administration.
- Vanity URLs include the host name and the vanity name. They are part of the metadata of the portal page to which they resolve. However, they are not bound to the hierarchy of the friendly name path.
- WebSphere Portal Express provides a new vanity URL servlet. That servlet is called if a URL contains the portion `host/wps/vanityurl`, followed by a vanity URL segment. For example, in the vanity URL `host/wps/vanityurl/hotnews`, `hotnews` is the vanity URL segment. The servlet resolves the vanity URL segment to the appropriate portal page.
- When a site visitor goes to a vanity URL, this URL does not remain in the browser **URL address** field. Instead, the portal resolves the vanity URL to the full portal page URL and redirects the site visitor to the appropriate portal page.

- Vanity URLs are part of the portal page and are syndicated with the page. They are stored in the portal page site area item. Therefore, to be able to use vanity URLs, you must enable Managed pages.
- You can give a portal page multiple vanity URLs.
- You can use a vanity URL to address one or more specific content items on the page. When a user accesses the vanity URL, the portal shows the page with the content item that you specified.
- You can choose a specific locale for the target page of the vanity URL. When a user accesses the page, the portal shows the page in the locale that you specified. You can also attach multiple locales to a vanity URL. This way, the user can view the page in the preferred language.
- In their ease of use, vanity URLs are similar to friendly URLs. However, unlike friendly URLs, they do not include the portal context. They are also not constructed as a portal navigation path; therefore portal site visitors cannot go to a vanity URL by using the portal navigation.
- You can make your vanity URLs even shorter by using an HTTP server rewrite rule. You can then omit the segment `wps/vanityurl` and reduce the vanity URL to the host name and the vanity URL segment as follows: `host/hotnews`. For more information, read *Providing short vanity URLs*.

Example of a vanity URL for a portal page:

- A portal page can have the following friendly name path: `host/wps/portal/home/products/newandexiciting`.
- You can define a vanity URL segment that is named `coolstuff`. You can then post the following vanity URL to your customers: `host/wps/vanityurl/coolstuff`.
- You can reduce the vanity URL even further by using an HTTP server rewrite rule and then omitting the segment `wps/vanityurl`. You can now give your site visitors the shorter vanity URL: `host/coolstuff`.
- When a user enters the vanity URL in the browser address field, the vanity URL servlet resolves the vanity URL segment to the URL of the appropriate portal page.

Vanity URLs and URL mappings:

URL mappings were deprecated starting with WebSphere Portal Express Version 8.5. Instead, you can now use friendly URLs or Vanity URLs as an alternative to URL mapping. Vanity URLs are similar to URL mappings in that you can define them independent from the page hierarchy. However, the following differences apply between vanity URLs and URL mappings:

- Vanity URLs are managed by business and marketing users by using the toolbar, not by administrators by using the portal administration. This way, vanity URLs are easier to manage than URL mappings.
- Vanity URLs are not bound to the portal page hierarchy. Now, you can have a single path segment for the vanity URL.
- Vanity URLs are managed in Web Content Manager as part of a page. A vanity URL can therefore be part of a project. It can be affected by versioning, workflow and syndication.
- Vanity URLs do not have to contain the URL path segment `/wps/portal`. Additionally, if you use an HTTP server rewrite rule, you can omit the `/wps/vanityurl` segment as well.
- Vanity URLs are attached to pages and do not exist independently from pages.

- Vanity URLs are independent from the portal site and its content taxonomy and hierarchy. In this regard, vanity URLs are similar to URL Mappings, but they do not include the portal context.

If you prefer to use URL mappings, you can continue to use existing URL mappings. You can also reinstall the old URL mapping portlet. You find the installable WAR file for the portlet under `PortalServer/ap/wp.ap.urlmapping/installableApps/urlmapping.war`.

Deciding between vanity URLs and friendly URLs

Depending on your requirements, you can use vanity URLs, or friendly URLs:

- If you want to have a short URL as an entry point to a specific portal page or content item, use a vanity URL.
- If you want to have a friendly URL that your site visitors see when the portal shows the page, use a friendly name.
- If you want to be able to publish the page through the Web Content Manager workflow, use a vanity URL. For example, this URL can be useful for a marketing campaign.
- If you want to address a specific portal page through URL generation tags or APIs, use unique name IDs. For more information, see *URL generation in WebSphere Portal*.

You can create both vanity URLs and friendly URLs for the same portal page.

"Viewing and creating vanity URLs"
To view and create vanity URLs, you work with the WebSphere Portal Express toolbar.

"How vanity URLs work" on page 2097
Learn about vanity URLs, how they work, and how you work with them.

"Administering vanity URLs" on page 2099
IBM WebSphere Portal Express provides some configuration tasks. You can use these tasks to administer vanity URL support. You can also configure your IBM HTTP Server so that you can use short vanity URLs.

Related concepts:

"URL generation in WebSphere Portal" on page 2835
Generating Portal URLs correctly is one of the most important tasks in programming a WebSphere Portal Express based application. There are several programming tools and techniques available for generating WebSphere Portal Express URLs in custom code. The following section introduces the programming tools available and discusses when it is most appropriate to use each of the tools.

Related tasks:

"Providing short vanity URLs" on page 2101
You might want to make your vanity URLs as short and simple as possible for your customers. You can create vanity URLs that contain only the vanity segment by omitting the string `/wps/vanityurl`. In this case, you must use a web server and define a rewrite rule. If you also use IBM Web Application Bridge, or if you have static files in the root of the HTTP server document directory, adapt the rewrite rule.

Viewing and creating vanity URLs

To view and create vanity URLs, you work with the WebSphere Portal Express toolbar.

Before you begin

To create, update, and delete a vanity URL for a specific page, the user must have editor rights on the page and on the Virtual Resource Vanity_URL.

Procedure

1. From the portal action bar, select **Open the toolbar**.
2. Select **Page**.
3. To view existing vanity URLs for the page, select **General**.
4. To create a new vanity URL for the page, proceed as follows:
   a. Select **Vanity URL**.
   b. Click **New vanity URL**.
   c. To complete the creation of the vanity URL, follow the guidance that is given by the user interface.

Related concepts:

"Access permissions" on page 1537
Learn about sensitive operations for resources and the roles that are required to perform those operations. Sensitive operations include common tasks such as viewing portlets on specific pages and complex, high-risk tasks like running XML configuration interface scripts.

Related tasks:

"Setting user and group permissions" on page 1567
The User and Group Permissions portlet lets you view and modify the roles that users and groups have on resources.

How vanity URLs work

Learn about vanity URLs, how they work, and how you work with them.

In a new WebSphere Portal Express Version 8.5 installation, vanity URL support is enabled. If you upgrade your WebSphere Portal Express from a previous version to Version 8.5, vanity URL support is disabled. You can enable and disable vanity URL support as required by using a portal configuration task. For more information, see *Enabling vanity URL support*. For vanity URLs to work, you must also enable managed pages. If vanity URL support is not enabled and a user tries to access a vanity URL, the portal gives a 404 return code.

Vanity URLs are stored as part of the page data on the Web Content Manager portal page site area. When you create a vanity URL, both the vanity URL and the information for resolving the vanity URL is stored in the Vanity URL property of the portal page site area item. This storage method has the following consequences:

- You manage vanity URLs by using the portal toolbar and the Managed Pages features. Therefore, if you want to use vanity URLs, Managed Pages must be enabled. The Managed Pages feature is enabled by default in a WebSphere Portal Express Version 8.5 installation. If you upgrade your portal from a previous version to Version 8.5 and want to use vanity URLs, you must enable Managed Pages.
- Assigning a new vanity URL to a page, or changing or deleting an existing vanity URL in the scope of a project creates a draft of the page.
- To export vanity URL information, you do not use the XML configuration interface (XMLAccess), but the JCR export feature of the portal page site area.

WebSphere Portal Express provides a vanity URL servlet. It resolves an incoming vanity URL request to the appropriate portal page or content item.

You can assign multiple vanity URLs to a page. A vanity URL can have different locales that are attached to it.

Vanity URLs are not included in the portal search seedlist. To provide good ranking for your vanity URL pages with search engines, assign friendly names to those pages. The friendly names are listed in the portal search seedlist. To achieve better search rankings, the portal search seedlist does not list the additional Vanity URLs.

If you have virtual portals, the following rules apply:

- You must assign a vanity URL that is unique across all of your virtual portals.
- The host name of the current virtual portal is used as the host name of the vanity URL. If the virtual portal was defined by using a context path, the host name of the virtual portal context path is used.
- For a vanity URL in the default virtual portal, the host name of that default virtual portal is used as the host name of the vanity URL.

An example process flow of a business user who works with a vanity URL is as follows:

1. You have a shoe shop, and you want to advertise a shoe promotion sale.
2. You have a portal page that has the following friendly URL:
   `http://hostname:port/wps/portal/home/shoe_promotion_page`
3. When you create a vanity URL, you define the vanity URL segment. Example: `shoe-sale`. You can also select from the following options:
   - Portal content item: Use this option to specify a content item on the target page for the vanity URL. When a site visitor accesses the vanity URL, the portal shows the site visitor the page with the content item that you specified.
   - Locale: If you use this option to specify a locale for the target page for the vanity URL. When a site visitor accesses the vanity URL, the portal shows the site visitor the page in the language and locale that you specified.
4. To advertise your shoe sale, insert your vanity URL segment `shoe-sale` into the URL, and add a preceding segment `vanityurl`. The URL now looks as follows:
   `http://hostname/wps/vanityurl/shoe-sale`

   This URL is the vanity URL that you pass out in your advertisements for the shoe sale.
5. When a user accesses the page by using this URL, the portal redirects the user to your shoe sale page under the URL given earlier:
   `http://hostname:port/wps/portal/home/shoe_promotion_page`

   When the portal redirects the user, it also adds state information to the URL. For example, this state information includes information about the portal content and the locale as selected when you created the vanity URL in earlier steps.
6. You can make the vanity URLs shorter by omitting the portal root context `/wps` and the string `/vanityurl`. In this case, you must use a web server and define a rewrite rule. For more information, see *Providing short vanity URLs*. You can then pass out the following URL for your advertising campaign:
   `http://hostname:port/shoe-sale`

When a user accesses this URL, the HTTP server rewrites this URL to the longer URL, and then redirects the user as described before.

Related tasks:

"Enabling vanity URL support" on page 2100
In a new WebSphere Portal Express Version 8.5 installation, vanity URL support is enabled. If you upgrade your WebSphere Portal Express from a previous version to Version 8.5, vanity URL support is disabled. You can enable and disable vanity URL support as required by using a portal configuration task.

"Providing short vanity URLs" on page 2101
You might want to make your vanity URLs as short and simple as possible for your customers. You can create vanity URLs that contain only the vanity segment by omitting the string `/wps/vanityurl`. In this case, you must use a web server and define a rewrite rule. If you also use IBM Web Application Bridge, or if you have static files in the root of the HTTP server document directory, adapt the rewrite rule.

Administering vanity URLs

IBM WebSphere Portal Express provides some configuration tasks. You can use these tasks to administer vanity URL support. You can also configure your IBM HTTP Server so that you can use short vanity URLs.

About this task

Staging vanity URLs to production

A vanity URL is a portal artifact that is stored with the portal page that it targets. Vanity URLs are managed and stored in the Web Content Manager Portal Site library. To stage vanity URLs to production, you stage the IBM Web Content Manager page that the vanity URL targets.

"Enabling vanity URL support" on page 2100
In a new WebSphere Portal Express Version 8.5 installation, vanity URL support is enabled. If you upgrade your WebSphere Portal Express from a previous version to Version 8.5, vanity URL support is disabled. You can enable and disable vanity URL support as required by using a portal configuration task.

"Providing short vanity URLs" on page 2101
You might want to make your vanity URLs as short and simple as possible for your customers. You can create vanity URLs that contain only the vanity segment by omitting the string `/wps/vanityurl`. In this case, you must use a web server and define a rewrite rule. If you also use IBM Web Application Bridge, or if you have static files in the root of the HTTP server document directory, adapt the rewrite rule.

"Configuring the vanity URL preview link" on page 2103
The user interface for managing vanity URLs has a preview link. By default, this link shows the full vanity URL. If you installed an HTTP server and configured it to allow short vanity URLs, you can configure the preview link to show the short vanity URL instead.

"Synchronizing the vanity URL database" on page 2104
Vanity URLs are stored as part of the page in the JCR database in the portal page site area of Web Content Manager. For performance reasons, the data is also stored in the WebSphere Portal Express database. When the data is modified, the portal synchronizes the data between both sides. However, under certain circumstances it can happen that the data is not synchronized. For such cases, the portal provides a configuration task that synchronizes the data.

"Setting an error URI for undefined vanity URLs" on page 2105
You can configure how the portal behaves if a user tries to access an undefined vanity URL.

Related concepts:

"Syndication and staging" on page 2503
You can use syndication to update content that was originally created by deploying portal solution releases with either the XML configuration interface or through a Portal Application Archive (PAA) file. You can also set up syndication between virtual portals or primary portals on the same system or between virtual portals on different systems.

Enabling vanity URL support

In a new WebSphere Portal Express Version 8.5 installation, vanity URL support is enabled. If you upgrade your WebSphere Portal Express from a previous version to Version 8.5, vanity URL support is disabled. You can enable and disable vanity URL support as required by using a portal configuration task.

About this task

If vanity URL support is not enabled and a user tries to access a vanity URL, the portal gives a 404 return code.

Note: For vanity URLs to work, managed pages also must be enabled as a prerequisite. For more information, read *Enabling managed pages after migration*. If you disable managed pages, this step also disables vanity URLs. If you do not disable vanity URLs and you enable managed pages again, vanity URLs also work again.

Enabling vanity URL support

To enable vanity URL support, you use the configuration task **enable-vanityurl-support**. This task sets a new custom property in the Resource Environment Provider of the WP Configuration Service. The property name is **vanityurl.support.enabled**. When you run the configuration task, the property is set to the value `true`.

Note: It is mandatory to enable managed pages to have vanity URLs enabled. If managed pages support is not enabled, the task **enable-vanityurl-support** fails. In this case, run the task **enable-managed-pages** first.

Disabling vanity URL support

To disable vanity URL support, you use the configuration task **disable-vanityurl-support**. This task deletes the custom property **vanityurl.support.enabled** from the Resource Environment Provider of the WP Configuration Service. If the portal then receives a request to serve a vanity URL, it gives a 404 return code.

Syntax

You call these configuration tasks as follows:

IBM i

Enable: `ConfigEngine.sh enable-vanityurl-support -DPortalAdminPwd=`*`password`* `-DWasPassword=`*`password`*

Disable: `ConfigEngine.sh disable-vanityurl-support -DPortalAdminPwd=`*`password`* `-DWasPassword=`*`password`*

Linux

Enable: `./ConfigEngine.sh enable-vanityurl-support -DPortalAdminPwd=password -DWasPassword=password`

Disable: `./ConfigEngine.sh disable-vanityurl-support -DPortalAdminPwd=password -DWasPassword=password`

Windows

Enable: `ConfigEngine.bat enable-vanityurl-support -DPortalAdminPwd=password -DWasPassword=password`

Disable: `ConfigEngine.bat disable-vanityurl-support -DPortalAdminPwd=password -DWasPassword=password`

Remember: After you run the **enable-vanityurl-support** or **disable-vanityurl-support** task, restart the server. Go to "Starting and stopping servers, deployment managers, and node agents" on page 1216 for specific instructions.

Related tasks:

"Enabling managed pages" on page 376
By default, support for managed pages is enabled for the default virtual portal. However, you can also manually enable this support if managed pages are disabled.

"Enabling managed pages" on page 913
After migration, you must manually enable support for managed pages. Without managed pages support, some features like the project menu are not available on the migrated server.

Related reference:

"Troubleshooting pages" on page 1741
When you work with pages, you might encounter problems that are related to projects, access rights, or other issues.

Providing short vanity URLs

You might want to make your vanity URLs as short and simple as possible for your customers. You can create vanity URLs that contain only the vanity segment by omitting the string `/wps/vanityurl`. In this case, you must use a web server and define a rewrite rule. If you also use IBM Web Application Bridge, or if you have static files in the root of the HTTP server document directory, adapt the rewrite rule.

About this task

Example: You advertise your shoe sale by using the short vanity URL `http://hostname/shoe-sale`. The HTTP server rewrites this URL to `http://hostname/wps/vanityurl/shoe-sale`. The portal then redirects the user to `http://hostname/wps/portal/home/shoe_promotion_page`.

To be able to use such short vanity URLs, you must use a web server. For details about using a web server with your portal, see the topic about Preparing a remote web server for your environment in this portal product documentation.

Procedure

If you use the IBM HTTP Server as your web server, modify the file `httpd.conf` to define the rewrite rule. Proceed by the following steps:

1. Open the file `httpd.conf` with an editor.
2. Activate the following modules:

```
proxy_module
proxy_http_module
rewrite_module
```

3. Activate the rewrite engine by adding the following line:

   ```
   RewriteEngine   On
   ```

4. Add the rewrite rule by adding the following line:

   ```
   RewriteRule ^/([^/]+)$              /wps/vanityurl/$1        [P]
   ```

   This rule redirects all single path requests to the vanity URL servlet. The `P` flag at the end of the line tells the rewrite engine to use a proxy request. This flag is required for the IBM WebSphere Application Server plug-in to handle the request without an extra redirect.

5. Optional: If your website editors work in projects and you want them to be able to use short vanity URLs by the HTTP server, you need an extra rewrite rule as follows:

   ```
   RewriteRule     ^/\$project/([^/]+)/([^/]+)$             /wps/vanityurl/\$project/$1/$2        [
   ```

   This rule rewrites all URLs that start with `/$project/`*`project name`*`/`*`vanity name`*.

6. Optional: The rules that are given in the previous steps might prevent static files in the document directories of your HTTP server from being served. To have them served, add conditions before the rewrite rule as in the following example:

   ```
   RewriteCond     %{DOCUMENT_ROOT}%{REQUEST_FILENAME}     !-f
   RewriteCond     %{DOCUMENT_ROOT}%{REQUEST_FILENAME}     !-d
   RewriteCond     %{DOCUMENT_ROOT}%{REQUEST_FILENAME}     !-l
   ```

   In this example, `f` means file, `d` means directory, and `l` means symbolic link. With these conditions added, the portal does not apply the rewrite rule on requests that match a file, directory, or symbolic link. The user accesses the file, directory, or link and is not redirected to the target website of the vanity URL. For more information, see the information under *Apache Module mod_rewrite*.

   Note: Depending on your environment, it is good practice not to use periods in your vanity URLs to avoid conflicts with files in the HTTP server context root.

7. Optional: If you use Web Application Bridge, add a rewrite rule to avoid namespace conflicts. Web Application Bridge must be mapped to the root context. As a result, vanity URLs and Web Application Bridge are in the same namespace, which can result in conflicts. Therefore, if you use Web Application Bridge, you might also define a more specific rewrite rule than the rule given earlier. Example:

   ```
   RewriteRule ^/([^/\.]+)$             /wps/vanityurl/$1        [P]
   ```

   With this rewrite rule, the portal redirects only names that consist of only one segment and contain no periods. This rewrite rule avoids conflicts with file names, such as `shoe_sale.html`. For project work, modify the rule as required.

8. Optional: If required, configure the preview link in the vanity URL user interface to show the short vanity URL. The user interface for managing vanity URLs has a preview link. By default, this link goes directly to the vanity servlet, and the portal shows the full vanity URL, for example `http://`*`hostname`*`/wps/vanityurl/shoe-sale`. You can configure the preview link to point to the HTTP server instead, which shows the short version of the vanity URL. To configure the preview link target, you use the portal

configuration task `enable-vanityurl-httpserver-preview`. For more information, see *Configuring the vanity URL preview link*.

Related concepts:

"Integrating with web applications" on page 977
The web application bridge uses reverse proxy technology to integrate web-based content providers, such as the Microsoft SharePoint server, with IBM WebSphere Portal Express. Administrators must first define the virtual web applications or content providers. A lightweight iFrame portlet renders the content from the backend applications. Users can then access the iFrame on a page without requiring direct network access to the backend application. A special engine maps Uniform Resource Identifier (URIs) on the iFrame portlet to real URIs from the content providers.

Related tasks:

"Preparing a remote web server" on page 166
Install and configure the web server plug-in. The IBM WebSphere Application Server provides the plug-in. Configure the web server to communicate with IBM WebSphere Portal Express.

"Configuring the vanity URL preview link"
The user interface for managing vanity URLs has a preview link. By default, this link shows the full vanity URL. If you installed an HTTP server and configured it to allow short vanity URLs, you can configure the preview link to show the short vanity URL instead.

Related information:

IBM HTTP Server

Apache mod_rewrite

Apache mod_proxy

Configuring the vanity URL preview link

The user interface for managing vanity URLs has a preview link. By default, this link shows the full vanity URL. If you installed an HTTP server and configured it to allow short vanity URLs, you can configure the preview link to show the short vanity URL instead.

About this task

By default, the preview link points directly to the vanity URL servlet. Example:

`http://hostname/wps/vanityurl/shoe-sale`

If you installed an HTTP server and configured it to allow short vanity URLs, you can configure the preview link to point to the HTTP server instead. It then shows the short vanity URL, for example as follows:

`http://hostname/shoe-sale`

You switch between these two options by using the appropriate portal configuration tasks:

- If you want the preview link to point to the HTTP server, run the portal configuration task **en**`able-vanityurl-httpserver-preview`.
- If you want the link to go to the vanity servlet and show the full vanity URL, run the portal configuration task **dis**`able-vanityurl-httpserver-preview`. This setting is the default.

The syntax for these configuration tasks is as follows:

IBM i

```
ConfigEngine.sh enable-vanityurl-httpserver-preview
-DPortalAdminPwd=password -DWasPassword=password

ConfigEngine.sh disable-vanityurl-httpserver-preview
-DPortalAdminPwd=password -DWasPassword=password
```

Linux

```
./ConfigEngine.sh enable-vanityurl-httpserver-preview
-DPortalAdminPwd=password -DWasPassword=password

./ConfigEngine.sh disable-vanityurl-httpserver-preview
-DPortalAdminPwd=password -DWasPassword=password
```

Windows

```
ConfigEngine.bat enable-vanityurl-httpserver-preview
-DPortalAdminPwd=password -DWasPassword=password

ConfigEngine.bat disable-vanityurl-httpserver-preview
-DPortalAdminPwd=password -DWasPassword=password
```

Related tasks:

"Providing short vanity URLs" on page 2101
You might want to make your vanity URLs as short and simple as possible for your customers. You can create vanity URLs that contain only the vanity segment by omitting the string `/wps/vanityurl`. In this case, you must use a web server and define a rewrite rule. If you also use IBM Web Application Bridge, or if you have static files in the root of the HTTP server document directory, adapt the rewrite rule.

Synchronizing the vanity URL database

Vanity URLs are stored as part of the page in the JCR database in the portal page site area of Web Content Manager. For performance reasons, the data is also stored in the WebSphere Portal Express database. When the data is modified, the portal synchronizes the data between both sides. However, under certain circumstances it can happen that the data is not synchronized. For such cases, the portal provides a configuration task that synchronizes the data.

About this task

For example, if the JCR database on the Web Content Manager side is restored, but the portal database is not restored, the data is not synchronized any more. The configuration task for synchronizing the data is `sync-vanityurl-data`. It reads the data that is stored in Web Content Manager and updates the WebSphere Portal Express database.

Syntax

You call the task as follows:

IBM i

```
ConfigEngine.sh sync-vanityurl-data -DPortalAdminPwd=password
-DWasPassword=password
```

Linux

```
./ConfigEngine.sh sync-vanityurl-data -DPortalAdminPwd=password
-DWasPassword=password
```

Windows

```
ConfigEngine.bat sync-vanityurl-data -DPortalAdminPwd=password
-DWasPassword=password
```

Extra parameters:

You can specify the following parameters with this task. Each individual parameter requires the prefix `-D` on the command.

`RunParallel = (false)|true`

Use this parameter to specify whether you want the task to run with multiple threads or not. If you want the task to run in a single thread, specify the value `false`. This value is the default value. If you want the task to run with multiple threads, specify the value `true`. Each thread requires a database connection.

Parameters for virtual portals:

If you have virtual portals, the portal applies this configuration task to all virtual portals by default. To limit the task to a specific virtual portal, you identify the virtual portal by adding one of the following parameters to the command. Each individual parameter requires the prefix `-D` on the command.

`VirtualPortalHost`

Use this parameter to specify the host name of the virtual portal. For example, the host name can be `vp.example.com`.

Note: You can specify the `VirtualPortalHost` parameter alone only if the host name is unique. If the host name of the virtual portal is the same as the host name of the default virtual portal, you must also specify the `VirtualPortalContext` parameter.

`VirtualPortalContext`

Use this parameter to specify the virtual portal context that identifies the virtual portal. For example, the context can be `vp1`.

Example:

```
./ConfigEngine.sh sync-vanityurl-data
                  -DPortalAdminPwd=password -DWasPassword=password
                  -DVirtualPortalHost=vp.example.com
```

Related tasks:

"Enabling managed pages" on page 913
After migration, you must manually enable support for managed pages. Without managed pages support, some features like the project menu are not available on the migrated server.

Related reference:

"Troubleshooting pages" on page 1741
When you work with pages, you might encounter problems that are related to projects, access rights, or other issues.

Setting an error URI for undefined vanity URLs

You can configure how the portal behaves if a user tries to access an undefined vanity URL.

About this task

The vanity URL servlet resolves the appropriate portal page or content item according to the incoming vanity URL. In a request to an undefined vanity URL, this servlet responds by either of the following two ways:

- It sends a 404 return code.
- It redirects the user to a defined error URI. You can configure this error URI by using a portal configuration task.

The configuration task for configuring an error URI for vanity URLs is `set-vanityurl-error-uri`. It sets a new custom property in the Resource Environment Provider of the WP ConfigService. The property name is `vanityurl.error.uri`. Calling the configuration task sets the value for this property. If the property is not set in the Resource Environment Provider and a user requests an undefined vanity URL, the portal sends a 404 return code.

Syntax

You call the task as follows:

IBM i

```
ConfigEngine.sh set-vanityurl-error-uri
-DPortalAdminPwd=password -DWasPassword=password
```

Linux

```
./ConfigEngine.sh set-vanityurl-error-uri
-DPortalAdminPwd=password -DWasPassword=password
```

Windows

```
ConfigEngine.bat set-vanityurl-error-uri
-DPortalAdminPwd=password -DWasPassword=password
```

Extra parameters:

If you want the vanity URL to redirect the user to an error URI, you must specify the following parameter with this task. It requires the prefix `-D` in the command.

ErrorURI = *URI_for_redirect*

If the vanity URL of the request is not defined, the vanity URL servlet redirects the requests to the URI specified here. If the value for the parameter is the empty string and the vanity URL of the request is undefined, the portal sends a 404 return code.

Example:

```
./ConfigEngine.sh set-vanityurl-error-uri
                  -DPortalAdminPwd=password -DWasPassword=password
                  -DErrorURI=cm:oid:ibm.portal.Home
```

The error URI specified in this example redirects the user to the portal home page.

Related tasks:

"Enabling managed pages" on page 913
After migration, you must manually enable support for managed pages. Without managed pages support, some features like the project menu are not available on the migrated server.

Related reference:

"Troubleshooting pages" on page 1741
When you work with pages, you might encounter problems that are related to projects, access rights, or other issues.

Social rendering

IBM WebSphere Portal Express page editors can use social rendering to feature social data that is hosted on a remote IBM Connections server in the context of portal pages.

Page editors can create web content items that represent lists of social objects and detail views for individual social objects. A list of social objects shows the results of a specific query for social data from Connections. For example, the list can

consist of specific blog posts, files, or discussion topics. A detail view shows detailed information about a specific social object.

Page editors can control the visual appearance of the social data that is displayed on your portal pages. They do so by selecting the formatting component of choice from a predefined, yet extensible set of formatting components. These components are also called appearance components.

Site designers create the appearance components by using the IBM Web Content Manager Authoring portlet. They define them in a style that is consistent with your portal pages. The appearance component concept provides a clear separation between the following two roles:

- Website designers who define the corporate appearance of your website and deliver the appearance components to your page editors
- Page editors who build your portal pages by assembling the appropriate set of components and content to a meaningful context for your site visitors.

With this concept, your page editors do not need to know the markup generation and CSS styling details of the appearance components. Even without that knowledge they can still choose from a rich, but consistent set of visual designs for your social data.

To control the social data that is displayed on your portal pages, your page editors use inline editing on the underlying social rendering content items. The social rendering content items are also referred to as view definitions. They combine both the selection logic that defines which social objects are displayed and the appearance component selection that controls the visual representation of the data. For example, they can show the most recent blog entries that are created in the Connections community to which the current page is associated. Page editors can select individual view definitions from a set of predefined view definitions and drop them onto portal pages. When a page editor drops a view definition onto a portal page, the view definition is copied to the page. The page editor can then modify it independently of other view definitions on the same page or other pages.

As a result, with social rendering the social data displayed on your portal pages is fully controlled by Web Content Manager. The appearance components define the visual appearance by which the social data is rendered on the page. The view definition content items define which data is selected and which appearance component is used to visualize the data. This way, you control the social data that is rendered on your portal pages in the same way as other web content. Social rendering includes support for projects, versioning, workflows, and syndication.

Social rendering provides you the following components:

- The following set of view definitions for social lists:
  - List of Blog Posts
  - List of Communities
  - List of Community Events
  - List of Community Forum Topics
  - List of Community Blog Posts
  - List of Community Content
  - List of Community Files
  - List of Files
  - List of Forum Topics

  - List of People
- A view definition for forum topic details named Forum Topic Details
- Two appearance components for visualizing the social lists in two different ways:

  Simple:
  A condensed simple list design

  Comprehensive:
  A comprehensive list design
- Two appearance components for visualizing the details of an individual forum topic:

  Forum Topic Details:
  An appearance component for visualizing the detailed information about the topic itself

  Replies
  An appearance component for visualizing the nested thread of replies for the forum topic.
- A set of DDC list-rendering profiles. You can use them to extend social rendering. They are based on the IBM Digital Data Connector (DDC) for WebSphere Portal Express.

You can use these view definitions and appearance components as starting points for creating your own solution. Do not modify the view definitions and appearance components that WebSphere Portal Express provides, but copy them and modify the copies.

To use social rendering, you need to set up your WebSphere Portal Express for integration with Connections.

"Roadmap: How to work with social rendering" on page 2109
Before you can use social rendering in your IBM WebSphere Portal Express, you need to set up IBM Connections integration. After you do that, your users can start working with the default view definitions that social rendering provides. You might want to configure and customize the social data that is rendered on your portal pages for your site visitors. In this case, you can configure and administer social rendering as required for your portal. You can also customize the view definitions, or create your own custom appearance components.

"Working with lists of social objects" on page 2110
You can design lists of social objects in different ways that are related to their content, visual design, and other characteristics. You can also use the IBM WebSphere Portal Express Tag Cloud portlet in combination with your lists of social objects, so that your site visitors can navigate social data based on tags. These tasks are usually done by page editors.

"Configuring global settings for social rendering" on page 2121
You can apply some global configuration settings to social rendering. These global settings apply to all social lists in your WebSphere Portal Express.

"Administering social lists" on page 2131
You can administer several aspects of social lists. For example, you can add your own social lists to the content shelf, or you can tune the social object caches for performance.

"Customizing view definitions for portal site visitors" on page 2136
You can customize social rendering view definitions in a number of ways. For example, you can determine the social data that is shown and the visual appearance of the data. You customize social rendering by working with the

social list definition authoring template. For custom view definitions, you can also create your own authoring templates.

"Adding widgets to a community" on page 2235
As an owner of an IBM Connections community, you can define the set of widgets that are available in the community. For this purpose, you use the Customize option in Connections. For example, you might choose to add the Blog widget to a specific community. This way, community owners have control over the set of services available in the communities they own.

"Extending social lists by using the digital data connector" on page 2236
The social rendering feature in IBM WebSphere Portal Express Version 8.5 is implemented as a IBM Digital Data Connector (DDC) for WebSphere Portal Express plug-in.

Related concepts:

"IBM Digital Data Connector (DDC) for WebSphere Portal Express" on page 3255
You can use the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework to integrate data from external data sources on your portal pages by using IBM Web Content Manager presentation components. External data means that the data does not need to be stored directly in IBM Web Content Manager. For example, you can use DDC to render social data that you have on your IBM Connections server or on other social platforms in the context of your portal pages. Other possible data sources include news feeds, task lists, product catalog information, to name just a few.

Related reference:

"Digital Data Connector profiles for social rendering" on page 2147
Starting with Version 8.5, IBM WebSphere Portal Express includes a set of IBM Digital Data Connector (DDC) for WebSphere Portal Express profiles. You can use them with the social rendering DDC plug-in that is identified by the extension ID `ibm.portal.ddc.sr`.

Roadmap: How to work with social rendering

Before you can use social rendering in your IBM WebSphere Portal Express, you need to set up IBM Connections integration. After you do that, your users can start working with the default view definitions that social rendering provides. You might want to configure and customize the social data that is rendered on your portal pages for your site visitors. In this case, you can configure and administer social rendering as required for your portal. You can also customize the view definitions, or create your own custom appearance components.

To go the fast path of using social lists, proceed as follows:

1. Set up Connections integration.
2. If you want to use the Forum Topic Details view, you must also install the Connections page resolver component.
3. You can now work with the default social lists immediately. Read the tasks that are listed under *Working with lists of social objects*. These tasks are done by a page editor or by site visitors.

To go the advanced path of working with social lists, proceed as follows:

1. Set up Connections integration.
2. If you want to use the Forum Topic Details view, you must also install the Connections page resolver component.
3. You can now work with the default social lists immediately. Read the tasks that are listed under *Working with lists of social objects*. These tasks are done by a page editor or by site visitors.

4. To configure social rendering in your portal, read *Configuring global settings for social rendering*. These tasks are done by an administrator.
5. To administer social rendering, read *Administering social lists*. These tasks are done by an administrator.
6. To customize your social rendering view definitions and appearance components, read *Customizing social lists for portal site visitors*. These tasks are done by a website designer.
7. To enable your page editors to create your own social lists, read *Creating custom authoring templates for social lists*. This work task is done by a website designer.

Related concepts:

"Roadmap: Integrating with IBM Connections" on page 97

Related tasks:

Integrating with IBM Connections
Connections portlets give IBM WebSphere Portal Express users access to additional collaboration and social networking features such as Activities, Blogs, and Bookmarks. Users can also view Connections business card information and tags and link to Connections features directly from the portal server.

"Working with lists of social objects"
You can design lists of social objects in different ways that are related to their content, visual design, and other characteristics. You can also use the IBM WebSphere Portal Express Tag Cloud portlet in combination with your lists of social objects, so that your site visitors can navigate social data based on tags. These tasks are usually done by page editors.

"Configuring global settings for social rendering" on page 2121
You can apply some global configuration settings to social rendering. These global settings apply to all social lists in your WebSphere Portal Express.

"Administering social lists" on page 2131
You can administer several aspects of social lists. For example, you can add your own social lists to the content shelf, or you can tune the social object caches for performance.

"Customizing view definitions for portal site visitors" on page 2136
You can customize social rendering view definitions in a number of ways. For example, you can determine the social data that is shown and the visual appearance of the data. You customize social rendering by working with the social list definition authoring template. For custom view definitions, you can also create your own authoring templates.

"Creating custom authoring templates for list definitions" on page 2230
Social rendering provides you a set of view definitions that you can use to add social data to your portal pages. These list view definition content items are created from the social list definition IBM Web Content Manager authoring template. You can create your own custom list view definitions by creating new content items from this authoring template. To customize social rendering even further, you can also create your own authoring templates to extend the data set that makes up your list view definitions.

Related information:


IBM Connections Portlets for WebSphere Portal

Working with lists of social objects

You can design lists of social objects in different ways that are related to their content, visual design, and other characteristics. You can also use the IBM WebSphere Portal Express Tag Cloud portlet in combination with your lists of

social objects, so that your site visitors can navigate social data based on tags. These tasks are usually done by page editors.

About this task

"Concept of the lists of social objects provided with the social rendering feature"
The social rendering feature provides you with a set of predefined lists view definitions and a detail view definition for forum topic details.

"Using the view definitions provided with social rendering on your portal pages" on page 2115
The social rendering feature provides you with a set of predefined view definitions. You can add them to the pages of your WebSphere Portal Express and modify them according to your requirements. For example, you can define which types of social objects are listed, how they are filtered, sorted, and presented. These tasks are done by a page editor.

"Using social lists with your own custom theme" on page 2117
The social lists that social rendering provides work with portal pages that have the Portal 8.5 theme with a Basic Content theme profile. To use them with your own custom theme, you add the `wp_social_rendering_85` theme module to your theme.

CF03 "Configuring a page with lists of social objects for Tag Cloud support" on page 2117
Learn how to enable Tag Cloud portlet support on your portal page. You can enable Tag Cloud support if you are using a default portal theme profile or a custom portal theme profile.

"Using the portal Tag Cloud with lists of social objects" on page 2118
To get the most benefit from social rendering together with tagging, you can put the WebSphere Portal Express Tag Cloud portlet on pages that contain social lists. This way your site visitors can reduce the contents of social lists by selecting individual tags from the tag cloud. The list of social objects is then restricted to content items that are tagged with the tag or tags that the user selected.

Related tasks:

Integrating with IBM Connections
Connections portlets give IBM WebSphere Portal Express users access to additional collaboration and social networking features such as Activities, Blogs, and Bookmarks. Users can also view Connections business card information and tags and link to Connections features directly from the portal server.

"Enabling remote rendering with WSRP and the Web Content Viewer" on page 2069
To display web content on a portal that does not include IBM Web Content Manager, you can use the Web Content Viewer and the WSRP support in the portal. The Web Content Viewer can then retrieve and display content from a web content system on a different server.

Concept of the lists of social objects provided with the social rendering feature

The social rendering feature provides you with a set of predefined lists view definitions and a detail view definition for forum topic details.

The lists of social objects are defined by web content items of content type Social List Definition. The following lists are available after you installed IBM WebSphere Portal Express:

- List of Blog Posts
- List of Communities
- List of Community Events
- List of Community Forum Topics
- List of Community Blog Posts
- List of Community Content
- List of Community Files
- List of Files
- List of Forum Topics
- List of People

These list definitions define individual queries against the search service of the remote IBM Connections server. The specific queries for the individual list definitions are built based on the element data that is contained in those content items. In addition to the query, the content items also hold the reference to the list appearance component responsible for transforming the query result data into visual markup.

The lists generate a visual enumeration of social objects that are relevant in the specific context of a portal page. For example, these social objects can be all recent blog posts that are tagged with a specific tag. To see a detail view of a social object, a site visitor can click the link that is rendered for the item in the list. The process of bringing up this details view is called social object resolution. You can choose between the following options for resolving a social object that a user clicks:

- Taking the user to a portal page where the social object details are rendered by a social rendering detail view
- Taking the user to a portal page where the social object details are rendered by a Connections portlet
- Taking the user to the details view for the social object in the Connections user interface.

The resolution process can be controlled by various concepts described in *Configuring globally how social object links are resolved*.

In contrast to social list view definitions, social rendering detail view definitions render detailed information about a single social object. For example, such detail information can be the full body text and the nested thread of replies for an individual forum topic. Additionally, the details view typically also renders user interfaces that allow interactions with the social object, such as form fields by which site visitors can post new replies. WebSphere Portal Express Version 8.5 provides a social rendering details view definition for forum topics named Forum Topic Details.

You can create more list and details view definitions as extensions within the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework.

"Social object resolution" on page 2113
When a portal user clicks a link to an object, the portal takes the user to the details view of that object. This process is called social object resolution. For example, a user might click a specific forum topic that is listed in the Community Forum Topics list. In this case, the social object resolution takes the user to a portal page that provides a details view of the forum topic that the user clicked. You can influence the result of the resolution that the user views by setting various parameters. These parameters are described here.

Related concepts:

"IBM Digital Data Connector (DDC) for WebSphere Portal Express" on page 3255
You can use the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework to integrate data from external data sources on your portal pages by using IBM Web Content Manager presentation components. External data means that the data does not need to be stored directly in IBM Web Content Manager. For example, you can use DDC to render social data that you have on your IBM Connections server or on other social platforms in the context of your portal pages. Other possible data sources include news feeds, task lists, product catalog information, to name just a few.

Related tasks:

"Configuring globally how social object links are resolved" on page 2127
You can include one or many attributes of social objects in the design component that defines the visual design of your social list. Among other features, social objects have different resolvable links that enable users to open details views of the social objects or the community to which the social objects belong. If you plan to add these links to your social list, you can decide how you want the social objects and their home community to be resolved when users click the corresponding links. IBM WebSphere Portal Express can either resolve the links in the context of the portal itself, or redirect the user to the IBM Connections user interface. You can globally configure how these types of links of social objects are resolved for the users. To do so, you use a setting in the WP Connections Integration Service resource environment provider in the WebSphere Integrated Solutions Console.

Social object resolution:

When a portal user clicks a link to an object, the portal takes the user to the details view of that object. This process is called social object resolution. For example, a user might click a specific forum topic that is listed in the Community Forum Topics list. In this case, the social object resolution takes the user to a portal page that provides a details view of the forum topic that the user clicked. You can influence the result of the resolution that the user views by setting various parameters. These parameters are described here.

The social object resolution process is implemented by the connections resolver that is contained in the IBM Connections Portlets for IBM WebSphere Portal Express. To enable social object resolution, you need to install this resolver first.

The resolution result for a specific social object depends on the following parameters:

1. The type of link that points to the social object. This link can be either of the following two types:
   - A portal URL. In this case, the link is called a POC (piece of content) link.
   - An Connections URL.
2. The resolution root page parameter that is defined in your view definition.
3. Whether the social object is contained in a community or not, and if it is contained in a community, which community that is.
4. The type of object that the user clicked, for example whether it is a forum topic or a blog post.

The type of link that points to the social object determines the resolution outcome as follows:

Connections URL:

If the link to the social object is an Connections URL, the social object

resolution process is not triggered. Instead, the user is taken directly to the corresponding Connections user interface that shows the details of the object that the user clicked. You can set the portal-wide default type for your social object links in the portal configuration. For more information about how to do so, read *Configuring globally how social object links are resolved*. You can always overrule that default in your appearance components by using the `[AttributeResource attributeName=""]` tag as follows:

- To generate default links to the current social object, specify `[AttributeResource attributeName="link"]`.
- To generate a WebSphere Portal Express type link, specify `[AttributeResource attributeName="portalLink"]`.
- To generate an Connections type link, specify the `[AttributeResource attributeName=rawLink]`.

WebSphere Portal Express URL:

If the link to the social object is a WebSphere Portal Express URL, the connections resolver starts the social object resolution. It evaluates this process in the following three phases:

1. Determine the set of candidate pages
2. Determine the appropriate page from the set of candidate pages
3. Determine the final fallback resolution target for the case that the resolver finds no matching page from the set of candidate pages.

The connections resolver determines the candidate pages based on the resolution root page parameter and the ID of the community that contains the social object. You can specify the resolution root page parameter in the social list definition authoring template by using the element Custom Resolution Root Page. For more information, read *Customizing list view definitions by using inline editing*. You can use the resolution root page parameter to identify a specific portal page by using its custom unique name. To identify the current portal page that renders the current view definition, you can also use the value `current`. If you identify a portal page by using the resolution root page parameter, the connections resolver considers the identified page and all of its descendant pages to be candidate pages for the current social object resolution process.

Note: The value of the resolution root page parameter is passed to the connections resolver as a root page URL parameter. If the resolution root page parameter is not set and the social object belongs to a community, the connections resolver considers all portal pages that are associated with this community to be candidate pages. For more information, read *Managing community associations*.
If the set of candidate pages is not empty, the social object resolution process tries to determine the appropriate page from the set of candidate pages. To do so, it searches the candidate pages for a page that contains a suitable viewer portlet that matches the type of the social object that is to be resolved. Whether a portlet entity can render a detail view for a specific social object is determined based on specific portlet preferences. For example, the drag-and-drop configuration for the Forum Topic Details view definition sets the portlet preference `IC_Forums.topic=true` to indicate that this portlet can render a detail view for a forum topic. For a full list of the supported portlet preferences, read *Connections Portlets for WebSphere Portal Express*.

If the connections resolver finds no candidate page or candidate page that contains a suitable viewer portlet, the portal takes the user to the default page for the social object that the user clicked. To identify the default pages for specific types of social

objects, you assign those pages-specific unique names. For example, the default name for identifying the default page for social objects that are served by the Forums service is `ibm.conn.forums`. An administrator can change the default unique names. For more information, read *Connections Portlets for WebSphere Portal Express: Configuring unique names*.

If the connections resolver does not find the default page, the final fallback is either to redirect the user to a defined portal error page or to take the user to the details view in the Connections user interface. An administrator can control this behavior by running the portal configuration tasks `disable-poc-redirect-to-connections` or `enable-poc-redirect-to-connections`. For more details about the social object resolution, read *Connections Portlets for WebSphere Portal Express: IBM Connections Portlets for WebSphere Portal: Configuring navigation between portlets*.

Related tasks:

"Managing community associations" on page 1290
You can create, view, modify, or delete community associations on a page with the Page Associations window, the XML configuration interface, and Portal Scripting Interface.

"Configuring globally how social object links are resolved" on page 2127
You can include one or many attributes of social objects in the design component that defines the visual design of your social list. Among other features, social objects have different resolvable links that enable users to open details views of the social objects or the community to which the social objects belong. If you plan to add these links to your social list, you can decide how you want the social objects and their home community to be resolved when users click the corresponding links. IBM WebSphere Portal Express can either resolve the links in the context of the portal itself, or redirect the user to the IBM Connections user interface. You can globally configure how these types of links of social objects are resolved for the users. To do so, you use a setting in the WP Connections Integration Service resource environment provider in the WebSphere Integrated Solutions Console.

Related information:

IBM Connections Portlets for WebSphere Portal

IBM Connections Portlets for WebSphere Portal: Configuring unique names

IBM Connections Portlets for WebSphere Portal: Configuring navigation between portlets

Using the view definitions provided with social rendering on your portal pages

The social rendering feature provides you with a set of predefined view definitions. You can add them to the pages of your WebSphere Portal Express and modify them according to your requirements. For example, you can define which types of social objects are listed, how they are filtered, sorted, and presented. These tasks are done by a page editor.

About this task

Notes:

- Social rendering view definitions are represented by Web Content Manager content items. Therefore, you can drag and drop these view definitions only to managed pages or pages that have an explicitly defined web content association to a site area in Web Content Manager. For more information, see *Web content associations*.

- The list view definitions that are provided with social rendering use the Connections search service remote API to retrieve the social objects that are displayed in the list. This action has the following consequences:
  - Only information that is available in the Connections search feed can be displayed in the social lists.
  - Updates to social objects in Connections do not appear in the social lists that are provided with social rendering until the index of the Connections search service was updated.

  For more details about how to administer the Connections search service, read the information about *Administering Search* in the Connections product documentation.

Procedure

To add a social rendering view definition to a portal page and modify it as required, proceed as follows:

1. Select the **Edit Mode** of the page.
2. Add the social list of your choice to the portal page:
   a. Select **Create** > **Content** from the toolbar.
   b. Open the **Social Content** site area from the selection list.
   c. Select the view definition of your choice and place it on the page by drag-and-drop or by clicking the **Add this content to the page** plus sign icon. You can choose from the following types of lists:
      - List of Community Files
      - List of Community Content
      - List of Community Blog Posts
      - List of Community Forum Topics
      - List of Community Events
      - List of Files
      - List of Social Content
      - List of Blog Posts
      - List of Forum Topics
      - List of Communities
      - List of People
      - Forum Topic Details.
   d. Save your changes.
3. Remaining in page edit mode, click **Display component action menu** in the social list portlet skin. The component action menu opens.
4. Select the **Open Edit Form** menu item. The Web Content Manager inline edit dialog opens up and shows the authoring experience for the view definition.
5. Modify the social list definition to serve the information appropriate to the current page. For a list and descriptions of the available options, see *Customizing social list definitions by using inline editing*.

Related concepts:

"Web content associations" on page 2023
Web content associations are used to combine portal pages and associated web content items that are managed by IBM Web Content Manager so they can be managed and rendered consistently. Web content associations map portal pages to the site structure in the IBM Web Content Manager system.

Related tasks:

"Customizing social list definitions by using inline editing" on page 2137
You can customize your list view definitions by defining the following settings in inline editing mode. List view definitions are represented by content items of the social list definition authoring template. This customization is normally done by a web designer or a page editor.

Related information:

Administering Search

Using social lists with your own custom theme

The social lists that social rendering provides work with portal pages that have the Portal 8.5 theme with a Basic Content theme profile. To use them with your own custom theme, you add the `wp_social_rendering_85` theme module to your theme.

About this task

The predefined social lists require that the social rendering CSS styles are loaded by the theme that is associated to the page with the social list. The theme loads these styles if the theme profile assigned to the current theme incorporates the `wp_social_rendering_85` theme module. The Portal 8.5 theme with either full or deferred profiles fulfills this requirement.

If you want to use social rendering with your own custom theme or a custom theme profile, add the `wp_social_rendering_85` theme module to that theme profile. For example, you can do so by using the `add-theme-modules` configuration task. If the theme profile that is active on the current page does not contain the social rendering theme module, the social rendering CSS classes are missing. In this case, the markup does not look as expected.

Related tasks:

"Customizing the CSS styles of social lists" on page 2225
The lists of social objects rely on the availability of several CSS class definitions in the `wp_social_rendering_85` theme module.

"Adding the social rendering theme module to a theme profile" on page 2132
For the social lists provided by social rendering to work, the theme that renders these lists needs to load CSS and JavaScript files. To include the CSS and JavaScript files in your theme, you need to add the `wp_social_rendering_85` theme module to your theme.

Configuring a page with lists of social objects for Tag Cloud support

Learn how to enable Tag Cloud portlet support on your portal page. You can enable Tag Cloud support if you are using a default portal theme profile or a custom portal theme profile.

Procedure

- If you are using default portal theme profiles within your portal installation, complete the following steps to enable Tag Cloud portlet support on your portal page:
  1. Locate the page in the Manage pages portlet.
  2. Click **Edit Page Properties** for the specific page.
  3. Click **Advanced options**.
  4. Click **I want to set parameters**.

5. Add the new parameter **resourceaggregation.profile** with the value set as the profile override for the specific page. For example, `profiles/profile_search_tag.json`.

   Note: The search_tag theme profile is a hidden profile, therefore it cannot be selected in the theme profile drop-down list. This profile needs to be set manually.
6. Click **Add**.
7. Click **OK** to save the new parameter.
8. Click **OK** to save your changes to the page properties.

- If you are using a custom theme, complete the following steps to enable Tag Cloud support on your portal page:
  1. Verify that your page inherits the default portal theme or a custom theme that supports the default theme profiles.
  2. Locate the theme profile definition file by using WebDAV. For more information about accessing theme profile definition files, see *Profile schema definition* in the related links.
  3. Open your theme profile `JSON` file and add the following lines to the `moduleIDs` JSON array:

     Note: If one or more of the following theme modules are already in the list of `moduleIDs`, you do not have to add them again.
     - `wp_pagebuilder_ui`
     - `wp_tagging_rating_tagcloud`
     - `dojo`
  4. Save your changes and restart your portal server.

Related concepts:

"Profile schema definition" on page 2558
You can write a profile schema with valid JSON.

Using the portal Tag Cloud with lists of social objects

To get the most benefit from social rendering together with tagging, you can put the WebSphere Portal Express Tag Cloud portlet on pages that contain social lists. This way your site visitors can reduce the contents of social lists by selecting individual tags from the tag cloud. The list of social objects is then restricted to content items that are tagged with the tag or tags that the user selected.

About this task

To configure tag selection for social lists from the tag cloud, a page editor proceeds by the following steps:

Procedure

1. CF03 Set a theme profile that supports lists of social objects as well as the Tag Cloud portlet. For more information, see *Configuring a page with lists of social objects for Tag Cloud support* in the related links.
2. Add the WebSphere Portal Express Tag Cloud portlet to the page that contains the list of social objects.
3. Optional: Depending on your portal theme, you might need to select **Edit Mode** for the page. Some themes require this step before you can select the Edit Shared settings option for the Tag Cloud portlet in the next step.

4. Select the **Edit Shared Settings** mode of the Tag Cloud portlet. If you cannot open the portlet menu, first select the **Edit Mode** for the page as described in the previous step.
5. Configure the Tag Cloud portlet to listen to Transmitted Tags. Set the **Tag display modes** setting to **Transmitted tags only**. For more information about configuring the Tag Cloud, see *Configuring the portal Tag Cloud for social rendering*.
6. Set the portlet **Action Mode** setting to **Expose public render parameter**.
7. Click **OK** to save your changes.
8. Enable the **Tag Selection Support** option for the list of social objects for which you want to make the Tag Cloud available to your users. For more information about how to set options for social lists, see *Customizing social list definitions by using inline editing*.

Results

After you set this configuration, the Tag Cloud portlet communicates with those lists of social objects on the page for which the Tag Selection Support option is activated in their social list definition. The tag cloud is filled dynamically with all tags associated with any of the individual social objects that are contained in those lists. If a user selects one or more tags from the tag cloud, those social lists on the page that have the Tag Selection Support option activated in their social list definition are dynamically filtered to contain only those entries that are tagged with all of the tags that the user selected from the tag cloud. Lists of social objects that have the Tag Selection Support option disabled in their definition do not transmit their tags to the tag cloud. These lists are not affected when site users select tags in the tag cloud.

Notes:

- By default, the Tag Selection Support option is disabled.
- If a user selects tags in the tag cloud, that tag selection remains active until one of the following conditions occurs:
  - The user deselects the tags again in the tag cloud.
  - The render state of the page that the user views is cleared. For example, this occurs when the user logs out from the portal.
- The page editor can add a general tag filter to the social list definition. To do this, the Page Editor specifies tag names in the **Filter by Tags** option of the social list definition. Such a tag filter is always active for all site visitors. Users cannot remove the tags that are specified in that filter by deselecting them in the tag cloud. Additional tags that a user selects by using the tag cloud are dynamically added to the list of tags that are applied to the list of social objects.
- If a social list has many different tags associated with it, the tag cloud might show only a subset of the available tags. This restriction can depend on two configuration options:
  - You can configure the limit for tags that an individual social list transmits from Connections. You set this limit in the WP Connections Integration Service resource environment provider. For more information, see the topic about *Configuring the tags transmission limit*.
  - You can configure the Tag Cloud portlet to show only a limited number of tags. This subset contains the most relevant tags, in other words tags that users assigned most often.
- The tag selection support for social list does not rely on tag federation between WebSphere Portal Express and Connections to be enabled.

- The social lists do not support interactions with the **Tags** portlet that is provided as one of the Connections portlets for WebSphere Portal Express.

  "Configuring the portal Tag Cloud for social rendering"
  If you include the WebSphere Portal Express Tag Cloud portlet on your social pages, you need to configure it to work with social lists.

Related tasks:

"Configuring the portal Tag Cloud for social rendering"
If you include the WebSphere Portal Express Tag Cloud portlet on your social pages, you need to configure it to work with social lists.

"Customizing social list definitions by using inline editing" on page 2137
You can customize your list view definitions by defining the following settings in inline editing mode. List view definitions are represented by content items of the social list definition authoring template. This customization is normally done by a web designer or a page editor.

CF03 "Configuring a page with lists of social objects for Tag Cloud support" on page 2117
Learn how to enable Tag Cloud portlet support on your portal page. You can enable Tag Cloud support if you are using a default portal theme profile or a custom portal theme profile.

"Configuring the tags transmission limit" on page 2124
You can configure your lists of social objects that retrieve data from the IBM Connections to transmit tags to the Tag Cloud portlet. You can limit the number of tag names that are loaded from Connections. You configure this limit in the WP Connections Integration Service resource environment provider.

Related reference:

Configuring the tag cloud
Administrators can configure the Tag Cloud portlet. You can configure each individual tag cloud portlet instance separately. To do this, use the **Edit Shared Settings** option from the portlet menu.

Configuring the portal Tag Cloud for social rendering:

If you include the WebSphere Portal Express Tag Cloud portlet on your social pages, you need to configure it to work with social lists.

About this task

You can configure each individual Tag Cloud portlet instance separately. Use the Edit Shared Settings option from the portlet menu and configure the **Tag display modes** option. You can choose between two tag display modes. Depending on the option that you select, the tag cloud displays either persisted tags or transmitted tags only. There is no option to display both types of tags at the same time. For tagging to work with social lists, you need to set this option to Transmitted tags only.

Persisted tags only
: Persisted tags are served by the portal server and can include federated tags, depending on your tagging federation settings. This setting is the default setting.

Transmitted tags only
: Transmitted tags are transmitted by portlets on the same page. As tag transmission within the browser works without server interaction. The transmitted tags are displayed without further filtering. For your users to be able to tag items in social lists, you need to set the tag display mode to

this option. In transmitted tags mode, the Tag Cloud always shows the All view. Other views are not available in this mode. Users cannot switch views in this mode.

These options are added to the ones that are described in *Configuring the Tag Cloud*.

Related reference:

Configuring the tag cloud
Administrators can configure the Tag Cloud portlet. You can configure each individual tag cloud portlet instance separately. To do this, use the **Edit Shared Settings** option from the portlet menu.

Configuring global settings for social rendering

You can apply some global configuration settings to social rendering. These global settings apply to all social lists in your WebSphere Portal Express.

About this task

Typically, an administrator configures these settings.

To configure these settings, you configure the appropriate properties in the WP Connections Integration Service resource environment provider. You access this service in the WebSphere Integrated Solutions Console under the resource environment providers.

These properties define the default settings if no other values are set for individual social lists in the corresponding social view definitions. Page editors with the appropriate access permissions can override some of these settings for individual social lists.

"Configuring the Connections server type" on page 2122
You can use social rendering with an on-premises IBM Connections server or with an Connections server that runs in the Smart Cloud for Social Business. If you use the latter type of connections server, you need to adapt the configuration accordingly.

"Configuring the maximum number of items loaded from Connections" on page 2123
You can define a value for the maximum number of social objects that you want the IBM Connections to return when data for a list of social objects is requested.

"Configuring portal user ID conversion based on directory service" on page 2124
When you use social rendering with an IBM Connections server, you must configure both IBM WebSphere Portal Express and Connections against the same directory service. Depending on the directory service that you use, WebSphere Portal Express needs to convert the user IDs of portal users to a format that the Connections server accepts. For example, this conversion is required if both WebSphere Portal Express and Connections are configured against a Domino Directory service or a Microsoft Active Directory.

"Configuring the tags transmission limit" on page 2124
You can configure your lists of social objects that retrieve data from the IBM Connections to transmit tags to the Tag Cloud portlet. You can limit the number of tag names that are loaded from Connections. You configure this limit in the WP Connections Integration Service resource environment provider.

"Configuring globally how social object data is served" on page 2125
Among other features, social objects have different links that enable users to download related data. For example, users can use links to download a file or the profile image of the author. You can globally configure how social object data is served to the users. To do so, you use a setting in the WP Connections Integration Service resource environment provider in the WebSphere Integrated Solutions Console.

"Configuring globally how social object links are resolved" on page 2127
You can include one or many attributes of social objects in the design component that defines the visual design of your social list. Among other features, social objects have different resolvable links that enable users to open details views of the social objects or the community to which the social objects belong. If you plan to add these links to your social list, you can decide how you want the social objects and their home community to be resolved when users click the corresponding links. IBM WebSphere Portal Express can either resolve the links in the context of the portal itself, or redirect the user to the IBM Connections user interface. You can globally configure how these types of links of social objects are resolved for the users. To do so, you use a setting in the WP Connections Integration Service resource environment provider in the WebSphere Integrated Solutions Console.

"Configuring file type icon mappings" on page 2129
Social rendering provides two types of list appearance components for result lists: `simple` and `comprehensive`. Both list appearances components show file type-specific icons when they render list entries that refer to individual files. In this case, the file type is determined based on the file extensions of the individual files. You can configure the set of file types that you want to use in the WP Connections Integration Service resource environment provider. In the context of social lists, a file type is defined by a file type name and a list of file extensions. This list defines the mapping between individual files and your file types. In your social list appearance components, you can then access the file type name for a specific file by using the Web Content Manager `[AttributeResource attributeName="fileType"]` tag. That tag is defined in the IBM Digital Data Connector (DDC) for WebSphere Portal Express profile. You can then use the file type name to render the appropriate image, for example, by assigning a corresponding CSS class.

Related tasks:

CF03 "Using the business card" on page 2223
You can integrate the business card and online status in a list of social objects by using live text micro format.

Configuring the Connections server type

CF03

You can use social rendering with an on-premises IBM Connections server or with an Connections server that runs in the Smart Cloud for Social Business. If you use the latter type of connections server, you need to adapt the configuration accordingly.

About this task

To configure the Connections server type, you set a custom property named `server.type` in the WP Connections Integration Service resource environment provider in the WebSphere Integrated Solutions Console. Supported values for this property are as follows:

on-premise
Specify this value if the Connections server runs on your premises. This value is the default value.

SC4SB
Specify this value if your Connections server runs in the Smart Cloud for Social Business.

Procedure

1. Log in to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Select **WP ConnectionsIntegrationService**.
4. Under **Additional properties**, click **Custom properties**.
5. Locate the property named `server.type`. If this property does not exist yet, create it.
6. Set the value for the `server.type` property to one of the values listed earlier as required.
7. Save your changes.
8. Restart your portal server for the changes to take effect.

What to do next

Related tasks:

CF03 "Configuring a connection between WebSphere Portal Express and IBM Connections in SmartCloud for Social Business" on page 769
Learn how to configure the Connections integration assets to complete calls to IBM Connections in SmartCloud for Social Business.

Configuring the maximum number of items loaded from Connections

You can define a value for the maximum number of social objects that you want the IBM Connections to return when data for a list of social objects is requested.

About this task

To configure this maximum value, proceed by the following steps:

Procedure

1. Log in to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Select **WP ConnectionsIntegrationService**.
4. Under **Additional properties**, click **Custom properties**.
5. Edit the value for the `search.page.size` property as required. The default value is `50`.
6. Save your changes.
7. Restart your portal server for the changes to take effect.

What to do next

Performance note: Setting this value to a high figure can affect the response time of the Connections server.

Note: A page editor can override this default value for an individual social list by editing the social list definition and setting the value in the field Maximum Results. For more information, see *Customizing social list definitions by using inline editing*.

Related tasks:

"Customizing social list definitions by using inline editing" on page 2137
You can customize your list view definitions by defining the following settings in inline editing mode. List view definitions are represented by content items of the social list definition authoring template. This customization is normally done by a web designer or a page editor.

Configuring portal user ID conversion based on directory service

When you use social rendering with an IBM Connections server, you must configure both IBM WebSphere Portal Express and Connections against the same directory service. Depending on the directory service that you use, WebSphere Portal Express needs to convert the user IDs of portal users to a format that the Connections server accepts. For example, this conversion is required if both WebSphere Portal Express and Connections are configured against a Domino Directory service or a Microsoft Active Directory.

About this task

To configure the user ID conversion, set the following custom property in the WP Connections Integration Service Resource Environment Provider in the WebSphere Integrated Solutions Console:

`use.userid.conversion`
Set this property to one the following values:

`true`
Specify this value if you want to apply the user ID conversion.

`false`
Specify this value if you do not want the user ID conversion to be applied. This value is the default value.

Supported values for this property are as follows:

Procedure

1. Log in to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Select **WP ConnectionsIntegrationService**.
4. Under **Additional properties**, click **Custom properties**.
5. Locate the property `use.userid.conversion`. If this property is not listed yet, create it new.
6. Set the value for the property `use.userid.conversion` to one of the values that are listed earlier.
7. Save your changes.
8. Restart your portal server for the changes to take effect.

Configuring the tags transmission limit

You can configure your lists of social objects that retrieve data from the IBM Connections to transmit tags to the Tag Cloud portlet. You can limit the number of tag names that are loaded from Connections. You configure this limit in the WP Connections Integration Service resource environment provider.

About this task

Notes:

- Loading too many tags can affect performance.
- You set this maximum limit in the WP Connections Integration Service. The Tag Cloud can further reduce the number of shown tag names. This reduction depends on the configuration of the tag cloud.

To configure the tags transmission limit, proceed by the following steps:

Procedure

1. Log in to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Select **WP ConnectionsIntegrationService**.
4. Under **Additional properties**, click **Custom properties**.
5. Edit the value for the `transmitted.tags.limit` property as required. The default value is `100`.
6. Save your changes.
7. Restart your portal server for the changes to take effect.

What to do next

For more information, see the topic about *Using the portal Tag Cloud with social lists*.

Related tasks:

"Using the portal Tag Cloud with lists of social objects" on page 2118
To get the most benefit from social rendering together with tagging, you can put the WebSphere Portal Express Tag Cloud portlet on pages that contain social lists. This way your site visitors can reduce the contents of social lists by selecting individual tags from the tag cloud. The list of social objects is then restricted to content items that are tagged with the tag or tags that the user selected.

Configuring globally how social object data is served

Among other features, social objects have different links that enable users to download related data. For example, users can use links to download a file or the profile image of the author. You can globally configure how social object data is served to the users. To do so, you use a setting in the WP Connections Integration Service resource environment provider in the WebSphere Integrated Solutions Console.

About this task

Social object data can be served to the users in the following two ways:

- Directly from IBM Connections.
- By using the Ajax proxy of WebSphere Portal Express. Benefits include the possibility to serve all data to your users through IBM WebSphere Portal Express without direct web access from the web browser to Connections.

You have two methods to specify how social object data is served:

- You can globally configure how social object data is served by default. You configure the data serve method in the WP Connections Integration Service resource environment provider. This setting sets the default data serve method for all social lists in your portal. As a result, all your design components that do

not explicitly request a specific data serve mode use this default method for serving resources. This configuration method makes the data serve method switchable for all social lists.

- You can explicitly override the default data serve method in your individual design components. To override the default setting, you use specific *attributeName* values in the AttributeResource tags that you use in your design components. These values determine a specific data serve method. This way, you can specify for individual types of data by which method you want them to be served.

For example, this configuration affects the following attributes: authorImageLink, communityEntryLink, downloadLink, and entryLink. For details about which attributes are available for a specific type of social object, read *Rendering profiles for social lists*.

To globally configure how social object data is served by default, set the resource.serving.url.type property of the WP Connections Integration Service resource environment provider. You can then still choose between using the default method or overriding it for individual data in individual design components:

- To use the default method for serving resource, use the attribute name without any prefix. For example, such attribute names are authorImageLink, communityEntryLink, downloadLink, and entryLink values for the attributeName. The portal then generates links that point either to Connections or to the portal. This depends on the value that you specified for the resource.serving.url.type property.
- To override the default method, you prefix the corresponding attributeName values in the design component with either of the following prefixes. To determine whether you can apply the prefixed version, look at the profile details:
  - raw. Use this prefix to serve resources directly through Connections.
  - portal. Use this prefix to serve resources from the portal to the browser.

Example: You want to generate a file download link that always points directly to Connections, independently of the resource.serving.url.type setting. In this case, you add the [AttributeResource attributeName="*raw*DownloadLink"] tag to your design component instead of using the tag [AttributeResource attributeName="downloadLink"]. This option keeps the data serve method switchable based on the resource.resolution.url.type setting.

Specify either of the following two values for the resource.serving.url.type property:

portal

Specify this value to define that the authorImageLink, communityEntryLink, downloadLink, entryLink, and other link attributes contain URLs through which you can access the social objects by using the Ajax proxy of the portal. In this case, their values equal the values of the corresponding portalAuthorImageLink, portalCommunityEntryLink, portalDownloadLink, and portalEntryLink attributes for the design component. This value is the default setting.

connections

Specify this value to define that the authorImageLink, communityEntryLink, downloadLink, entryLink and other link attributes contain URLs by which you can access the social objects directly from Connections. In this case,

their values equal the values of the corresponding `rawAuthorImageLink`, `rawCommunityEntryLink`, `rawDownloadLink`, and `rawEntryLink` attributes for the design component.

To globally configure the method by which social object data is served, proceed by the following steps:

Procedure

1. Log in to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Select **WP ConnectionsIntegrationService**.
4. Under **Additional properties**, click **Custom properties**.
5. Set the `resource.serving.url.type` property to a value of either `connections` or `portal` as required.
6. Save your changes.
7. Restart your portal server for the changes to take effect.

Related reference:

"Digital Data Connector profiles for social rendering" on page 2147
Starting with Version 8.5, IBM WebSphere Portal Express includes a set of IBM Digital Data Connector (DDC) for WebSphere Portal Express profiles. You can use them with the social rendering DDC plug-in that is identified by the extension ID `ibm.portal.ddc.sr`.

Configuring globally how social object links are resolved

You can include one or many attributes of social objects in the design component that defines the visual design of your social list. Among other features, social objects have different resolvable links that enable users to open details views of the social objects or the community to which the social objects belong. If you plan to add these links to your social list, you can decide how you want the social objects and their home community to be resolved when users click the corresponding links. IBM WebSphere Portal Express can either resolve the links in the context of the portal itself, or redirect the user to the IBM Connections user interface. You can globally configure how these types of links of social objects are resolved for the users. To do so, you use a setting in the WP Connections Integration Service resource environment provider in the WebSphere Integrated Solutions Console.

About this task

Social object links can be resolved for the users in the following two ways:

- The linked resources can be rendered in the Connections user interfaces.
- The linked resources can be rendered on a portal page

You have two ways to specify how social object links are resolved:

- You can globally configure in the WP Connections Integration Service resource environment provider how social object links are resolved by default. This setting sets the link resolution method for all social lists in your portal. As a result, all your design components that do not explicitly request a specific resource link resolution method use this default method for resolving links. You can use this configuration method to switch the link resolution method globally for all social lists.
- You can explicitly override the default link resolution method in your individual design components. To override the default setting, you use specific *`attributeName`* values in the `AttributeResource` tags that you use in your design

components. These values determine a specific link resolution method. This way, you can specify for individual types of links by which method you want them to be resolved.

For example, this configuration affects the `link` and `communityLink` attributes. For details about which attributes are available for a specific type of social object, read *Rendering profiles for social lists*.

To globally configure how social object links are resolved by default, you set the `resource.resolution.url.type` property of the WP Connections Integration Service resource environment provider. You can then still choose between using the default method or overriding it for individual data in individual design components:

- To use the default method for resolving links, use the attribute names without any prefix. For example, such attribute names are `link` and `communityLink` values for the `attributeName`. The portal then generates links that resolve either to the Connections user interface or to a portal page. The link resolution method depends on the value that you specified for the `resource.resolution.url.type` property.
- To override the default method, you prefix the corresponding `attributeName` values in the design component with either of the following prefixes:
  - `raw`. Use this prefix to resolve links directly in Connections.
  - `portal`. Use this prefix to resolve links in the portal.

Example: You want to generate a social object link that always points directly to the Connections server, independently of the `resource.serving.url.type` setting. In this case, you add the `[AttributeResource attributeName="rawLink"]` tag to your design component instead of using the tag `[AttributeResource attributeName="link"]`. This option keeps the link resolution method switchable based on the `resource.resolution.url.type` setting.

You can specify one of the following values for the `resource.resolution.url.type` property:

contextual

This value is the default value. Specify this value to determine that the portal sets the social object resolving mode to either `portal` or `connections`. The portal determines the setting, depending on whether the Connections portlets "refresh" for WebSphere Portal Express are installed on your portal or not:

- If these portlets are installed, the portal uses the `portal` mode.
- If the portlets are not installed, the portal uses the `connections` mode.

Therefore, if you install these portlets while the social object resolving mode is set to `contextual`, the social object resolving mode switches from `connections` to `portal`.

portal

Specify this value to determine that the `link` and `communityLink` attributes contain URLs that display social objects in the context of the WebSphere Portal Express that renders the social list. In this case, their values equal the values of the corresponding `portalLink` and `portalCommunityLink` attributes for the design component. For details about the portal resolution mechanism, read *Social object resolution*. You can use this setting only if the Connections portlets are installed.

connections

Specify this value to determine that the `link` and `communityLink` attributes

contain URLs that route the user to Connections and resolve the links in the context of Connections. In this case, their values equal the values of the corresponding `rawLink` and `rawCommunityLink` attributes for the design component.

To globally configure how social object links are resolved by default, proceed by the following steps:

Procedure

1. Log in to the WebSphere Integrated Solutions Console.
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Select **WP ConnectionsIntegrationService**.
4. Under **Additional properties**, click **Custom properties**.
5. Set the `resource.resolution.url.type` property to a value of either `contextual`, `connections`, or `portal` as required.
6. Save your changes.
7. Restart your portal server for the changes to take effect.

What to do next

To determine the default link resolution method that is currently in effect in your social list designs, check the value of the portlet request attribute `ibm.portal.default.social.object.resolution.mode`. This attribute is set to either `connections` or `portal`. This value indicates the default social object resolution mode.

You can disable this request attribute by setting the property `enable.default.social.object.resolution.mode.request.param` to the value `false` in WP Configuration Service resource environment provider. In this case, the attribute `ibm.portal.default.social.object.resolution.mode` is always set to the value `disabled`.

Related concepts:

"Social object resolution" on page 2113
When a portal user clicks a link to an object, the portal takes the user to the details view of that object. This process is called social object resolution. For example, a user might click a specific forum topic that is listed in the Community Forum Topics list. In this case, the social object resolution takes the user to a portal page that provides a details view of the forum topic that the user clicked. You can influence the result of the resolution that the user views by setting various parameters. These parameters are described here.

Related reference:

"Digital Data Connector profiles for social rendering" on page 2147
Starting with Version 8.5, IBM WebSphere Portal Express includes a set of IBM Digital Data Connector (DDC) for WebSphere Portal Express profiles. You can use them with the social rendering DDC plug-in that is identified by the extension ID `ibm.portal.ddc.sr`.

Configuring file type icon mappings

Social rendering provides two types of list appearance components for result lists: `simple` and `comprehensive`. Both list appearances components show file type-specific icons when they render list entries that refer to individual files. In this case, the file type is determined based on the file extensions of the individual files. You can configure the set of file types that you want to use in the WP Connections

Integration Service resource environment provider. In the context of social lists, a file type is defined by a file type name and a list of file extensions. This list defines the mapping between individual files and your file types. In your social list appearance components, you can then access the file type name for a specific file by using the Web Content Manager `[AttributeResource attributeName="fileType"]` tag. That tag is defined in the IBM Digital Data Connector (DDC) for WebSphere Portal Express profile. You can then use the file type name to render the appropriate image, for example, by assigning a corresponding CSS class.

About this task

The mappings are defined in the WP Connections Integration Service resource environment provider in the WebSphere Integrated Solutions Console. To modify the mappings, proceed as follows:

Procedure

1. Log in to the WebSphere Integrated Solutions Console
2. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Click **WP ConnectionsIntegrationService**.
4. Under **Additional properties**, click **Custom properties**.
5. Edit the values for the appropriate `file.type.mapping.`*`type`* properties as required. The value for each property is a comma-separated list of file extensions. An example setting is `file.type.mapping.text = log, txt`. This mapping means that the Web Content Manager `[AttributeResource attributeName="fileType"]` tag returns the string `text` for all files with a `log` or `txt` file type extension.

What to do next

For your reference, the following topic lists the default file type mappings.

"File type mappings reference"
The social lists show specific icons for the different file and service types in the result lists. You can modify the file type mappings that are used for displaying these icons. For your reference, these mappings are listed here.

File type mappings reference:

The social lists show specific icons for the different file and service types in the result lists. You can modify the file type mappings that are used for displaying these icons. For your reference, these mappings are listed here.

audio .aac, .aif, .aifc, .aiff, .au, .kar, .m3u, .m4a, .mid, .midi, .mp3, .mpga, .ra, .ram, .rm, .rmi, .rpm, .snd, .wav, .wma

code .asp, .axs, .css, .htc, .htm, .html, .js, .jsp, .php, .sct, .stm, .wml, .xml, .xsl

compressed
.cab, .dmg, .gtar, .gz, .jar, .rar, .sit, .sqx, .tar, .taz, .tgz, .z, .zip

contact
.crd, .vcard, .vcf, .vcrd

data .123, .12m, .acs, .csv, .dbs, .dez, .ens, .fcd, .fcs, .mwkd, .mwks, .odf, .ods, .oss, .otf, .ots, .qad, .smd, .sms, .stc, .sxc, .sxm, .wg2, .wk4, .wk6, .xla,.xlam,.xlc,.xlm,.xls,.xlsb,.xlsm,.xlsx,.xlt,.xltm,.xltx,.xlw

flash .swf, .fla

graphic
.bmp, .cgm, .cmx, .cod, .djv, .djvu, .drw, .dxf, .eshr, .gif, .hgs, .ico, .ief, .img, .jfif, .jpe, .jpeg, .jpg, .odi, .oti, .pbm, .pcx, .pgl, .pgm, .pic, .pict, .png, .pnm, .pntg, .ppm, .psd, .psp6, .ras, .rgb, .sdw, .svg, .svgz, .tga, .tif, .tif6, .tiff, .wbmp, .wpg, .xbm, .xpm, .xwd

pdf .ai, .eps, .ich, .ich6, .iwp, .pdf, .ps

presentation
.flw, .key, .mass, .odp, .ope, .otc, .otp, .pot, .potm, .potx, .pp2, .pp97, .ppam, .pps, .ppsm, .ppsx, .ppt, .pptm, .pptx, .prz, .shw3, .sti, .sxi

text .323, .asc, .bas, .c, .etx, .h, .jw, .log, .mcw, .msg, .ow, .pfs, .qa, .sow, .text, .tsv, .tw, .txt, .uls, .vw3, .ws, .xy

video .asf, .asr, .asx, .avi, .divx, .flv, .lsf, .lsx, .mov, .movie, .mp2, .mp4, .mpa, .mpe, .mpeg, .mpg, .mpv2, .qt, .svi, .wmv

wordProcessing
.doc, .docm, .docx, .dot, .dotm, .dotx, .en4, .enw, .leg, .lwp, .lwp7, .manu, .msw, .mwp, .mwp2, .mwpf, .odc, .odg, .odt, .ort, .otg, .oth, .otm, .ott, .pages, .pcl, .rtf, .rtx, .std, .stw, .sxd, .sxq, .sxw, .w6, .w97, .wm, .wp5, .wp6, .wpf

The following types and services have no default mappings:

- dataDocs
- folder
- folderOpen
- link
- page
- presentationDocs
- visualization
- wordProcessingDocs

Administering social lists

You can administer several aspects of social lists. For example, you can add your own social lists to the content shelf, or you can tune the social object caches for performance.

About this task

Typically, an administrator does these administration tasks.

"Adding the social rendering theme module to a theme profile" on page 2132
For the social lists provided by social rendering to work, the theme that renders these lists needs to load CSS and JavaScript files. To include the CSS and JavaScript files in your theme, you need to add the `wp_social_rendering_85` theme module to your theme.

"Performance tuning for lists of social objects" on page 2133
Social data that is rendered on your portal pages by using the social rendering feature is retrieved from a remote IBM Connections server. To reduce network traffic and improve performance, the data is cached on IBM WebSphere Portal Express.

"Enabling social rendering in a virtual portal" on page 2133
Before you use social rendering in a virtual portal, you must deploy the new

web content library and templates. The version and fix pack of your IBM WebSphere Portal Express determines how you do so.

"Using Social Rendering with Tivoli Access Manager and WebSEAL" on page 2135
After you enable social rendering, your IBM WebSphere Portal Express acts as a client that sends HTTP requests to the IBM Connections server. If you use IBM Security Access Manager and WebSEAL to manage the access to your portal and Connections server, configure the portal so that it does not encode the WebSEAL session cookies.

CF06 "Removing the previous version of social rendering" on page 2135
You can remove the Social Lists 1.0 library and its configuration to remove it from the toolbar after you deploy the Social Lists 1.1 Library. Once you have removed the library, remove the drag and drop configuration with a separate command.

Adding the social rendering theme module to a theme profile

For the social lists provided by social rendering to work, the theme that renders these lists needs to load CSS and JavaScript files. To include the CSS and JavaScript files in your theme, you need to add the `wp_social_rendering_85` theme module to your theme.

About this task

The presentation components that the social rendering feature provides and that the social lists use rely on specific CSS styles and JavaScript files that the theme needs to load. To load those styles, you need to make sure that the `wp_social_rendering_85` theme module is available in the theme profile that is active on the page that renders the social list. If the `wp_social_rendering_85` theme module is not loaded for a page, the social list does not look as expected, and users cannot interact with the social lists on that page.

To determine which WebSphere Portal Express theme profiles already specify the `wp_social_rendering_85` theme module, review the theme profiles that come with the portal installation.

To add the theme module to your theme, you register the theme module by using the `add-theme-modules` portal configuration task. For more information, see the following examples.

- To add the social rendering theme module to a custom profile of a custom theme, run the task as follows:
  - For Linux :

```
./ConfigEngine.sh add-theme-modules
     -DThemeUniqueName=my.custom.theme
     -DThemeProfileFileName=profile_my_custom_profile.json
     -DModuleIDs=wp_social_rendering_85
```

  - For IBM i:

```
ConfigEngine.sh add-theme-modules
     -DThemeUniqueName=my.custom.theme
     -DThemeProfileFileName=profile_my_custom_profile.json
     -DModuleIDs=wp_social_rendering_85
```

  - For Windows:

```
ConfigEngine.bat add-theme-modules
     -DThemeUniqueName=my.custom.theme
     -DThemeProfileFileName=profile_my_custom_profile.json
     -DModuleIDs=wp_social_rendering_85
```

- To remove the social rendering theme module from the lightweight profile of the Portal 8.5 theme, run the remove task as follows:
  - For Linux :

```
./ConfigEngine.sh remove-theme-modules
     -DThemeUniqueName=my.custom.theme
     -DThemeProfileFileName=profile_my_custom_profile.json
     -DModuleIDs=wp_social_rendering_85
```

  - For IBM i:

```
ConfigEngine.sh remove-theme-modules
     -DThemeUniqueName=my.custom.theme
     -DThemeProfileFileName=profile_my_custom_profile.json
     -DModuleIDs=wp_social_rendering_85
```

  - For Windows:

```
ConfigEngine.bat remove-theme-modules
     -DThemeUniqueName=my.custom.theme
     -DThemeProfileFileName=profile_my_custom_profile.json
     -DModuleIDs=wp_social_rendering_85
```

Performance tuning for lists of social objects

Social data that is rendered on your portal pages by using the social rendering feature is retrieved from a remote IBM Connections server. To reduce network traffic and improve performance, the data is cached on IBM WebSphere Portal Express.

Social rendering uses the caching infrastructure of the IBM Digital Data Connector (DDC) for WebSphere Portal Express. For more information, read *Digital Data Connector cache tuning*.

Related reference:

"Digital Data Connector cache tuning" on page 3297
To improve performance, you can tune the caches for the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework.

Enabling social rendering in a virtual portal

Before you use social rendering in a virtual portal, you must deploy the new web content library and templates. The version and fix pack of your IBM WebSphere Portal Express determines how you do so.

Related tasks:

"Social Lists" on page 905
After you migrate from a previous WebSphere Portal Version to Version 8.5, you must run a configuration task. The configuration task is run to deploy the new web content library and templates before you can use the Social Lists features. If you already used the social rendering feature from 8.0.0.1, your existing web content libraries and all portlet clones that were created during the enablement on 8.0.0.1 is not changed.

Deploying social rendering in a virtual portal up to CF05:

In WebSphere Portal Express Version 8.5 with fix pack 05 or earlier, run the portal configuration task `action-enable-social-rendering`. Running this task deploys the Social Lists 1.0 library to the virtual portal. It also creates the required drag-and-drop configurations for the social rendering view definitions in the toolbar.

About this task

When you run this task on a virtual portal, identify the virtual portal by adding either one of the following parameters to the task. Each parameter requires the prefix `-D`.

`VirtualPortalHostName`
Specify the virtual portal context that identifies the virtual portal. For example, `vp.example.com`

`VirtualPortalContext`
Specify the virtual portal context that identifies the virtual portal. For example, `vp1`.

Procedure

1. Open a command prompt.
2. Change to the *wp_profile_root*`/ConfigEngine` directory.
3. Run the following command:
   - Linux : `./ConfigEngine.sh action-enable-social-rendering -DPortalAdminPwd=`*password* `-DWasPassword=`*password* `-DVirtualPortalContext=`*vp1*
   - IBM i: `ConfigEngine.sh action-enable-social-rendering -DPortalAdminPwd=`*password* `-DWasPassword=`*password* `-DVirtualPortalContext=`*vp1*
   - Windows: `ConfigEngine.bat action-enable-social-rendering -DPortalAdminPwd=`*password* `-DWasPassword=`*password* `-DVirtualPortalContext=`*vp1*

Deploying social rendering in a virtual portal with CF06 or later: CF06

WebSphere Portal Express Version 8.5 fix pack 06 introduces a new configuration task. In portal Version 8.5 fix pack 06 or later, use the configuration task **`deploy-social-rendering`** to enable social rendering in a virtual portal. You must also run this configuration task on the base portal to deploy the Social Lists 1.1 library.

About this task

To deploy the Social Lists 1.1 library on the base portal, proceed as follows:

Procedure

1. Open a command prompt.
2. Change to the *wp_profile_root*`/ConfigEngine` directory.
3. Run the following command:
   - Linux : `./ConfigEngine.sh deploy-social-rendering -DPortalAdminPwd=`*password* `-DWasPassword=`*password*
   - IBM i: `ConfigEngine.sh deploy-social-rendering -DPortalAdminPwd=`*password* `-DWasPassword=`*password*
   - Windows: `ConfigEngine.bat deploy-social-rendering -DPortalAdminPwd=`*password* `-DWasPassword=`*password*

To deploy the Social Lists 1.1 library on a virtual portal, proceed as follows:

4. Open a command prompt.
5. Change to the *wp_profile_root*`/ConfigEngine` directory.

6. Run the following command:
   - Linux : `./ConfigEngine.sh deploy-social-rendering -DPortalAdminPwd=password -DWasPassword=password -DVirtualPortalContext=vp1`
   - IBM i: `ConfigEngine.sh deploy-social-rendering -DPortalAdminPwd=password -DWasPassword=password -DVirtualPortalContext=vp1`
   - Windows: `ConfigEngine.bat deploy-social-rendering -DPortalAdminPwd=password -DWasPassword=password -DVirtualPortalContext=vp1`

Using Social Rendering with Tivoli Access Manager and WebSEAL

After you enable social rendering, your IBM WebSphere Portal Express acts as a client that sends HTTP requests to the IBM Connections server. If you use IBM Security Access Manager and WebSEAL to manage the access to your portal and Connections server, configure the portal so that it does not encode the WebSEAL session cookies.

About this task

If you do not configure your portal as described here, the social rendering portlets do not show any results. Examples of such portlets are List of Blog Posts, List of Files, List of Forum Topics.

Procedure

1. On a stand-alone portal, log in to the WebSphere Integrated Solutions Console. On a portal cluster, log in to the WebSphere Integrated Solutions Console on the Deployment Manager.
2. Go to **Resources** > **Resource Environment** > **Resource Environment Providers.**.
3. Select **WP ConfigService**.
4. Under the Additional Properties heading, select **Custom Properties**.
5. Create or edit the property `cookie.escaping.ignore` and specify the value according to your WebSEAL configuration. The value of this property is a regular expression. It must match the name of the cookie that you do NOT want to be escaped. For example, if the WebSEAL session cookies have the names `PD-H-SESSION-ID` and `PD-S-SESSION-ID`, specify the value `PD-H-SESSION-ID|PD-S-SESSION-ID`.
6. For the changes to take effect, restart the portal server.

Results

You can now use the social rendering portlets.

Related tasks:

"Security Access Manager" on page 1626
IBM WebSphere Portal Express supports the use of IBM Security Access Manager. Existing Security Access Manager users can use the Security Access Manager services to assist them in their deployment.

Removing the previous version of social rendering

You can remove the Social Lists 1.0 library and its configuration to remove it from the toolbar after you deploy the Social Lists 1.1 Library. Once you have removed the library, remove the drag and drop configuration with a separate command.

About this task

The difference between the deprecated Social Lists 1.0 library and the Social lists 1.1 library is the use of OneUI styles. The latest versions of the social rendering uses the Social Lists 1.1 library and does not depend on the OneUI styles.

Procedure

1. In the Web Content Manager Authoring portlet, click **Social Lists 1.0** > **Content** > **Social Content site area**.
2. Click **Show hidden fields**.
3. Select the **Properties** tab.
4. In the Keywords text field, remove the text `ibm.portal.toolbar.NewContent`.
5. Click **Save**.
6. Remove references in your applications to Social Lists 1.0 library and its related configuration artifacts.
7. Change to the *wp_profile_root*`/ConfigEngine` directory.
8. Run the following command.
   - Linux : `./ConfigEngine.sh remove-social-rendering-library -DPortalAdminPwd=`*`password`* `-DWasPassword=`*`password`*
   - IBM i: `ConfigEngine.sh remove-social-rendering-library -DPortalAdminPwd=`*`password`* `-DWasPassword=`*`password`*
   - Windows: `ConfigEngine.bat remove-social-rendering-library -DPortalAdminPwd=`*`password`* `-DWasPassword=`*`password`*
9. Examine the log output with the console, or the ConfigTrace.log file. If the **`remove-social-rendering-library`** task completed successfully, continue to the next step and remove the configuration that removes the drag and drop configurations for social rendering view definitions in the toolbar..
10. Run the following command.
    - Linux : `./ConfigEngine.sh remove-social-rendering-config -DPortalAdminPwd=`*`password`* `-DWasPassword=`*`password`*
    - IBM i: `ConfigEngine.sh remove-social-rendering-config -DPortalAdminPwd=`*`password`* `-DWasPassword=`*`password`*
    - Windows: `ConfigEngine.bat remove-social-rendering-config -DPortalAdminPwd=`*`password`* `-DWasPassword=`*`password`*

Related concepts:

"Static resources" on page 2522
Static resources include the markup that is defined by `.html`, `.css`, and `.js` files that are used by the theme. Some `.json` files are used to define menu options, module definitions, and module profiles.

Customizing view definitions for portal site visitors

You can customize social rendering view definitions in a number of ways. For example, you can determine the social data that is shown and the visual appearance of the data. You customize social rendering by working with the social list definition authoring template. For custom view definitions, you can also create your own authoring templates.

About this task

Typically, a website designer does these customizing tasks.

Note: If you want to customize the web content that the sample web content library named Social Lists 1.0 provides, create a copy of that library and customize that copy.

"Customizing social list definitions by using inline editing"
You can customize your list view definitions by defining the following settings in inline editing mode. List view definitions are represented by content items of the social list definition authoring template. This customization is normally done by a web designer or a page editor.

"Customizing forum topic details view definitions by using inline editing" on page 2142
You can customize your forum topics details view definitions by defining the following settings in inline editing mode. Forum topic details are represented by content items of the forum topic details authoring template. This customization is normally done by a web designer or a page editor.

"Request attributes for changing search queries dynamically" on page 2143
The definitions of your list views are represented by content items of the Social List Definition authoring template. You can override specific elements of such list view definitions by using the request attributes that are described here. To work with request attributes in your web content, use the `RequestAttribute` rendering plug-in. Changing a list view definition by using request attributes is a way to modify the content of your lists dynamically, for example based on user input.

"Customizing the visual design of your view definitions" on page 2146
You can customize the visual appearance of the markup that your social rendering view definitions generate. The markup generation is controlled by the appearance component that your individual list view definition content items reference.

"Creating custom authoring templates for list definitions" on page 2230
Social rendering provides you a set of view definitions that you can use to add social data to your portal pages. These list view definition content items are created from the social list definition IBM Web Content Manager authoring template. You can create your own custom list view definitions by creating new content items from this authoring template. To customize social rendering even further, you can also create your own authoring templates to extend the data set that makes up your list view definitions.

"Implementing interactions with social objects" on page 2231
Site designers can implement design components that suppor interactions between the user and the social data. For example, a user can post a new reply to a forum topic or delete a previous reply.

Customizing social list definitions by using inline editing

You can customize your list view definitions by defining the following settings in inline editing mode. List view definitions are represented by content items of the social list definition authoring template. This customization is normally done by a web designer or a page editor.

About this task

To define these settings, you need the access permissions for editing the web content item that represents the view definition. You do not need edit access permissions to the **Web Content Viewer** portlet that shows the view definition.

Note: The list view definitions that are provided with social rendering use the Connections search service feed to retrieve the social objects displayed in the list. This has the following consequences:

- Only information that is available in the Connections search feed can be displayed in the social lists.
- Updates to social objects in Connections do not appear in the social lists provided with social rendering until the index of the Connections search service was updated.

For more details about how to administer the Connections search service, read the information about *Administering Search* in the Connections product documentation.

The following settings and options are available for customizing social list view definitions by using inline editing:

Name Use this setting to specify a name for the list of social items. The name must consist of at least one alphanumeric character (`a-z, A-Z,0-9`). It can include spaces and the following special characters: `$ - _ . ! ( ) ,`

Display Title
Use this setting to specify the display title for the list of social objects. If you use the default presentation template, a change to the display title also changes the heading shown for the list of social objects on the page. By alternative, you can specify **Localizations** instead of a display title. In this case, the portal ignores the default display title and uses the display title of the respective language instead. Social rendering provides a default text provider named Social Rendering. It includes a translated default display title for each language that the portal supports. To display the translated default titles, proceed as follows:

1. Click **Localizations**.
2. Select the Text Provider **Social Rendering**.
3. In the **Text Provider Key** field, type the key `SR_CONTENT_FILES`.
4. To save your updates, click **Save and Close**.

Content Sources
Use this setting to select the content sources that you want to include in the list of social objects. You can select multiple content sources. Each content source corresponds to a IBM Connections service. Social rendering supports the following Connections services: activities, blogs, bookmarks, communities, files, libraries, forums, profiles, wikis, and events.

Note: Outside social rendering, some of the Connections services serve multiple social objects. For example, the **Forums** service serves objects of type **Forum topic** and **Reply**. Social lists serve only the object types that are listed in the following table.

Table 338. Content sources, the corresponding Connections services, and the content types that they serve in social lists

Content source	Connections service	Content type that the content source serves in social lists
Activities	Activities	Activity
Blogs	Blogs	Blog post
Bookmarks	Bookmarks	Bookmark
Communities	Communities	Community
Events	Events	Event
Files	Files, Libraries	File

Table 338. Content sources, the corresponding Connections services, and the content types that they serve in social lists (continued)

Content source	Connections service	Content type that the content source serves in social lists
Forums	Forums	Forum topic
Profiles	Profiles	Profile
Wikis	Wikis	Wiki page

Filter by Community
You can limit the contents of a social list to a community by filtering. The following options are available:

Limit to community that is associated with this page
If you select this option, the list contains only social objects from the community that is associated with the current page.

Limit to selected community
You can use this option to select a community. If you select a community, the list contains only social objects from the selected community.

Do not limit to a community
If you select this option, the list is not limited to social objects that belong to communities. Social objects are listed, independent of whether they belong to a community or not.

For more information, read the WebSphere Portal Express product documentation under the topic about *Managing community associations*.

Note: User profiles are not considered to be content of a community. Therefore, if you activate community filtering, the resulting lists do not contain any user profile entries.

Filter by Type of Access
Select this option to display only public objects in this list. If you do not select this option, the list can also include social objects that are owned by or shared with the current user who views the list.

Filter by Search Term
Use this setting to specify the text for which you want the portal to search when it retrieves the social objects for the list from Connections. This field corresponds to the `query` parameter of the Connections Search API. For more information about the supported values and operators, read the Search service documentation for the Connections server.

Filter by Tags
Use this setting to specify a comma-separated list of tags to adjust the list of social objects. If you specify a tag filter here, the list displays only social objects that are tagged with these specific terms. If the same tags are deselected from a tag cloud portlet, this filter still applies as it is specified by the author, not by the end user. For more information, read *Using the portal Tag Cloud with lists of social objects*.

Tag Selection Support
Select this option to make this social list reflect tag selection information

that is generated by the portal Tag Cloud portlet. If a user selects a tag in the portal Tag Cloud, the portal adds this tag dynamically to the existing tag filter of the current social list.

Dynamic Filtering
Use this setting to select the dynamic contexts for filtering the list. The following options are available:

Enable dynamic search term filtering
If you select this option, the list shows the social objects for a search query based on the public render parameter `filters` with the value `sr:searchterm:`*`searchterm`*, where *`searchterm`* is the search term that the user specified. If a dynamic search term is available, it overrides the `Filter by Search Term` setting. For more information about the public rendering parameter `filters`, read *Dynamic list-rendering contexts*.

Enable dynamic author filtering
If you select this option, the list is filtered based on the selected IBM Connections user profile. The list shows the social objects of which the author is the user represented by the selected user profile. Selecting a user profile means that the user clicks a user profile link from a list that is resolved by the connections POC resolver in the portal. For more information about the social object resolution, read *Specifying in the design component how social object links are resolved*.

Sorting Criteria
Use this setting to specify the sorting criteria that are used for the list of social objects. You can choose between the **Last updated** and **Relevance** options.

Sorting Order
Use this setting to specify the sorting order that is used for the list of social objects. You can choose between **Ascending** and **Descending** order.

List Appearance
Use this setting to select a component that formats the list of social objects. You can select only IBM Web Content Manager Personalization components that have the keyword `ibm.portal.socialrendering` assigned in their profile section. To enable the profile section for IBM Web Content Manager components, you must set the `control.Cmpnt=com.aptrix.pluto.taxonomy.ProfileControl` property in the `WCMConfigService.properties` file and restart your portal. For more information about this type of Web Content Manager Personalization components, see *Customizing the visual design of social lists*.

By default, you can choose between two appearance types:

Simple
The Simple appearance is useful for a mobile view, as it contains fewer data.

Comprehensive
The Comprehensive appearance provides more information of the following types:

- The person who made the last update to the social object
- The date of the last update to the social object
- Tags that are assigned to the social object

- The summary of the social object.

Both of these default appearances are responsive to the theme columns, screen width, and screen orientation. Social rendering uses default CSS styles to provide this responsiveness. For example, they can be useful if users use different types of devices to view the lists or change between portrait and landscape views on a mobile device.

Maximum Results (hidden)

This field is an optional input field. This setting is hidden by default. To make it show, click **Show hidden fields** at the beginning or end of the screen. Use this setting to specify the maximum number of content items included in this list. This figure determines the maximum number of social objects that Connections returns. Specify a positive integer. It is good practice to match this number with the number of results that the Personalization component can display that you specify at the List Appearance element. To determine that number, multiply the **Results per page** by the **Maximum pages to include** of the Personalization component.

You can also configure this setting globally for all social list portlets in the WP Connections Integration Service in the WebSphere Integrated Solutions Console. For more information, see *Configuring the maximum number of items loaded from Connections*. If you leave the **Maximum Results** field empty for a social list, then the value that is set in the WP Connections Integration Service applies as the default.

Custom Properties (Hidden)

This field is an optional input field. It is hidden by default. To make it show, click **Show hidden fields** at the beginning or end of the screen. Add one or more custom properties in this input field. You can use such more custom properties to add more flexibility to the layout of your list as required. For example, you can change the layout of your social list by defining a specific condition that uses the custom property added in this field. The following code sample shows how you change the color of the social list heading by specifying the string `blue` at the `Custom Properties` element:

```
[Plugin:Equals text1="blue" text2="[Element context='current' type='content' key='custom']"]
   <h2 style="color:blue;">[Property context="current" type="content" field="title"]</h2>
[/Plugin:Equals]
[Plugin:NotEquals text1="blue" text2="[Element context='current' type='content' key='custom']"]
   <h2 style="color:black;">[Property context="current" type="content" field="title"]</h2>
[/Plugin:NotEquals]
```

Custom Link Resolution Root Page (Hidden)

You can use this field to control the social object resolution behavior for links that users click in this list. For more detailed information, read *Social Object Resolution*.

Related concepts:

"Social object resolution" on page 2113
When a portal user clicks a link to an object, the portal takes the user to the details view of that object. This process is called social object resolution. For example, a user might click a specific forum topic that is listed in the Community Forum Topics list. In this case, the social object resolution takes the user to a portal page that provides a details view of the forum topic that the user clicked. You can influence the result of the resolution that the user views by setting various parameters. These parameters are described here.

Related tasks:

"Using the portal Tag Cloud with lists of social objects" on page 2118
To get the most benefit from social rendering together with tagging, you can put the WebSphere Portal Express Tag Cloud portlet on pages that contain social lists. This way your site visitors can reduce the contents of social lists by selecting individual tags from the tag cloud. The list of social objects is then restricted to content items that are tagged with the tag or tags that the user selected.

"Customizing the visual design of your view definitions" on page 2146
You can customize the visual appearance of the markup that your social rendering view definitions generate. The markup generation is controlled by the appearance component that your individual list view definition content items reference.

"Managing community associations" on page 1290
You can create, view, modify, or delete community associations on a page with the Page Associations window, the XML configuration interface, and Portal Scripting Interface.

"Configuring the maximum number of items loaded from Connections" on page 2123
You can define a value for the maximum number of social objects that you want the IBM Connections to return when data for a list of social objects is requested.

Related information:

Dynamic list-rendering contexts


Administering Search

Customizing forum topic details view definitions by using inline editing

You can customize your forum topics details view definitions by defining the following settings in inline editing mode. Forum topic details are represented by content items of the forum topic details authoring template. This customization is normally done by a web designer or a page editor.

About this task

To define these settings, you need the access permissions for editing the web content item that represents the view definition. You do not need edit access permissions to the **Web Content Viewer** portlet that shows the view definition.

The following settings and options are available for customizing social list view definitions by using inline editing:

Name Use this setting to specify a name for this item. The name must consist of at least one alphanumeric character (`a-z, A-Z,0-9`). It can include spaces and the following special characters: `$ - _ . ! ( ) ,`

Display Title
Use this setting to specify the display title for this item.

Details Appearance
Use this setting to select the appearance component that generates the details view of this item. You can select only IBM Web Content Manager Personalization components that have the keyword `ibm.portal.socialrendering` assigned in their profile section. To enable the profile section for Web Content Manager components, you must set the `control.Cmpnt=com.aptrix.pluto.taxonomy.ProfileControl` property in the `WCMConfigService.properties` file and restart your portal. For more information about this type of Web Content Manager Personalization components, read *Customizing the visual design of social lists*. By default,

there is only one appearance component for forum topic details view definitions. It is named Forum Topic Details.

Index Page
Use this setting to control the target for the **Show Index Page** link that is shown by default by the forum topic details appearance component. Specify one of the following values:

parent With this setting, the **Show Index Page** link points to the parent page of the current page.

unique_name_of_a_portal_page
With this setting, the **Show Index Page** link points to the portal page with the unique name that you specify.

Maximum Results (hidden)
This field is an optional input field. The default value is `200`. This setting is hidden by default. To make it show, click **Show hidden fields** at the beginning or end of the screen. Use this setting to specify the maximum number of replies that you want to be loaded for the given forum topic. Specify a positive integer. It is good practice to match this number with the number with the **Results per page** value thst you configure in the appearance component that you use with this forum topic details view definition.

Initial List Size (hidden)
This field is an optional input field. The default value is `10`. This setting is hidden by default. To make it show, click **Show hidden fields** at the beginning or end of the screen. Use this setting to specify the maximum number of replies that you want to show in the initial rendering of the forum topic details. The forum topic details appearance component supports a **View more replies** link that shows all replies that have been loaded.

Request attributes for changing search queries dynamically

The definitions of your list views are represented by content items of the Social List Definition authoring template. You can override specific elements of such list view definitions by using the request attributes that are described here. To work with request attributes in your web content, use the `RequestAttribute` rendering plug-in. Changing a list view definition by using request attributes is a way to modify the content of your lists dynamically, for example based on user input.

You can use the following request attributes:

ibm.portal.sr.search.access
Use this request attribute to dynamically set the type of access that you want to use for filtering the list. This request attribute overrides the Filter by Type of Access setting of your list view definition. You can use the following values:

public Use this value to include only public objects in the list.

private
Use this value to include public objects and private objects to which the user has access. The value `private` corresponds to the value `personalOnly` of the IBM Connections Scopes API. This value `personalOnly` is described in the scopes feed summary element as `Search for your content`. This scope matches public documents

and documents that are private and to which the user has access based on the user ID, group IDs, communities to which the user belongs.

shared
Use this value to include only private objects to which the user has access. The value `shared` corresponds to the value `personalOnlyByACL` of the Connections Scopes API. This value `personalOnlyByACL` is described in the scopes feed summary element as `Search for your private content`. This scope matches only documents to which the user has access based on the user ID, group IDs, communities to which the user belongs. This scope includes only documents that the user owns or that were explicitly or implicitly shared with the user.

ibm.portal.sr.search.authorid
Use this request attribute to dynamically set the external ID (`extID`) of the user that you want to use for filtering the list. The external ID is specific to Connections. You can retrieve the `extID` that belongs to the user of a selected Connections user profile by using the `ConnectionsContext` rendering plug-in. With regards to author filtering, this request attribute overrides the Dynamic Filtering setting of your list view definition.

ibm.portal.sr.search.community
Use this request attribute to dynamically set the UUID of a specific Connections community. If you specify a community, the list contains only social objects from that community. This request attribute overrides the Filter by Community setting of your list view definition.

ibm.portal.sr.search.max
Use this request attribute to dynamically set the maximum number of items that you want to include in the list. This request attribute overrides the Maximum Results setting of your list view definition.

ibm.portal.sr.search.queryURI
Use this request attribute to override the URI used to query the Connections server. Normally, the portal generates the query URI based on the list view definition and the request attributes described here. By using this request attribute, you can provide a URI that replaces the one computed by the portal. You can use the following placeholders in a custom query URI:

{$search.public}
If your custom query URI starts with this string, the portal replaces it with the HTTP link of the Search service of the Connections server.

{$search.protected}
If your custom query URI starts with this string, the portal replaces it with the HTTPS link of the Search service of the Connections server.

ibm.portal.sr.search.searchterm
Use this request attribute to dynamically set the search term that you want to use for filtering the list. With regards to search term filtering, this request attribute overrides the Filter by Search Term setting and the Dynamic Filtering setting of your list view definition.

ibm.portal.sr.search.sortby
Use this request attribute to dynamically set the criteria that you want to

use for sorting the list. Supported values depend on the Search service of your Connections server. Usually, the following values are supported:

date Use this value to sort your list by the last modified date of the list entries.

relevance
Use this value to sort your list by the relevance of the list entries.

In addition to these two values, you can also specify the name of any other sortable field of Connections objects, for example the `title`. This request attribute overrides the Sorting Criteria setting of your list view definition.

ibm.portal.sr.search.sortdir
Use this request attribute to dynamically set the order by which you want to sort the list. You can use the following values:

asc Use this value to sort your list in ascending order.

desc Use this value to sort your list in descending order.

This request attribute overrides the Sorting Order setting of your list view definition.

ibm.portal.sr.search.source
Use this request attribute to dynamically set the type of content source that you want to use for filtering the list. This request attribute overrides the Filter by Content Source setting of your list view definition. You can use the following values:

allconnections
Use this value to get results from all content sources. This value is the default value.

activities
Use this value to get results from activities only.

blogs Use this value to get results from blogs only.

communities
Use this value to get results from communities only.

dogear
Use this value to get results from dogears and bookmarks only.

events Use this value to get results from events only.

forums
Use this value to get results from forums only.

profiles
Use this value to get results from profiles.

wikis Use this value to get results from wikis only.

files Use this value to get results from files only.

To combine different content source types for your search, provide multiple values and separate them by commas. Example:
`ibm.portal.sr.search.source=blogs,forums,files`.

ibm.portal.sr.search.tags
Use this request attribute to dynamically set the tags that you want to use for filtering the list. To specify more than one tag as a filter, use blanks or

commas as separator characters between the tags. This request attribute overrides the Filter by Tags setting of your list view definition.

Customizing the visual design of your view definitions

You can customize the visual appearance of the markup that your social rendering view definitions generate. The markup generation is controlled by the appearance component that your individual list view definition content items reference.

"Customizing the markup generation"
You create the visual designs for your social rendering view definitions by using social rendering appearance components. Social rendering appearance components are implemented as IBM Web Content Manager Personalization components. Personalization components that you use with social rendering must wrap a IBM Digital Data Connector (DDC) for WebSphere Portal Express selection rule.

"Using the business card" on page 2223
You can integrate the business card and online status in a list of social objects by using live text micro format.

"Customizing the CSS styles of social lists" on page 2225
The lists of social objects rely on the availability of several CSS class definitions in the `wp_social_rendering_85` theme module.

Related concepts:

"Personalization element" on page 1820
A personalization element stores a reference to a personalization rule or content spot that is generated by Portal Personalization. To use a personalization element, you must create a personalization component.

Customizing the markup generation:

You create the visual designs for your social rendering view definitions by using social rendering appearance components. Social rendering appearance components are implemented as IBM Web Content Manager Personalization components. Personalization components that you use with social rendering must wrap a IBM Digital Data Connector (DDC) for WebSphere Portal Express selection rule.

About this task

In general, a Web Content Manager Personalization component combines a portal Personalization selection rule with formatting information. This information defines how data returned by that selection rule is transformed into markup that is rendered on a portal page. Social rendering delegates the data selection logic to the underlying DDC framework and uses only the markup generation capabilities of Web Content Manager Personalization components. To trigger the loading of the data, social rendering DDC selection rules that work on the Pluggable Resources resource collection. For more information, read *Technical concepts*.

For extracting the individual pieces of metadata of social objects contained in a social list, you can use the Web Content Manager `AttributeResource` tag. For more information about Web Content Manager Personalization components, see the topic about *Personalization element* in the Web Content Manager product documentation.

"Digital Data Connector profiles for social rendering" on page 2147
Starting with Version 8.5, IBM WebSphere Portal Express includes a set of IBM Digital Data Connector (DDC) for WebSphere Portal Express profiles. You can use them with the social rendering DDC plug-in that is identified by the extension ID `ibm.portal.ddc.sr`.

Related concepts:

"Technical concepts" on page 3257
Before you use the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework, you can familiarize yourself with its main technical concepts and building blocks.

Digital Data Connector profiles for social rendering:

Starting with Version 8.5, IBM WebSphere Portal Express includes a set of IBM Digital Data Connector (DDC) for WebSphere Portal Express profiles. You can use them with the social rendering DDC plug-in that is identified by the extension ID `ibm.portal.ddc.sr`.

You can use these profiles to define social lists and detail views for the following types of social data that is served by the IBM Connections server:

- Blogs, blog posts, and blog post comments, as served by the Blogs service:
  - Blogs (`ibm.portal.sr.blogs`)
  - Blog details (`ibm.portal.sr.blogs.details`)
  - Blog posts (`ibm.portal.sr.blogs.posts`)
  - Blog post comments (`ibm.portal.sr.blogs.post.comments`)
- Communities and community members as served by the Communities service:
  - Communities (`ibm.portal.sr.communities`)
  - Community members (`ibm.portal.sr.communities.members`)
- Forums, forum topics, forum topic replies as served by the Forums service:
  - Forums (`ibm.portal.sr.forums`)
  - Forum topics (`ibm.portal.sr.forums.topics`)
  - Forum topic replies (`ibm.portal.sr.forums.replies`)
  - Forum topic and reply attachments (`ibm.portal.sr.forums.attachments`)
- User profiles, user network connections as served by the Profiles service:
  - Profiles (`ibm.portal.sr.profiles`)
  - Profiles connections (`ibm.portal.sr.profiles.connections`)
- Wiki pages, as served by the Wikis service:
  - Wiki pages (`ibm.portal.sr.wikis.pages`)
  - Wiki page comments (`ibm.portal.sr.wikis.page.comments`)
  - Wiki page attachments (`ibm.portal.sr.wikis.attachments`)
- Search results as served by the Connections search service:
  - Social objects (`ibm.portal.sr.search`)
- Tags that are served with the search results of the Connections search service: (`ibm.portal.sr.tags`).
- Common profile: (`ibm.portal.sr.common`).

The profiles listed so far partly inherit attribute definitions from a set of profiles that are available for inheritance purposes only. You cannot see these profiles in the tag helper because they are used only by other profiles that inherit attribute definitions from them and thereby reuse them. Note that you might need to selectively overwrite parts of these profiles if the feed that you intend to consume does not serve data as expected by the profiles listed in the following:

- Common profile: (`ibm.portal.sr.common`)
- Modifier support profile: (`ibm.portal.sr.modifiersupport`)
- Tag support profile: (`ibm.portal.sr.shared.tagsupport`)

- OpenSearch support profile: (`ibm.portal.sr.shared.opensearchsupport`).

All list view definitions that WebSphere Portal Express provides use the Social Objects 1.0 profile. This profile works against the search services. for more information, read *The social objects profile*.

The Forum Topic Details view definition is uses the Forum Topics profile. For more information, read *Forums, forum topics, and forum topic replies profiles*.

The profiles define the attribute names that you can use in your `[AttributeResource]` tags when you generate the HTML markup for your bean lists. You can either manually add the `AttributeResource` tags to the design component of a social list, or you can use the **Insert a Tag** user interface of IBM Web Content Manager. Example: If a profile defines an attribute that is named `title`, you can write out the title value of the items in your list by using the following tag: `[AttributeResource attributeName="title"]`.

Note: CF03 When you integrate an Connections server that runs in the Smart Cloud for Social Business, the following item attributes always return empty string values: `authorObjectID`, `memberObjectID`, and `modifierObjectID`.

The profiles that WebSphere Portal Express includes are listed in the following topics.

"Blog-related profiles" on page 2150
These profiles provide access to IBM Connections blog-related feed data for blogs, blogs details, blog posts, and blog post comments.

"Communities and community members profiles" on page 2164
These profiles provide access to IBM Connections community and community members feed data.

"Forum-related profiles" on page 2172
These profiles provide access to IBM Connections forum-related feed data for forums, forum topics, and forum topic replies profiles.

"Profiles and profiles connections profiles" on page 2189
These profiles provide access to IBM Connections profile related feed data.

"Social objects profile" on page 2196
The social objects profile provides access to IBM Connections social objects feed data.

"Tags profile" on page 2205
The tags profile provides access to the IBM Connections tags facet that is served by the connections search service.

"Wiki-related profiles" on page 2205
These profiles provide access to wiki-related feed data for wiki pages, wiki page comments, and wiki page attachments.

"Common profile" on page 2218
The common profile provides general aspects that are available in various feeds that are served by IBM Connections. You can use the common profile when you create your own custom profiles.

"Modifier support profile" on page 2221
The modifier support profile provides general modifier aspects. These modifier aspects are available in various feeds that are served by IBM Connections. You can reuse them when you create your own profiles.

"Tag support profile" on page 2222
The tag support profile provides access to a list of tags that is associated with the social objects served by IBM Connections feed data.

"OpenSearch support profile" on page 2223
The OpenSearch support profile provides access to information that is defined in the OpenSearch specification. The OpenSearch specification is commonly used in the Atom feed format.

Related concepts:

"Web content tags" on page 1833
You use tags to reference elements within presentation templates and element designs.

Related tasks:

"Configuring file type icon mappings" on page 2129
Social rendering provides two types of list appearance components for result lists: `simple` and `comprehensive`. Both list appearances components show file type-specific icons when they render list entries that refer to individual files. In this case, the file type is determined based on the file extensions of the individual files. You can configure the set of file types that you want to use in the WP Connections Integration Service resource environment provider. In the context of social lists, a file type is defined by a file type name and a list of file extensions. This list defines the mapping between individual files and your file types. In your social list appearance components, you can then access the file type name for a specific file by using the Web Content Manager `[AttributeResource attributeName="fileType"]` tag. That tag is defined in the IBM Digital Data Connector (DDC) for WebSphere Portal Express profile. You can then use the file type name to render the appropriate image, for example, by assigning a corresponding CSS class.

Integrating with IBM Connections
Integrating with IBM Connections is a multiple step process. Review the high-level steps before you begin.

"Configuring globally how social object data is served" on page 2125
Among other features, social objects have different links that enable users to download related data. For example, users can use links to download a file or the profile image of the author. You can globally configure how social object data is served to the users. To do so, you use a setting in the WP Connections Integration Service resource environment provider in the WebSphere Integrated Solutions Console.

"Configuring globally how social object links are resolved" on page 2127
You can include one or many attributes of social objects in the design component that defines the visual design of your social list. Among other features, social objects have different resolvable links that enable users to open details views of the social objects or the community to which the social objects belong. If you plan to add these links to your social list, you can decide how you want the social objects and their home community to be resolved when users click the corresponding links. IBM WebSphere Portal Express can either resolve the links in the context of the portal itself, or redirect the user to the IBM Connections user interface. You can globally configure how these types of links of social objects are resolved for the users. To do so, you use a setting in the WP Connections Integration Service resource environment provider in the WebSphere Integrated Solutions Console.

Related reference:

Setting parameters to format dates
These parameters are used to set the format of dates.

Blog-related profiles:

These profiles provide access to IBM Connections blog-related feed data for blogs, blogs details, blog posts, and blog post comments.

Blogs profile

The blogs profile provides access to Connections blogs feed data. It declares the following attribute names:

Author's email address
`authorEmail`
: This attribute references the email address of the author of the social object.

Author's ID
`authorID`
: This attribute references the internal ID of the author of the social object. It represents the author ID from the Connections server where the social object is stored.

Author's image URL
`authorImageLink`
: This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL where you can download the image either directly from Connections or by using the Ajax proxy of the portal. For more information about this attribute, read *Configuring globally how social object data is served*. This attribute is computed by the social rendering IBM Digital Data Connector (DDC) for WebSphere Portal Express plug-in.

Author's name
`authorName`
: You can use this attribute to include the name of the author of the social object in the design of a social list.

Author's object ID
`authorObjectID`
: This attribute references the serialized object ID of the author of the social object. In contrast to the `authorID` attribute, the `authorObjectID` attribute represents an ID that is used by WebSphere Portal Express rather than by Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's UID
`authorUID`
: This attribute specifies the value of the UID attribute of the author of the social object. This value reflects the UID user attribute as defined by the user repository of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Body HTML
`body HTML`
: The HTML content of this item.

Body plain
`body plain`
: The plain content of this item.

Body content type
bodyContentType
This attribute specifies the content type identifier for the body attribute of this item.

Can create blog post
canCreateBlogPost
If the current user has permission to create posts for this blog, this attribute returns the string `true`. Otherwise, it returns `false`. This attribute is computed by the social rendering Digital Data Connector plug-in.

Community UUID
communityUUID
If the current blog is a community blog, this attribute specifies the UUID of the community that contains the blog. If the current blog is not contained in a community, this attribute is empty.

Feed entry
entry The original feed entry of this item.

Handle
handle This attribute represents the handle of this blog.

ID
id The unique identifier for this item.

Is closed
isClosed
If the current blog is in state `closed`, this attribute returns the string `true`. Otherwise, it returns `false`.

Is commenting graduated ideas enabled
isCommentingGraduatedIdeasEnabled
If the Commenting Graduated Ideas feature is enabled, this attribute returns the string `true`. Otherwise, it returns `false`.

Is community blog
isCommunityBlog
If the current blog is a community blog, this attribute returns the string `true`. Otherwise, it returns `false`.

Is frozen
isFrozen
If the current blog is in state `frozen`, this attribute returns the string `true`. Otherwise, it returns `false`.

Is ideation blog
isIdeationBlog
If the current blog is an ideation blog, this attribute returns the string `true`. Otherwise, it returns `false`.

Is moderated
isModerated
If the moderation feature is enabled for the current blog, this attribute returns the string `true`. Otherwise, it returns `false`.

Is open
isOpen If the current blog is in state `open`, this attribute returns the string `true`. Otherwise, it returns `false`.

Is private
isPrivate
If the current blog is contained in a private community, this attribute returns the string `true`. Otherwise, it returns `false`.

Is public
isPublic
If the current blog is contained in a public community, this attribute returns the string `true`. Otherwise, it returns `false`.

Is restricted
isRestricted
If the current blog is contained in a restricted community, this attribute returns the string `true`. Otherwise, it returns `false`.

Is Vote Limit enabled
isVoteLimitEnabled
If the Vote Limit feature is enabled, this attribute returns the string `true`. Otherwise, it returns `false`.

Is Voting Graduated Ideas enabled
isVotingGraduatedIdeasEnabled
If the Voting Graduated Ideas feature is enabled, this attribute returns the string `true`. Otherwise, it returns `false`.

View URL
link This attribute specifies the URL that points to the details view of this blog. This attribute is computed by the social rendering Digital Data Connector plug-in.

Number of Likes
numberOfLikes
This attribute specifies the number of people who like this blog.

Author's portal image URL
portalAuthorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL where you can download the image from Connections by using the Ajax proxy of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's portal URL
portalAuthorLink
This attribute specifies the link for rendering the details view of the author of this item in the context of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Portal view URL
portalLink
This attribute specifies the WebSphere Portal Express URL that points to the details view of this blog. This attribute is computed by the social rendering Digital Data Connector plug-in.

Published date
published
This attribute indicates the time when the social object was published. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Author's raw Atom entry URL
rawAuthorEntryLink
Link to the Atom entry that represents the author of this item. This URL directly addresses the Connections server. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's raw image URL
rawAuthorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL for downloading the image directly from Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Raw Community entry URL
rawCommunityEntryLink
This attribute specifies the Connections URL of the Atom entry that represents the community that contains this blog. This attribute is empty for stand-alone blogs.

Raw view URL
rawLink
This attribute specifies the URL that points to the details view of this blog on the Connections server.

Raw service doc URL
rawServiceDocLink
This attribute specifies the URL of the service document feed of this blog. This attribute is computed by the social rendering Digital Data Connector plug-in.

Summary
summary
A summary of the content of this item.

Tags
tags This attribute specifies the tags that are associated with this item.

Time zone
timezone
This attribute specifies the time zone information for the current blog.

Title
title The title of this item.

Updated date
updated
This attribute indicates the time of the last update of the social object. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Blog UUID
uuid This attribute specifies the UUID of the current blog.

Vote limit
voteLimit
This attribute represents the configured vote limit for this blog.

Number of remaining votes
votesLeft
This attribute represents the number of votes that remain for this blog.

Accept value for weblog entries
weblogEntriesAccept
This attribute specifies the accept value for weblog entries for this blog. For more information, read the Connections Remote API documentation.

In addition to these attributes, the blogs profile also provides access to the following list properties:

Updated date
updated
This list property indicates the time of the last update of the list that is represented by the feed at hand. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Raw URL to this feed
selfLink
This list property specifies the link to this feed.

Feed title
title This list property specifies the link to this feed.

Total Number of Items
totalNumberOfItems
This list property specifies the total number of items that were found for this search.

Requested Number of Items
requestedNumberOfItems
This list property specifies the requested number of items for this search.

Start Index
startIndex
This list property specifies the start index for the paged list of items.

Raw first URL
rawFirstLink
This list property specifies the Connections URL of the first page of items that were found for this search.

Raw next URL
rawNextLink
This list property specifies the Connections URL of the next page of items that were found for this search.

Raw previous URL
rawPreviousLink
This list property specifies the Connections URL of the previous page of items that were found for this search.

Raw last URL
rawLastLink
This list property specifies the Connections URL of the last page of items that are found for this search.

Is truncated
isTruncated
If the list that represents a search result is truncated, this list property returns the string `true`. Otherwise, it returns `false`.

Raw tags link
rawTagsLink
This attribute specifies the URL of the Atom feed that represents the tags for this list of blogs.

Blog details profile

The blog details profile provides access to Connections blog details feed data. It declares the following attribute names:

Can create blog post
canCreateBlogPost
If the current user has permission to create posts for this blog, this attribute returns the string `true`. Otherwise, it returns `false`. This attribute is computed by the social rendering Digital Data Connector plug-in.

ID
id The unique ID of this blog.

Blog posts URL
rawBlogPostsLink
This attribute specifies the URL that points to the Atom feed serving the blog posts for this blog.

Raw Atom entry URL
rawEntryLink
This attribute specifies the URL of the Atom resource that represents this blog.

Raw view URL
rawLink
This attribute specifies the URL pointing to the details view of this blog on the Connections server.

Raw service doc URL
rawServiceDocLink
This attribute specifies the URL of the service document feed of this blog. This attribute is computed by the social rendering Digital Data Connector plug-in.

Raw Weblog Entries URL
rawWeblogEntriesLink
This attribute specifies the URL of the Weblog entries feed of this blog.

Summary
summary
This attribute specifies the summary of the content of this blog.

Title
title This attribute specifies the title of this blog.

Updated date
updated
This attribute indicates the most recent time when this blog was updated. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Accept value for weblog entries
weblogEntriesAccept
This attribute specifies the accept value for weblog entries for this blog.

Blog posts profile

The blog posts profile provides access to Connections blog posts feed data. It declares the following attribute names:

Author's email address
authorEmail
: This attribute references the email address of the author of the social object.

Author's ID
authorID
: This attribute references the internal ID of the author of the social object. It represents the author ID from the Connections server where the social object is stored.

Author's image URL
authorImageLink
: This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL where you can download the image either directly from Connections or by using the Ajax proxy of the portal. For more information about this attribute, read *Configuring globally how social object data is served*. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's name
authorName
: You can use this attribute to include the name of the author of the social object in the design of a social list.

Author's object ID
authorObjectID
: This attribute references the serialized object ID of the author of the social object. In contrast to the `authorID` attribute, the `authorObjectID` attribute represents an ID that is used by WebSphere Portal Express rather than by Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's UID
authorUID
: This attribute specifies the value of the UID attribute of the author of the social object. This value reflects the UID user attribute as defined by the user repository of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Body HTML
body HTML
: The HTML content of this item.

Body plain
body plain
: The plain content of this item.

Body content type
bodyContentType
: This attribute specifies the content type identifier for the body attribute of this item.

Can create blog post comment
canCreateBlogPostComment
If the current user has permission to create comments for this blog post, this attribute returns the string `true`. Otherwise, it returns `false`. This attribute is computed by the social rendering Digital Data Connector plug-in.

Can delete blog post.
canDeleteBlogPost
If the current user has permission to delete this blog post, this attribute returns the string `true`. Otherwise, it returns `false`.

Can edit blog post.
canEditBlogPost
If the current user has permission to edit this blog post, this attribute returns the string `true`. Otherwise, it returns `false`. This attribute is computed by the social rendering Digital Data Connector plug-in.

Accept value for comment entries
commentEntriesAccept
This attribute specifies the accept value for comment entries for this blog post.

Community UUID
communityUUID
If the current blog post is contained in a community blog, this attribute specifies the UUID of the community that contains the blog. If the current blog post is not contained in a community blog, this attribute is empty.

Feed entry
entry The original feed entry of this item.

ID
id The unique identifier for this item.

View URL
link This attribute specifies the URL that points to the details view of this blog post. This attribute is computed by the social rendering Digital Data Connector plug-in.

Number of comments
numberOfComments
This attribute specifies the number of comments that were posted on this blog post.

Number of Likes
numberOfLikes
This attribute specifies the number of people who like this blog post.

Author's portal image URL
portalAuthorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL where you can download the image from Connections by using the Ajax proxy of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's portal URL
portalAuthorLink
This attribute specifies the link for rendering the details view of the author

of this item in the context of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Portal view URL
portalLink
This attribute specifies the WebSphere Portal Express URL that points to the details view of this blog post. This attribute is computed by the social rendering Digital Data Connector plug-in.

Published date
published
This attribute indicates the time when the social object was published. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Author's raw Atom entry URL
rawAuthorEntryLink
Link to the Atom entry that represents the author of this item. This URL directly addresses the Connections server. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's raw image URL
rawAuthorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL for downloading the image directly from Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Raw publishing entry URL
rawBlogPostPublishingEntryLink
This attribute specifies the URL to the publishing entry of this blog post. This attribute is computed by the social rendering Digital Data Connector plug-in.

Raw edit comments URL
rawCommentsEditLink
This attribute specifies the Connections URL that points to the comments collection of this blog post.

Raw View comments URL
rawCommentsReadLink
This attribute specifies the Connections URL that points to the feed of comments for this blog post.

Raw edit entry URL
rawEditLink
This attribute specifies the URL to the editable entry of this blog post.

Raw Atom entry URL
rawEntryLink
This attribute specifies the Connections URL of the Atom entry that represents this blog post.

Raw likes edit URL
rawLikesEditLink
This attribute specifies the Connections URL of the entry that makes it possible to edit the "liked" status of this blog post.

Raw View Likes URL
rawLikesLink
This attribute specifies the Connections URL that points to the "recommend" collection of this blog post.

Raw view URL
rawLink
This attribute specifies the URL of the details view of this blog post on the Connections server.

Raw service doc URL
rawServiceDocLink
This attribute specifies the URL of the service document feed of this blog. This attribute is computed by the social rendering Digital Data Connector plug-in.

Summary
summary
A summary of the content of this item.

Tags
tags This attribute specifies the tags that are associated with this item.

Title
title The title of this item.

Updated date
updated
This attribute indicates the time of the last update of the social object. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

User likes
userLikes
If this blog post is liked by the current user, this attribute returns the string `true`. Otherwise, it returns `false`.

In addition to these attributes, the blog posts profile also provides access to the following list properties:

Updated date
updated
This list property indicates the time of the last update of the list that is represented by the feed at hand. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Raw URL to this feed
selfLink
This list property specifies the link to this feed.

Feed title
title This list property specifies the link to this feed.

Total Number of Items
totalNumberOfItems
This list property specifies the total number of items that were found for this search.

Requested Number of Items
requestedNumberOfItems
This list property specifies the requested number of items for this search.

Start Index
startIndex
This list property specifies the start index for the paged list of items.

Raw first URL
rawFirstLink
This list property specifies the Connections URL of the first page of items that were found for this search.

Raw next URL
rawNextLink
This list property specifies the Connections URL of the next page of items that were found for this search.

Raw previous URL
rawPreviousLink
This list property specifies the Connections URL of the previous page of items that were found for this search.

Raw last URL
rawLastLink
This list property specifies the Connections URL of the last page of items that are found for this search.

Is truncated
isTruncated
If the list that represents a search result is truncated, this list property returns the string `true`. Otherwise, it returns `false`.

Raw tags link
rawTagsLink
This attribute specifies the URL of the Atom feed that represents the tags for this list of blog posts.

Blog post comments profile

The blog post comments profile provides access to Connections blog post comments feed data. It declares the following attribute names:

Author's email address
authorEmail
This attribute references the email address of the author of the social object.

Author's ID
authorID
This attribute references the internal ID of the author of the social object. It represents the author ID from the Connections server where the social object is stored.

Author's image URL
authorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL where you can download the image either directly from Connections or by using the Ajax proxy of the portal. For more information about this attribute, read *Configuring globally how social object data is served*. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's name
authorName
You can use this attribute to include the name of the author of the social object in the design of a social list.

Author's object ID
authorObjectID
This attribute references the serialized object ID of the author of the social object. In contrast to the `authorID` attribute, the `authorObjectID` attribute represents an ID that is used by WebSphere Portal Express rather than by Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's UID
authorUID
This attribute specifies the value of the UID attribute of the author of the social object. This value reflects the UID user attribute as defined by the user repository of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Body HTML
body HTML
The HTML content of this item.

Body plain
body plain
The plain content of this item.

Body content type
bodyContentType
This attribute specifies the content type identifier for the body attribute of this item.

Can delete blog post comment
canDeleteBlogPostComment
If the current user has permission to delete this comment, this attribute returns the string `true`. Otherwise, it returns `false`. This attribute is computed by the social rendering Digital Data Connector plug-in.

Feed entry
entry The original feed entry of this item.

ID
id The unique identifier for this item.

Number of Likes
numberOfLikes
This attribute specifies the number of people who like this blog post comment.

Author's portal image URL
portalAuthorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL where you can download the image from Connections by using the Ajax proxy of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's portal URL
portalAuthorLink

This attribute specifies the link for rendering the details view of the author of this item in the context of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Published date
published

This attribute indicates the time when the social object was published. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Author's raw Atom entry URL
rawAuthorEntryLink

Link to the Atom entry that represents the author of this item. This URL directly addresses the Connections server. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's raw image URL
rawAuthorImageLink

This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL for downloading the image directly from Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Raw publishing entry URL
rawBlogPostCommentPublishingEntryLink

This attribute specifies the URL to the publishing entry of this blog post comment. This attribute is computed by the social rendering Digital Data Connector plug-in.

Raw edit URL
rawEditLink

This attribute specifies the Connections URL of the entry that makes it possible to edit this blog post comment.

Raw Atom entry URL
rawEntryLink

This attribute specifies the Connections URL of the Atom entry that represents this blog post comment.

Raw likes edit URL
rawLikesEditLink

This attribute specifies the Connections URL of the entry that makes it possible to edit the "liked" status of this blog post comment.

Raw view Likes URL
rawLikesLink

This attribute specifies the Connections URL that points to the "recommend" collection of this blog post comment.

Raw view URL
rawLink

This attribute specifies the URL of the details view of this blog post comment on the Connections server.

Summary
summary

This attribute specifies the summary of the content of this blog post comment.

Title
title The title of this item.

Updated date
updated
This attribute indicates the time of the last update of the social object. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

User likes
userLikes
If this blog post comment is liked by the current user, this attribute returns the string `true`. Otherwise, it returns `false`.

In addition to these attributes, the blog post comments profile also provides access to the following list properties:

Updated date
updated
This list property indicates the time of the last update of the list that is represented by the feed at hand. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Raw URL to this feed
selfLink
This list property specifies the link to this feed.

Feed title
title This list property specifies the link to this feed.

Total Number of Items
totalNumberOfItems
This list property specifies the total number of items that were found for this search.

Requested Number of Items
requestedNumberOfItems
This list property specifies the requested number of items for this search.

Start Index
startIndex
This list property specifies the start index for the paged list of items.

Raw first URL
rawFirstLink
This list property specifies the Connections URL of the first page of items that were found for this search.

Raw next URL
rawNextLink
This list property specifies the Connections URL of the next page of items that were found for this search.

Raw previous URL
rawPreviousLink
This list property specifies the Connections URL of the previous page of items that were found for this search.

Raw last URL
rawLastLink
This list property specifies the Connections URL of the last page of items that are found for this search.

Is truncated
isTruncated
If the list that represents a search result is truncated, this list property returns the string `true`. Otherwise, it returns `false`.

Related tasks:

"Configuring globally how social object data is served" on page 2125
Among other features, social objects have different links that enable users to download related data. For example, users can use links to download a file or the profile image of the author. You can globally configure how social object data is served to the users. To do so, you use a setting in the WP Connections Integration Service resource environment provider in the WebSphere Integrated Solutions Console.

Communities and community members profiles:

These profiles provide access to IBM Connections community and community members feed data.

Communities profile

The communities profile provides access to Connections communities feed data. It declares the following attribute names:

Author's email address
authorEmail
This attribute references the email address of the author of the social object.

Author's ID
authorID
This attribute references the internal ID of the author of the social object. It represents the author ID from the Connections server where the social object is stored.

Author's image URL
authorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL where you can download the image either directly from Connections or by using the Ajax proxy of the portal. For more information about this attribute, read *Configuring globally how social object data is served*. This attribute is computed by the social rendering IBM Digital Data Connector (DDC) for WebSphere Portal Express plug-in.

Author's name
authorName
You can use this attribute to include the name of the author of the social object in the design of a social list.

Author's object ID
authorObjectID
This attribute references the serialized object ID of the author of the social object. In contrast to the `authorID` attribute, the `authorObjectID` attribute

represents an ID that is used by WebSphere Portal Express rather than by Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's UID
authorUID
This attribute specifies the value of the UID attribute of the author of the social object. This value reflects the UID user attribute as defined by the user repository of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Body HTML
body HTML
The HTML content of this item.

Body plain
body plain
The plain content of this item.

Body content type
bodyContentType
This attribute specifies the content type identifier for the body attribute of this item.

Community image URL
communityImageLink
This attribute specifies the URL to the image of this community. This attribute is computed by the social rendering Digital Data Connector plug-in.

Community UUID
communityUUID
This attribute specifies the UUID of this community.

Feed entry
entry The original feed entry of this item.

ID
id The unique identifier for this item.

Is moderated
isModerated
If this community is a moderated community, this attribute returns the string `true`. Otherwise, it returns `false`.

Is post-moderated
isPostModerated
If the `post-moderation` feature was enabled for this community, this attribute returns the string `true`. Otherwise, it returns `false`.

Is pre-moderated
isPreModerated
If the `pre-moderation` feature was enabled for this community, this attribute returns the string `true`. Otherwise, it returns `false`.

Is public
isPublic
If this community is a public community, this attribute returns the string `true`. Otherwise, it returns `false`.

Is restricted
isRestricted
If this community is a restricted (invite-only) community, this attribute returns the string `true`. Otherwise, it returns `false`.

View URL
link This attribute specifies the URL that points to the details view of this community. This attribute is computed by the social rendering Digital Data Connector plug-in.

Number of members
numberOfMembers
This attribute specifies the number of members of this community.

Author's portal image URL
portalAuthorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL where you can download the image from Connections by using the Ajax proxy of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's portal URL
portalAuthorLink
This attribute specifies the link for rendering the details view of the author of this item in the context of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Portal community image URL
portalCommunityImageLink
This attribute specifies the WebSphere Portal Express URL to the image of this community. This attribute is computed by the social rendering Digital Data Connector plug-in.

Portal view URL
portalLink
This attribute specifies the WebSphere Portal Express URL that points to the details view of this community. This attribute is computed by the social rendering Digital Data Connector plug-in.

Published date
published
This attribute indicates the time when the social object was published. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Raw activity stream URL
rawActivityStreamLink
This attribute specifies the Connections URL of the subcommunities feed of this community.

Author's raw Atom entry URL
rawAuthorEntryLink
Link to the Atom entry that represents the author of this item. This URL directly addresses the Connections server. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's raw image URL
rawAuthorImageLink
This attribute specifies the link to the profile image of the author of the

social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL for downloading the image directly from Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Raw bookmarks URL
rawBookmarksLink
This attribute specifies the Connections URL of the bookmarks feed of this community.

Raw community image URL
rawCommunityImageLink
This attribute specifies the Connections URL to the image of this community. This attribute is computed by the social rendering Digital Data Connector plug-in.

Raw Atom entry URL
rawEntryLink
This attribute specifies the Connections URL of the Atom entry that represents this community.

Raw feeds URL
rawFeedsLink
This attribute specifies the Connections URL of the feeds feed of this community.

Raw forum topics URL
rawForumTopicsLink
This attribute specifies the Connections URL of the forum topics feed of this community.

Raw invites URL
rawInvitesLink
This attribute specifies the Connections URL of the invitations feed of this community.

Raw view URL
rawLink
This attribute specifies the URL of the details view of this community on the Connections server.

Raw members URL
rawMembersLink
This attribute specifies the Connections URL of the members feed of this community.

Raw parent community URL
rawParentCommunityLink
This attribute specifies the Connections URL of the parent communities feed of this community.

Raw subcommunities URL
rawSubCommunitiesLink
This attribute specifies the Connections URL of the subcommunities feed of this community.

Summary
summary
This attribute specifies the summary description of this community.

Tags
tags This attribute specifies the tags that are associated with this item.

Title
title The title of this item.

Updated date
updated
This attribute indicates the time of the last update of the social object. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

User is Member
userIsMember
If the current user is a member of the current community, this attribute returns the string `true`. Otherwise, it returns `false`. This attribute is computed by the social rendering Digital Data Connector plug-in.

User membership edit URL
userMembershipEditLink
This attribute specifies the URL to the editable resource that represents the membership relation between the currently logged in user and the current community.

User Membership URL
userMembershipLink
This attribute specifies the URL to the resource that represents the membership relation between the currently logged in user and the current community. This attribute is computed by the social rendering Digital Data Connector plug-in.

User Membership Role
userMembershipRole
This link specifies the membership role of the currently logged in user in the current community.

In addition to these attributes, the communities profile also provides access to the following list properties:

Updated date
updated
This list property indicates the time of the last update of the list that is represented by the feed at hand. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Raw URL to this feed
selfLink
This list property specifies the link to this feed.

Feed title
title This list property specifies the link to this feed.

Total Number of Items
totalNumberOfItems
This list property specifies the total number of items that were found for this search.

Requested Number of Items
requestedNumberOfItems
This list property specifies the requested number of items for this search.

Start Index
startIndex
This list property specifies the start index for the paged list of items.

Raw first URL
rawFirstLink
This list property specifies the Connections URL of the first page of items that were found for this search.

Raw next URL
rawNextLink
This list property specifies the Connections URL of the next page of items that were found for this search.

Raw previous URL
rawPreviousLink
This list property specifies the Connections URL of the previous page of items that were found for this search.

Raw last URL
rawLastLink
This list property specifies the Connections URL of the last page of items that are found for this search.

Is truncated
isTruncated
If the list that represents a search result is truncated, this list property returns the string `true`. Otherwise, it returns `false`.

Community members profile

The communities members profile provides access to Connections communities members feed data. It declares the following attribute names:

Body HTML
body HTML
The HTML content of this item.

Body plain
body plain
The plain content of this item.

Body content type
bodyContentType
This attribute specifies the content type identifier for the body attribute of this item.

ID
id ID

Member is active
memberActive
This attribute indicates whether this community member is currently available.

Member's email
memberEmail
This attribute specifies the email address of this community member.

Member's ID
memberID
This attribute specifies the internal ID of this community member.

Member's image URL
memberImageLink
This attribute specifies the URL of the profile image of this community member. This attribute is computed by the social rendering Digital Data Connector plug-in.

Member's name
memberName
This attribute specifies the name of this community member.

Member's role
memberRole
This attribute specifies the role of the community member.

Member's type
memberType
This attribute specifies the type of the community member, for example, whether the member is a person or a group.

Portal view URL
portalLink
This attribute specifies the WebSphere Portal Express URL that points to the details view of this community member. This attribute is computed by the social rendering Digital Data Connector plug-in.

Portal member's image URL
portalMemberImageLink
This attribute specifies the WebSphere Portal Express URL of the profile image of this community member. This attribute is computed by the social rendering Digital Data Connector plug-in.

Published date
published
This attribute indicates the time at which this community member record was created. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Raw Atom entry URL
rawEntryLink
This attribute specifies the Connections URL of the Atom entry that represents this community member.

Raw member's image URL
rawMemberImageLink
This attribute specifies the Connections URL of the profile image of this community member. This attribute is computed by the social rendering Digital Data Connector plug-in.

Raw profile URL
rawProfileLink
This attribute specifies the Connections URL of the user profile Atom entry that represents this community member.

Raw VCard URL
rawVCardLink
This attribute specifies the Connections URL of the VCard resource of this community member.

Summary
summary
This attribute specifies the summary information for this community member.

Title
title Title

Updated date
updated
This attribute indicates the most recent time at which this community record was updated. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

In addition to these attributes, the community members profile also provides access to the following list properties:

Updated date
updated
This list property indicates the time of the last update of the list represented by the feed at hand. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Raw URL to this feed
selfLink
This list property specifies the link to this feed.

Feed title
title This list property specifies the link to this feed.

Total Number of Items
totalNumberOfItems
This list property specifies the total number of items that were found for this search.

Requested Number of Items
requestedNumberOfItems
This list property specifies the requested number of items for this search.

Start Index
startIndex
This list property specifies the start index for the paged list of items.

Raw first URL
rawFirstLink
This list property specifies the Connections URL of the first page of items that were found for this search.

Raw next URL
rawNextLink
This list property specifies the Connections URL of the next page of items that were found for this search.

Raw previous URL
rawPreviousLink
This list property specifies the Connections URL of the previous page of items that were found for this search.

Raw last URL
rawLastLink
This list property specifies the Connections URL of the last page of items found for this search.

Is truncated
isTruncated
If the list that represents a search result is truncated, this list property returns the string `true`. Otherwise, it returns `false`.

Related tasks:

"Configuring globally how social object data is served" on page 2125
Among other features, social objects have different links that enable users to download related data. For example, users can use links to download a file or the profile image of the author. You can globally configure how social object data is served to the users. To do so, you use a setting in the WP Connections Integration Service resource environment provider in the WebSphere Integrated Solutions Console.

Forum-related profiles:

These profiles provide access to IBM Connections forum-related feed data for forums, forum topics, and forum topic replies profiles.

Forums profile

The forums profile provides access to Connections forums feed data. It declares the following attribute names:

Author's email address
authorEmail
This attribute references the email address of the author of the social object.

Author's ID
authorID
This attribute references the internal ID of the author of the social object. It represents the author ID from the Connections server where the social object is stored.

Author's image URL
authorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL where you can download the image either directly from Connections or by using the Ajax proxy of the portal. For more information about this attribute, read *Configuring globally how social object data is served*. This attribute is computed by the social rendering IBM Digital Data Connector (DDC) for WebSphere Portal Express plug-in.

Author's name
authorName
You can use this attribute to include the name of the author of the social object in the design of a social list.

Author's object ID
authorObjectID
This attribute references the serialized object ID of the author of the social object. In contrast to the `authorID` attribute, the `authorObjectID` attribute

represents an ID that is used by WebSphere Portal Express rather than by Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's UID
authorUID
This attribute specifies the value of the UID attribute of the author of the social object. This value reflects the UID user attribute as defined by the user repository of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Body HTML
body HTML
The HTML content of this item.

Body plain
body plain
The plain content of this item.

Body content type
bodyContentType
This attribute specifies the content type identifier for the body attribute of this item.

Can create topic
canCreateTopic
If the user can create topics for the current forum (community forum or stand-alone forum), this attribute returns the string `true`. Otherwise, it returns `false`. This attribute is computed by the social rendering Digital Data Connector plug-in.

CanPostTopic
canPostTopicCmtyForum
If the user can create topics for this community forum, this attribute returns the string `CanPostTopic`. Otherwise, it returns an empty string. This function includes stand-alone forums.

Raw community Atom entry URL
communityLink
This attribute specifies the Connections URL of the Atom entry that represents the community that contains this forum. This attribute is empty for stand-alone forums.

UUID of the associated community
communityUUID
If the current forum is a community forum. This attribute specifies the UUID of the community that contains the forum. If the current forum is not contained in a community, this attribute is empty.

Feed entry
entry The original feed entry of this item.

Atom feed URL
entryLink
This attribute specifies the URL of the Atom entry that represents this forum. This attribute is computed by the social rendering Digital Data Connector plug-in.

ID
id The unique identifier for this item.

Is community forum
isCommunityForum
If this forum is a community forum, this attribute returns the string `true`. Otherwise, it returns `false`.

Is locked
isLocked
If this forum is locked, this attribute returns the string `true`. Otherwise, it returns `false`.

View URL
link This attribute specifies the URL that points to the details view of this forum. This attribute is computed by the social rendering Digital Data Connector plug-in.

Member list
memberList
This attribute lists the members of this forum.

Raw member list link
memberListLink
This attribute specifies the Connections URL of the members feed of this forum.

Is moderated
moderation
If the moderation feature is enabled for the current forum, this attribute returns the string `true`. Otherwise, it returns `false`.

Number of topics
numberOfTopics
This attribute specifies the number of forum topics that exist in this forum.

Author's portal image URL
portalAuthorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL where you can download the image from Connections by using the Ajax proxy of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's portal URL
portalAuthorLink
This attribute specifies the link for rendering the details view of the author of this item in the context of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Portal Atom Feed URL
portalEntryLink
This attribute specifies the WebSphere Portal Express URL of the Atom entry that represents this forum. This attribute is computed by the social rendering Digital Data Connector plug-in.

Portal view URL
portalLink
This attribute specifies the WebSphere Portal Express URL that points to the details view of this forum. This attribute is computed by the social rendering Digital Data Connector plug-in.

Published date
published
This attribute indicates the time when the social object was published. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Author's raw Atom entry URL
rawAuthorEntryLink
Link to the Atom entry that represents the author of this item. This URL directly addresses the Connections server. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's raw image URL
rawAuthorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL for downloading the image directly from Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Raw replies URL for community forums
rawCmtyForumRepliesLink
This attribute specifies the Connections URL of the replies feed for this community forum. This feed includes data about access rights for creating topics. This attribute is computed by the social rendering Digital Data Connector plug-in.

Raw Atom entry URL
rawEntryLink
This attribute specifies the Connections URL of the Atom entry that represents this forum.

Raw view URL
rawLink
This attribute specifies the Connections URL that points to the details view of this forum.

Raw replies URL
rawRepliesLink
This attribute specifies the Connections URL of the replies feed for this forum.

Summary
summary
This attribute specifies the summary description of this forum.

Tags
tags This attribute specifies the tags that are associated with this item.

Title
title The title of this item.

Type
type This attribute specifies the type of this forum.

Updated date
updated
This attribute indicates the time of the last update of the social object. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

In addition to these attributes, the forums profile also provides access to the following list properties:

Updated date
updated
This list property indicates the time of the last update of the list that is represented by the feed at hand. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Raw URL to this feed
selfLink
This list property specifies the link to this feed.

Feed title
title This list property specifies the link to this feed.

Total Number of Items
totalNumberOfItems
This list property specifies the total number of items that were found for this search.

Requested Number of Items
requestedNumberOfItems
This list property specifies the requested number of items for this search.

Start Index
startIndex
This list property specifies the start index for the paged list of items.

Raw first URL
rawFirstLink
This list property specifies the Connections URL of the first page of items that were found for this search.

Raw next URL
rawNextLink
This list property specifies the Connections URL of the next page of items that were found for this search.

Raw previous URL
rawPreviousLink
This list property specifies the Connections URL of the previous page of items that were found for this search.

Raw last URL
rawLastLink
This list property specifies the Connections URL of the last page of items that are found for this search.

Is truncated
isTruncated
If the list that represents a search result is truncated, this list property returns the string `true`. Otherwise, it returns `false`.

Raw tags link
rawTagsLink
This attribute specifies the URL of the Atom resource that represents the tag cloud of this forum.

Forum topics profile

The forum topics profile provides access to Connections forum topics feed data. It declares the following attribute names:

Author's email address
authorEmail
This attribute references the email address of the author of the social object.

Author's ID
authorID
This attribute references the internal ID of the author of the social object. It represents the author ID from the Connections server where the social object is stored.

Author's image URL
authorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL where you can download the image either directly from Connections or by using the Ajax proxy of the portal. For more information about this attribute, read *Configuring globally how social object data is served*. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's name
authorName
You can use this attribute to include the name of the author of the social object in the design of a social list.

Author's object ID
authorObjectID
This attribute references the serialized object ID of the author of the social object. In contrast to the `authorID` attribute, the `authorObjectID` attribute represents an ID that is used by WebSphere Portal Express rather than by Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's UID
authorUID
This attribute specifies the value of the UID attribute of the author of the social object. This value reflects the UID user attribute as defined by the user repository of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Body HTML
body HTML
The HTML content of this item.

Body plain
body plain
The plain content of this item.

Body content type
bodyContentType
This attribute specifies the content type identifier for the body attribute of this item.

Can create reply
canCreateReply
If the current user has permission to reply to this forum topic, this attribute returns the string `true`. Otherwise, it returns `false`. This attribute is computed by the social rendering Digital Data Connector plug-in.

Can create reply
canDeleteTopic
If the current user has permission to delete this forum topic, this attribute returns the string `true`. Otherwise, it returns `false`. This attribute is computed by the social rendering Digital Data Connector plug-in.

Feed entry
entry The original feed entry of this item.

Atom entry URL
entryLink
This attribute specifies the URL of the Atom entry that represents this forum topic. This attribute is computed by the social rendering Digital Data Connector plug-in.

ID
id The unique identifier for this item.

Is Answered question
isAnswered
If this forum topic is a question and the question was answered, this attribute returns the string `true`. Otherwise, it returns `false`.

Is topic locked
isLocked
If this forum topic is locked, this attribute returns the string `true`. Otherwise, it returns `false`.

Is question
isQuestion
If this forum topic is a question, this attribute returns the string `true`. Otherwise, it returns `false`.

View URL
link This attribute specifies the URL that points to the details view of this forum. This attribute is computed by the social rendering Digital Data Connector plug-in.

Number of Likes
numberOfLikes
This attribute specifies the number of people who like this forum topic.

Number of replies
numberOfReplies
This attribute specifies the number of replies that exist for this forum topic.

Permissions
permissions
This attribute specifies the permissions that the current user was granted on the current forum topic.

Author's portal image URL
portalAuthorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL where

you can download the image from Connections by using the Ajax proxy of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's portal URL
portalAuthorLink
This attribute specifies the link for rendering the details view of the author of this item in the context of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Portal Atom entry URL
portalEntryLink
This attribute specifies the WebSphere Portal Express URL of the Atom entry that represents this forum topic. This attribute is computed by the social rendering Digital Data Connector plug-in.

Portal view URL
portalLink
This attribute specifies the WebSphere Portal Express URL that points to the details view of this forum. This attribute is computed by the social rendering Digital Data Connector plug-in.

Published date
published
This attribute indicates the time when the social object was published. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Author's raw Atom entry URL
rawAuthorEntryLink
Link to the Atom entry that represents the author of this item. This URL directly addresses the Connections server. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's raw image URL
rawAuthorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL for downloading the image directly from Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Raw edit URL
rawEditLink
This attribute specifies the Connections URL of the entry that makes it possible to edit this forum topic.

Raw Atom entry URL
rawEntryLink
This attribute specifies the Connections URL of the Atom entry that represents this forum topic.

Raw likes edit URL
rawLikesEditLink
This attribute specifies the Connections URL of the entry that makes it possible to edit the `liked` status of this forum topic.

Raw view URL
rawLink
This attribute specifies the Connections URL that points to the details view of this forum.

Raw Replies URL
rawRepliesLink
This attribute specifies the Connections URL of the replies feed for this forum topic.

Summary
summary
A summary of the content of this item.

Tags
tags This attribute specifies the tags that are associated with this item.

Title
title The title of this item.

Type
type This attribute specifies the type information for this forum topic.

Updated date
updated
This attribute indicates the time of the last update of the social object. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

User likes
userLikes
If this forum topic is liked by the current user, this attribute returns the string `true`. Otherwise, it returns `false`.

In addition to these attributes, the forum topics profile also provides access to the following list properties:

Updated date
updated
This list property indicates the time of the last update of the list that is represented by the feed at hand. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Raw URL to this feed
selfLink
This list property specifies the link to this feed.

Feed title
title This list property specifies the link to this feed.

Total Number of Items
totalNumberOfItems
This list property specifies the total number of items that were found for this search.

Requested Number of Items
requestedNumberOfItems
This list property specifies the requested number of items for this search.

Start Index
startIndex
This list property specifies the start index for the paged list of items.

Raw first URL
rawFirstLink
This list property specifies the Connections URL of the first page of items that were found for this search.

Raw next URL
rawNextLink
This list property specifies the Connections URL of the next page of items that were found for this search.

Raw previous URL
rawPreviousLink
This list property specifies the Connections URL of the previous page of items that were found for this search.

Raw last URL
rawLastLink
This list property specifies the Connections URL of the last page of items that are found for this search.

Is truncated
isTruncated
If the list that represents a search result is truncated, this list property returns the string `true`. Otherwise, it returns `false`.

Raw tags link
rawTagsLink
This attribute specifies the URL of the Atom resourceS that represents the tag cloud of this list of forum topics.

Forum topic replies profile

The forum topic replies profile provides access to Connections forum topic replies feed data. It declares the following attribute names:

Answer flag element
answerFlag
This attribute represents the feed element that indicates whether or not a forum topic reply is marked as the answer to a question.

Author's email address
authorEmail
This attribute references the email address of the author of the social object.

Author's ID
authorID
This attribute references the internal ID of the author of the social object. It represents the author ID from the Connections server where the social object is stored.

Author's image URL
authorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL where you can download the image either directly from Connections or by using the Ajax proxy of the portal. For more information about this attribute, read *Configuring globally how social object data is served*. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's name
authorName
You can use this attribute to include the name of the author of the social object in the design of a social list.

Author's object ID
authorObjectID
This attribute references the serialized object ID of the author of the social object. In contrast to the `authorID` attribute, the `authorObjectID` attribute represents an ID that is used by WebSphere Portal Express rather than by Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's UID
authorUID
This attribute specifies the value of the UID attribute of the author of the social object. This value reflects the UID user attribute as defined by the user repository of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Body HTML
body HTML
The HTML content of this item.

Body plain
body plain
The plain content of this item.

Body content type
bodyContentType
This attribute specifies the content type identifier for the body attribute of this item.

Can accept answer
canAcceptAnswer
If the current user has permission to accept this forum topic reply as the answer to the forum topic question, this attribute returns the string `true`. Otherwise, it returns `false`.

Can create reply
canCreateReply
If the current user has permission to reply on this forum topic reply, this attribute returns the string `true`. Otherwise, it returns `false`. This attribute is computed by the social rendering Digital Data Connector plug-in.

Can decline answer
canDeclineAnswer
If the current user has permission to decline that this forum topic reply answers the forum topic question, this attribute returns the string `true`. Otherwise, it returns `false`.

Can delete reply
canDeleteReply
If the current user has permission to delete this forum topic reply, this attribute returns the string `true`. Otherwise, it returns `false`. This attribute is computed by the social rendering Digital Data Connector plug-in.

Feed entry
entry The original feed entry of this item.

ID
id The unique identifier for this item.

Is answer
isAnswer
If this forum topic reply was marked as an answer to the question, this attribute returns the string `true`. Otherwise, it returns `false`.

Is deleted
isDeleted
If this forum topic reply was deleted, this attribute returns the string `true`. Otherwise, it returns `false`.

Modifier's email address
modifierEmail
This attribute references the email address of the modifier of the social object.

Modifier's ID
modifierID
This attribute references the internal ID of the modifier of the social object.

Modifier's image URL
modifierImageLink
This attribute specifies the link to the profile image of the modifier of the social object. The Connections server that provides the social object also stores the profile image of the modifier. This attribute contains a URL from which users can download the image either directly from Connections or by using the Ajax proxy of the portal. For more information about this attribute, read *Configuring globally how social object data is served*.This attribute is computed by the social rendering bean list provider plug-in.

Modifier's name
modifierName
You can use this attribute to include the name of the modifier of the social object in the design of a social list.

Modifier's object ID
modifierObjectID
This attribute references the serialized object ID of the modifier of the social object. In contrast to the `modifierID` attribute, the `modifierObjectID` attribute represents an ID that is used by WebSphere Portal Express rather than by Connections. This attribute is computed by the social rendering bean list provider plug-in.

Modifier's UID
modifierUID
This attribute specifies the value of the UID attribute of the modifier of the social object. This value reflects the UID user attribute as defined by the user repository of the portal. This attribute is computed by the social rendering bean list provider plug-in.

Nesting level
nestingLevel
This attribute specifies the nesting level of this forum topic reply. This attribute is computed by the social rendering Digital Data Connector plug-in.

Number of attachments
numberOfAttachments
This attribute specifies the number of attachments that were been uploaded for this reply.

Number of nesting levels to close
numberOfLevelsToClose
This attribute returns the delta between the nesting levels of the previous forum topic reply and the current forum topic reply. This attribute is computed by the social rendering Digital Data Connector plug-in.

Number of Likes
numberOfLikes
This attribute specifies the number of people who like this forum topic reply.

Number of replies
numberOfReplies
This attribute specifies the number of replies that were posted to this forum topic reply.

Permissions
permissions
This attribute specifies the permissions that the current user was granted on the current forum topic reply.

Author's portal image URL
portalAuthorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL where you can download the image from Connections by using the Ajax proxy of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's portal URL
portalAuthorLink
This attribute specifies the link for rendering the details view of the author of this item in the context of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Published date
published
This attribute indicates the time when the social object was published. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Author's raw Atom entry URL
rawAuthorEntryLink
Link to the Atom entry that represents the author of this item. This URL directly addresses the Connections server. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's raw image URL
rawAuthorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL for downloading the image directly from Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Raw edit URL
rawEditLink
This attribute specifies the Connections URL of the entry that makes it possible to edit this forum reply.

Raw Atom entry URL
rawEntryLink
This attribute specifies the Connections URL of the Atom entry that represents this forum topic reply.

Raw likes edit URL
rawLikesEditLink
This attribute specifies the Connections URL of the entry that makes it possible to edit the `liked` status of this forum topic reply.

Modifier's raw Atom entry URL
rawModifierEntryLink
Link to the Atom entry that represents the modifier of this item. This URL addresses the Connections server directly. This attribute is computed by the social rendering bean list provider plug-in.

Modifier's raw image URL
rawModifierImageLink
This attribute specifies the link to the profile image of the modifier of the social object. The Connections server that provides the social object also stores the profile image of the modifier. This attribute contains a URL for downloading the image directly from Connections. This attribute is computed by the social rendering bean list provider plug-in.

Modifier's portal image URL
portalModifierImageLink
This attribute specifies the link to the profile image of the modifier of the social object. The Connections server that provides the social object also stores the profile image of the modifier. This attribute contains a URL where you can download the image from Connections by using the Ajax proxy of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Modifier's portal URL
portalModifierLink
This attribute specifies the link for rendering the details view of the modifier of this item in the context of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Raw view replies URL
rawRepliesLink
This attribute specifies the Connections URL of the replies feed for this forum topic reply.

Raw type
rawType
This attribute specifies the type information for this forum topic reply as served by the Connections server.

Raw type document node
rawTypeNode
This attribute selects the document node that provides type information for this forum topic reply that is served by the Connections server.

Reply parent ID
replyParentID
This attribute specifies the internal ID of the parent object of this forum topic reply.

Requires new nesting level
requiresNewLevel
If this forum topic reply is the first nested reply for the previous forum topic reply, this attribute returns the string `true`. This attribute is computed by the social rendering Digital Data Connector plug-in.

Summary
summary
A summary of the content of this item.

Title
title The title of this item.

Type
type This attribute specifies the normalized type information for this forum topic reply. This attribute is computed by the social rendering Digital Data Connector plug-in.

Updated date
updated
This attribute indicates the time of the last update of the social object. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

User likes
userLikes
If this forum topic reply is liked by the current user, this attribute returns the string `true`. Otherwise, it returns `false`.

In addition to these attributes, the forum topic replies profile also provides access to the following list properties:

Updated date
updated
This list property indicates the time of the last update of the list that is represented by the feed at hand. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Raw URL to this feed
selfLink
This list property specifies the link to this feed.

Feed title
title This list property specifies the link to this feed.

Total Number of Items
totalNumberOfItems
This list property specifies the total number of items that were found for this search.

Requested Number of Items
requestedNumberOfItems
This list property specifies the requested number of items for this search.

Start Index
`startIndex`
This list property specifies the start index for the paged list of items.

Raw first URL
`rawFirstLink`
This list property specifies the Connections URL of the first page of items that were found for this search.

Raw next URL
`rawNextLink`
This list property specifies the Connections URL of the next page of items that were found for this search.

Raw previous URL
`rawPreviousLink`
This list property specifies the Connections URL of the previous page of items that were found for this search.

Raw last URL
`rawLastLink`
This list property specifies the Connections URL of the last page of items that are found for this search.

Is truncated
`isTruncated`
If the list that represents a search result is truncated, this list property returns the string `true`. Otherwise, it returns `false`.

Forums topic and reply attachments profile

The forums topic and reply attachments profile provides access to Connections forums attachments feed data. It declares the following attribute names:

Attachment's size
`byteSize`
This attribute specifies the size of the attachment in bytes.

Download URL
`downloadLink`
This attribute represents a link to this attachment. This attribute contains a URL by which users can download the attachment either directly from Connections or by using the Ajax proxy of the portal. If the download link is not available, this attribute does not have a value. For more information about this attribute, read *Configuring globally how social object data is served*. This attribute is computed by the social rendering bean list provider plug-in.

File extension
`fileExtension`
This attribute specifies the file extension of this attachment. For example, the file extension can be `.jpg`, `.mp3`, `.pdf`, or empty. This attribute is computed by the social rendering bean list provider plug-in.

File type
`fileType`
This attribute specifies the file type of this attachment. For example, the file type can be audio, code, compressed, contact, data, flash, graphic, pdf, presentation, text, video, wordProcessing, or dataDocs. The portal sets this attribute according to the value of the `fileExtension` attribute. The

`fileType` attribute value is derived from the configured file type extension mappings of the WP Connections Integration Service resource environment provider. For more information, read *Configuring file type icon mappings*. This attribute is computed by the social rendering bean list provider plug-in.

ID
`id` This attribute specifies the unique identifier for this attachment.

Mime type
`mimeType`
This attribute specifies the mime type of this attachment.

Attachment's name
`name` This attribute specifies the name of this attachment.

Portal download URL
`portalDownloadLink`
This attribute represents a link to this attachment. This attribute contains a URL from which users can download the attachment by using the Ajax proxy of the portal. To download the attachment from Connections without using the Ajax proxy, use the `rawDownloadLink` attribute instead. If the download link is not available, this attribute does not have a value. This attribute is computed by the social rendering bean list provider plug-in.

Portal thumbnail URL
`portalThumbnailLink`
This attribute represents a link to a thumbnail view of this attachment. To point to the thumbnail from Connections without using the Ajax proxy, use the `rawThumbnailLink` attribute instead. If the thumbnail link is not available, this attribute does not have a value. This attribute is computed by the social rendering bean list provider plug-in.

Published date
`published`
This attribute indicates the time when the attachment was published. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Raw download URL
`rawDownloadLink`
This attribute represents a link to this attachment. This attribute contains a URL from which users can download the attachment directly from Connections. To download the attachment from Connections by using the Ajax proxy of the portal, use the `portalDownloadLink` attribute instead. If the download link is not available, this attribute does not have a value.

Raw thumbnail URL
`rawThumbnailLink`
This attribute represents a link to a thumbnail view of this attachment. To point the thumbnail from Connections by using the Ajax proxy of the portal, use the `portalThumbnailLink` attribute instead. If the thumbnail link is not available, this attribute does not have a value.

Thumbnail URL
`thumbnailLink`
This attribute represents a link to a thumbnail view of this attachment. If the thumbnail link is not available, this attribute does not have a value. For

more information about this attribute, read *Configuring globally how social object data is served*. This attribute is computed by the social rendering bean list provider plug-in.

Updated date
`updated`
This attribute indicates the time of the last update of this attachment. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Related tasks:

"Configuring file type icon mappings" on page 2129
Social rendering provides two types of list appearance components for result lists: `simple` and `comprehensive`. Both list appearances components show file type-specific icons when they render list entries that refer to individual files. In this case, the file type is determined based on the file extensions of the individual files. You can configure the set of file types that you want to use in the WP Connections Integration Service resource environment provider. In the context of social lists, a file type is defined by a file type name and a list of file extensions. This list defines the mapping between individual files and your file types. In your social list appearance components, you can then access the file type name for a specific file by using the Web Content Manager `[AttributeResource attributeName="fileType"]` tag. That tag is defined in the IBM Digital Data Connector (DDC) for WebSphere Portal Express profile. You can then use the file type name to render the appropriate image, for example, by assigning a corresponding CSS class.

"Configuring globally how social object data is served" on page 2125
Among other features, social objects have different links that enable users to download related data. For example, users can use links to download a file or the profile image of the author. You can globally configure how social object data is served to the users. To do so, you use a setting in the WP Connections Integration Service resource environment provider in the WebSphere Integrated Solutions Console.

Profiles and profiles connections profiles:

These profiles provide access to IBM Connections profile related feed data.

Profiles profile

The profiles profile provides access to Connections profile feed data. It declares the following attribute names:

Assistant's ID
`assistantID`
This attribute specifies the internal ID of the assistant of the user who represented by this profile.

Assistant's profile URL
`assistantLink`
This attribute specifies the URL of the user profile that represents the assistant of the user who is represented by this profile. This attribute is computed by the social rendering IBM Digital Data Connector (DDC) for WebSphere Portal Express plug-in.

Assistant's name
assistantName
This attribute specifies the name of the assistant of the user who is represented by this profile.

Blog URL
blogURL
This attribute specifies the URL of the current profile.

Building
building
This attribute specifies the building identifier for this profile.

Email
email This attribute specifies the email address of this profile.

Alternate email
emailAlternate
This attribute specifies the alternate email address of this profile.

Fax number
faxNumber
This attribute specifies the FAX number of this profile.

Floor
floor This attribute specifies the floor identifier for this profile.

ID
id The unique identifier for this profile.

Image URL
imageLink
This attribute specifies the URL of the photo of this profile. This attribute is computed by the social rendering Digital Data Connector plug-in.

Is active
isActive
This attribute indicates whether the user represented by this profile is currently available.

Is manager
isManager
If the user represented by this profile is a manager, this attribute returns the string `true`. Otherwise, it returns `false`.

Job title
jobTitle
This attribute specifies the job title of this profile.

UID of the manager of this user
managerUID
This attribute specifies the UID of the profile that represents this user's manager.

Name
name This attribute specifies the name of this profile.

Office number
officeNumber
This attribute specifies the office number of this profile.

Organization unit
organizationUnit
This attribute specifies the organization unit identifier of this profile.

Pager number
pagerNumber
This attribute specifies the pager number of this profile.

Assistant's profile URL
portalAssistantLink
This attribute specifies the WebSphere Portal Express URL of the user profile that represents the assistant of the user who is represented by this profile. This attribute is computed by the social rendering Digital Data Connector plug-in.

Portal image URL
portalImageLink
This attribute specifies the WebSphere Portal Express URL of the photo of this profile. This attribute is computed by the social rendering Digital Data Connector plug-in.

Portal VCard URL
portalVCardLink
This attribute specifies the WebSphere Portal Express URL of the VCard that describes this user profile. This attribute is computed by the social rendering Digital Data Connector plug-in.

Profile key
profileKey
This attribute specifies the key of this profile.

Profile type
profileType
This attribute specifies the profile type of this profile.

Raw assistant's entry URL
rawAssistantEntryLink
This attribute specifies the Connections URL of the user profile entry that represents the assistant of the user who is represented by this profile.

Raw assistant's URL
rawAssistantLink
This attribute specifies the Connections URL of the user profile that represents the assistant of the user who is represented by this profile. This attribute is computed by the social rendering Digital Data Connector plug-in.

Raw Atom entry URL
rawEntryLink
This attribute specifies the Connections URL of the Atom entry that represents this profile.

Raw image URL
rawImageLink
This attribute specifies the Connections URL of the photo of this profile. This attribute is computed by the social rendering Digital Data Connector plug-in.

Raw view URL
rawLink
This attribute specifies the URL of the details view of this profile on the Connections server.

Raw profile type URL
rawProfileTypeLink
This attribute specifies the URL of the profile type resource that is associated with this profile.

Raw pronunciation URL
rawPronunciationLink
This attribute specifies the Connections URL of the pronunciation resource of this profile.

Raw VCard URL
rawVCardLink
This attribute specifies the Connections URL of the VCard resource that is associated with this profile.

Summary
summary
This attribute specifies the summary information for this profile.

Tags
tags This attribute specifies the tags that are associated with this item.

Phone number IP
telephoneNumberIP
This attribute specifies the IP phone number of this profile.

Phone number mobile
telephoneNumberMobile
This attribute specifies the mobile phone number of this profile.

Phone number office
telephoneNumberOffice
This attribute specifies the office phone number of this profile.

Title
title The display name of this profile.

Updated date
updated
This attribute indicates the time of the last update of this profile. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

User's ID
userID This attribute specifies the internal ID of this profile.

Raw user network connection link
rawUserNetworkConnectionLink
This attribute specifies the URL to the user network connection entry that contains data about whether the current user and the selected user are colleagues. This attribute is computed by the social rendering Digital Data Connector plug-in.

User network status
userNetworkStatus
This attribute specifies the status of the connection between the current user and a selected user. This attribute can take the following values:

accepted
This value means that the logged in user accepted an invitation.

pending
This value means that the logged in user can accept an invitation.

unconfirmed
This value means that the invited user did not yet accept the invitation of the current user.

User's UID
userUID
This attribute specifies the UID of this profile.

VCard URL
vCardLink
This attribute specifies the URL of the VCard that describes this user profile. This attribute is computed by the social rendering Digital Data Connector plug-in.

Organization code
xOrganizationCode
This attribute specifies the organization code of this profile.

In addition to these attributes, the profiles profile also provides access to the following list properties:

Raw tags link
rawTagsLink
This list property specifies the URL of the Atom feed that represents the tags for this list of profiles.

Raw colleague link
rawColleagueLink
This list property specifies the URL of the Atom feed that represents the colleagues for the user that is represented by the current feed.

Profiles connections profile

The profiles connections profile provides access to Connections profiles connections feed data. It declares the following attribute names:

Author's email address
authorEmail
This attribute references the email address of the author of the social object.

Author's ID
authorID
This attribute references the internal ID of the author of the social object. It represents the author ID from the Connections server where the social object is stored.

Author's image URL
authorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL where you can download the image either directly from Connections or by using the Ajax proxy of the portal. For more information about this attribute,

read *Configuring globally how social object data is served*. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's name
authorName
You can use this attribute to include the name of the author of the social object in the design of a social list.

Author's object ID
authorObjectID
This attribute references the serialized object ID of the author of the social object. In contrast to the `authorID` attribute, the `authorObjectID` attribute represents an ID that is used by WebSphere Portal Express rather than by Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's UID
authorUID
This attribute specifies the value of the UID attribute of the author of the social object. This value reflects the UID user attribute as defined by the user repository of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Body HTML
body HTML
The HTML content of this item.

Body plain
body plain
The plain content of this item.

Body content type
bodyContentType
This attribute specifies the content type identifier for the body attribute of this item.

Is connection pending
connectionIsPending
If this profile connection is pending, this attribute returns the string `true`. Otherwise, it returns `false`.

Connection message
connectionMessage
This attribute specifies the connection request message.

Connection status
connectionStatus
This attribute specifies the status information of this profile connection.

Connection type
connectionType
This attribute specifies the type information of this profile connection.

Feed entry
entry The original feed entry of this item.

ID
id The unique identifier for this item.

Author's portal image URL
portalAuthorImageLink
This attribute specifies the link to the profile image of the author of the

social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL where you can download the image from Connections by using the Ajax proxy of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's portal URL
portalAuthorLink
This attribute specifies the link for rendering the details view of the author of this item in the context of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Published date
published
This attribute indicates the time when the social object was published. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Author's raw Atom entry URL
rawAuthorEntryLink
Link to the Atom entry that represents the author of this item. This URL directly addresses the Connections server. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's raw image URL
rawAuthorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL for downloading the image directly from Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Raw edit URL
rawEditLink
This attribute specifies the URL of the editable resource that represents this profile connection.

Raw Atom entry URL
rawEntryLink
This attribute represents the Connections link to the Atom resource that represents this social object. This attribute contains a URL where you can access the feed directly from Connections. To retrieve the feed from Connections by using the Ajax proxy of the portal, use the `portalEntryLink` attribute instead.

Summary
summary
A summary of the content of this item.

Title
title The title of this item.

Updated date
updated
This attribute indicates the time of the last update of the social object. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

In addition to these attributes, the profiles connections profile also provides access to the following list properties:

Updated date
updated
This list property indicates the time of the last update of the list that is represented by the feed at hand. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Raw URL to this feed
selfLink
This list property specifies the link to this feed.

Feed title
title This list property specifies the link to this feed.

Related tasks:

"Configuring globally how social object data is served" on page 2125
Among other features, social objects have different links that enable users to download related data. For example, users can use links to download a file or the profile image of the author. You can globally configure how social object data is served to the users. To do so, you use a setting in the WP Connections Integration Service resource environment provider in the WebSphere Integrated Solutions Console.

Social objects profile:

The social objects profile provides access to IBM Connections social objects feed data.

It declares the following attribute names:

Access control
accessControl
This attribute indicates whether this social object is public, private, or shared.

Author's email address
authorEmail
This attribute references the email address of the author of the social object.

Author's ID
authorID
This attribute references the internal ID of the author of the social object. It represents the author ID from the Connections server where the social object is stored.

Author's image URL
authorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL where you can download the image either directly from Connections or by using the Ajax proxy of the portal. For more information about this attribute, read *Configuring globally how social object data is served*. This attribute is computed by the social rendering IBM Digital Data Connector (DDC) for WebSphere Portal Express plug-in.

Author's name
authorName
You can use this attribute to include the name of the author of the social object in the design of a social list.

Author's object ID
authorObjectID
This attribute references the serialized object ID of the author of the social object. In contrast to the `authorID` attribute, the `authorObjectID` attribute represents an ID that is used by WebSphere Portal Express rather than by Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's UID
authorUID
This attribute specifies the value of the UID attribute of the author of the social object. This value reflects the UID user attribute as defined by the user repository of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Body
body This attribute specifies the content of this item.

Body plain
body plain
This attribute specifies the plain content of this item.

Body content type
bodyContentType
This attribute specifies the content type identifier for the body attribute of this social object.

Community Atom feed URL
communityEntryLink
This attribute represents the link to the Atom resource that represents the community of this social object. This attribute contains a URL for accessing the feed either directly from Connections or by using the Ajax proxy of the portal. If the social object does not belong to a community, this attribute does not have a value. For more information about this attribute, read *Configuring globally how social object data is served*. This attribute is computed by the social rendering Digital Data Connector plug-in.

Community view URL
communityLink
This attribute represents the link to the community that contains this social object. This attribute contains a URL by which users can view the community in the context of either Connections or the portal. If the social object does not belong to a community, this attribute does not have a value. For more information about this attribute, read *Configuring globally how social object data is served*. This attribute is computed by the social rendering Digital Data Connector plug-in.

Community UUID
communityUUID
This attribute references the internal Universally Unique Identifier (UUID) of the community that contains this social object. It represents the community identifier from the Connections server where the social object is stored. If the social object does not belong to a community, this attribute does not have a value.

Download URL
downloadLink
If the social object is a file, this attribute represents a link to that file. This attribute contains a URL by which users can download the social object either directly from Connections or by using the Ajax proxy of the portal. If the download link is not available, this attribute does not have a value. For more information about this attribute, read *Configuring globally how social object data is served*. This attribute is computed by the social rendering Digital Data Connector plug-in.

Feed entry
entry The original feed entry of this item.

Atom entry URL
entryLink
This attribute represents the link to the Atom resource that represents this social object. This attribute contains a URL for accessing the feed either directly from Connections or by using the Ajax proxy of the portal. For more information about this attribute, read *Configuring globally how social object data is served*. This attribute is computed by the social rendering Digital Data Connector plug-in.

Event end date
eventEndDate
You can use this attribute to show the end date for this event. If the event recurs and therefore has multiple end dates, this attribute represents the end date that matches the event start date that is available through the `eventStartDate` attribute. To format the generated data string, you can use the format parameter of the `AttributeResource` tag. For more information about the Web Content Manager date formatting options, read *Setting parameters to format dates* in the Web Content Manager product documentation. This attribute is computed by the social rendering Digital Data Connector plug-in.

All event end dates
eventEndDates
This attribute specifies the list of all end dates of this event.

Event parent ID
eventParentID
This attribute specifies the internal ID of the parent event of this event. If this social object is not an event, this attribute is empty.

Event start date
eventStartDate
You can use this attribute to show the start date for this event. If the event recurs and therefore has multiple start dates, this attribute represents the next start date. If all start dates are in the past, the method returns the last start date. The event end date that corresponds to the event start date is available through the `eventEndDate` attribute. To format the generated data string, you can use the format parameter of the `AttributeResource` tag. For more information about the Web Content Manager date formatting options, read *Setting parameters to format dates* in the Web Content Manager product documentation. This attribute is computed by the social rendering Digital Data Connector plug-in.

All event start dates
eventStartDates
This attribute specifies the list of all start dates of this event.

File extension
fileExtension
This attribute carries the file extension of this social object. For example, the file extension can be `.jpg`, `.mp3`, or`.pdf`. If the social object is not a file, this attribute does not have a value.

File type
fileType
This attribute specifies the file type of this social object. For example, the file type can be `audio`, `code`, `compressed`, `contact`, `data`, `flash`, `graphic`, `pdf`, `presentation`, `text`, `video`, `wordProcessing`, or `dataDocs`. The portal sets this attribute according to the value of the `fileExtension` attribute. The `fileType` attribute value is derived from the configured file type extension mappings of the WP Connections Integration Service resource environment provider. For more information, read *Configuring file type icon mappings*. If the social object is not a file, this attribute does not have a value. This attribute is computed by the social rendering Digital Data Connector plug-in.

ID
id The unique identifier for this item.

Is answered question
isAnswered
If this social object is an answered forum topic question, this attribute returns the string `true`. If the social object is a forum topic but not an answered question, it returns `false`. If this social object is not a forum topic, this attribute is empty.

Is All-day event
isEventAllDay
If this social object is an all-day event, this attribute returns the string `true`. If the social object is an event but not an all-day event, it returns `false`. If this social object is not an event, this attribute is empty.

Is repeating event
isEventRepeating
If this social object is a repeating event, this attribute returns the string `true`. If the social object is a non-repeating event, it returns `false`. If this social object is not an event, this attribute is empty.

Is question
isQuestion
If this social object is a forum topic question, this attribute returns the string `true`. If the social object is a forum topic but no question, this attribute returns `false`. If this social object is not a forum topic, this attribute is empty.

Number of Likes
likes This attribute specifies the number of people who like this social object. This attribute is for internal use only and does not show up in the UI.

View URL
link This attribute represents the link to this social object. This attribute contains a URL where you can view the social object in the context of either Connections or the portal. For more information about this attribute, read *Configuring globally how social object data is served*. This attribute is computed by the social rendering Digital Data Connector plug-in.

Live number of comments
liveNumberOfComments

This attribute specifies the number of comments that exist for a file, blog post, or wiki page. In contrast to the numberOfComments attribute, the liveNumberOfComments attribute retrieves its value not from the search result feed but directly from the corresponding Connections service API. As a result, the value reflects the current value of this attribute, independent of the time of the last search crawl. Using this attribute instead of the numberOfComments attribute introduces some performance overhead for getting the live data. Use it only in lists in which the user can create or delete comments. If this social object is not a file, blog post, or wiki page, this attribute returns 0.

Live number of likes
liveNumberOfLikes

This attribute specifies the number of people who liked this social object. In contrast to the numberOfLikes attribute, the liveNumberOfLikes attribute retrieves its value not from the search result feed, but directly from the corresponding Connections service API. As a result, the value reflects the current value of this attribute, independent of the time of the last search crawl. Using this attribute instead of the numberOfLikes attribute introduces some performance overhead for getting the live data. Use it only in lists in which the user can like or unlike the social objects. If this social object does not support the like operation, this attribute returns 0.

Live number of replies
liveNumberOfReplies

This attribute specifies the number of replies that exist for this forum topic. In contrast to the numberOfReplies attribute, the liveNumberOfReplies attribute retrieves its value not from the search result feed but directly from the forums feed. As a result, the value reflects the current value of this attribute, independent of the time of the last search crawl. Using this attribute instead of the numberOfReplies attribute introduces some performance overhead for getting the live data. Use it only in lists in which the user can create or delete replies. If this social object is not a forum topic, this attribute returns 0.

Number of comments
numberOfComments

This attribute specifies the number of comments that are associated with this file, event, or blog post. If the current social object is not a file, event or blog post, this attribute returns 0. This attribute is computed by the social rendering Digital Data Connector plug-in.

Number of Likes
numberOfLikes

This attribute specifies the number of people who like this social object.

Number of replies
numberOfReplies

This attribute specifies the number of replies that exist in this forum topic. If this social object is not a forum topic, this attribute returns 0.

Author's portal image URL
portalAuthorImageLink

This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL where

you can download the image from Connections by using the Ajax proxy of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's portal URL
portalAuthorLink
This attribute specifies the link for rendering the details view of the author of this item in the context of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Portal community Atom entry URL
portalCommunityEntryLink
This attribute represents the link to the Atom resource that represents the community of this social object. This attribute contains a URL for accessing the feed by using the Ajax proxy of the portal. To retrieve the feed from Connections without using the Ajax proxy, use the `rawCommunityEntryLink` attribute instead. If the social object does not belong to a community, this attribute does not have a value. This attribute is computed by the social rendering Digital Data Connector plug-in.

Portal community view URL
portalCommunityLink
This attribute represents the link to the community that contains this social object. This attribute contains a URL by which users can view the community in the context of the portal. To access the community in the context of Connections, use the `rawCommunityLink` attribute instead. If the social object does not belong to a community, this attribute does not have a value. This attribute is computed by the social rendering Digital Data Connector plug-in.

Portal download URL
portalDownloadLink
If the social object is a file, this attribute represents a link to that file. This attribute contains a URL by which users can download the social object by using the Ajax proxy of the portal. To download the social object from Connections without using the Ajax proxy, use the `rawDownloadLink` attribute instead. If the download link is not available, this attribute does not have a value. This attribute is computed by the social rendering Digital Data Connector plug-in.

Portal Atom entry URL
portalEntryLink
This attribute represents the portal link to the Atom resource that represents this social object. This attribute contains a URL for accessing the feed by using the Ajax proxy of the portal. To retrieve the feed from Connections without using the Ajax proxy, use the `rawEntryLink` attribute instead. This attribute is computed by the social rendering Digital Data Connector plug-in.

Portal view URL
portalLink
This attribute represents the link to this social object. This attribute contains a URL where you can view the social object in the context of the portal. To access the social object in the context of Connections, use the `rawLink` attribute instead. This attribute is computed by the social rendering Digital Data Connector plug-in.

User profile job title
profileJobTitle

You can use this attribute to show the job title of individual users when social objects of the type profile are rendered. The attribute value is empty for social objects of types other than profile.

User profile phone number
profilePhoneNumber

You can use this attribute to show the phone number of individual users when social objects of the type profile are rendered. The attribute value is empty for social objects of types other than profile.

Published date
published

This attribute indicates the time when the social object was published. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Author's raw Atom entry URL
rawAuthorEntryLink

Link to the Atom entry that represents the author of this item. This URL directly addresses the Connections server. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's raw image URL
rawAuthorImageLink

This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL for downloading the image directly from Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Raw Comment IDs
rawCommentIDs

This attribute is internally used to compute the `numberOfComments` attribute. It does not show up in the web content authoring UI.

Number of comments
rawComments

This attribute specifies the number of comments that are associated with this social object. This attribute is for internal use only and does not show up in the UI.

Raw community Atom entry URL
rawCommunityEntryLink

This attribute represents the link to the Atom resource that represents the community of this social object. This attribute contains a URL where you can access the feed directly from Connections. To retrieve the feed from Connections by using the Ajax proxy of the portal, use the `portalCommunityEntryLink` instead. If the social object does not belong to a community, this attribute does not have a value.

Raw community view URL
rawCommunityLink

This attribute represents the link to the community that contains this social object. This attribute contains a URL where you can view the community in the context of Connections. To access the community in the context of the portal, use the `portalCommunityLink` attribute instead. If the social object does not belong to a community, this attribute does not have a value.

Raw download URL
rawDownloadLink

If this social object is a file, this attribute represents a link to that file. This attribute contains a URL where you can download the social object directly from Connections. To download the social object from Connections by using the Ajax proxy of the portal, use the `portalDownloadLink` attribute instead. If the download link is not available, this attribute does not have a value.

Raw Atom entry URL
rawEntryLink

This attribute represents the IBM Connections link to the Atom resource that represents this social object. This attribute contains a URL where you can access the feed directly from Connections. To retrieve the feed from Connections by using the Ajax proxy of the portal, use the `portalEntryLink` attribute instead.

Raw view URL
rawLink

This attribute represents the link to this social object. This attribute contains a URL where you can view the social object details in the context of Connections. To access the social object in the context of the portal, use the `portalLink` attribute instead.

Raw number of comments
rawNumberOfComments

This attribute is internally used to compute the `numberOfComments` attribute. It does not show up in the authoring UI.

Raw replies URL
rawRepliesLink

This attribute specifies the Connections URL of the replies feed for this forum topic. If this social object is not a forum topic, this attribute is empty.

Raw type
rawType

This attribute carries the Connections component category term of the social object as it is used by the Connections search API. Among other information, the value includes the name of the Connections service and the service-specific content type. For example, the `rawType` of a social object can be `communities:blogs:entry`. For more information about Connections content types, read the Connections documentation.

Relevance score
relevanceScore

This attribute specifies the relative relevance of this social object in the context of a search query. The value of this attribute typically ranges between 0 and 1. However, values less than 0 or more than 1 can occur. You can consider a score that is smaller than 0 to be equivalent to 0 and a score that is larger than 1 to be equivalent to 1.

Service
service

The service attribute carries the name of the Connections service that hosts the social object. The portal sets this attribute according to the value of the `rawType` attribute and the service type mappings that are configured in the WP Connections Integration Service resource environment provider. For example, such services can be activities, blogs, bookmarks, calendars,

communities, forums, profiles, or wikis. For more information about Connections services, read the Connections documentation. This attribute is computed by the social rendering Digital Data Connector plug-in.

Summary
summary
A summary of the content of this item.

Tags
tags This attribute specifies the tags that are associated with this item.

Title
title The title of this item.

Type
type The type attribute carries the name of the content type of the social object. The portal sets this attribute according to the value of the `rawType` attribute and the resource type mappings that are configured in the WP Connections Integration Service resource environment provider. For example, such types can be `activity`, `blog`, `bookmark`, `calendar`, `community`, `file`, `forum`, `profile`, `statusUpdate`, or `wiki`. For more information about Connections content types, read the Connections documentation. This attribute is computed by the social rendering Digital Data Connector plug-in.

Updated date
updated
This attribute indicates the time of the last update of the social object. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

URI type
uriType
This attribute specifies the type information for the URI of this social object. This attribute is computed by the social rendering Digital Data Connector plug-in.

Related tasks:

"Configuring file type icon mappings" on page 2129
Social rendering provides two types of list appearance components for result lists: `simple` and `comprehensive`. Both list appearances components show file type-specific icons when they render list entries that refer to individual files. In this case, the file type is determined based on the file extensions of the individual files. You can configure the set of file types that you want to use in the WP Connections Integration Service resource environment provider. In the context of social lists, a file type is defined by a file type name and a list of file extensions. This list defines the mapping between individual files and your file types. In your social list appearance components, you can then access the file type name for a specific file by using the Web Content Manager `[AttributeResource attributeName="fileType"]` tag. That tag is defined in the IBM Digital Data Connector (DDC) for WebSphere Portal Express profile. You can then use the file type name to render the appropriate image, for example, by assigning a corresponding CSS class.

"Configuring globally how social object data is served" on page 2125
Among other features, social objects have different links that enable users to download related data. For example, users can use links to download a file or the profile image of the author. You can globally configure how social object data is served to the users. To do so, you use a setting in the WP Connections Integration Service resource environment provider in the WebSphere Integrated Solutions Console.

Related reference:

Setting parameters to format dates
These parameters are used to set the format of dates.

Tags profile:

The tags profile provides access to the IBM Connections tags facet that is served by the connections search service.

The tags profile declares the following attribute names:

ID
id The tag name that represents this tag.

Frequency
frequency
This attribute specifies how often this tag was assigned.

Wiki-related profiles:

These profiles provide access to wiki-related feed data for wiki pages, wiki page comments, and wiki page attachments.

Wiki pages profile

The wiki pages profile provides access to IBM Connections wiki pages feed data. It declares the following attribute names:

Author's email address
authorEmail
This attribute references the email address of the author of the social object.

Author's ID
authorID
This attribute references the internal ID of the author of the social object. It represents the author ID from the Connections server where the social object is stored.

Author's image URL
authorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL from which users can download the image either directly from Connections or by using the Ajax proxy of the portal. For more information about this attribute, read *Configuring globally how social object data is served*. This attribute is computed by the social rendering IBM Digital Data Connector (DDC) for WebSphere Portal Express plug-in.

Author's name
authorName
You can use this attribute to include the name of the author of the social object in the design of a social list.

Author's object ID
authorObjectID
This attribute references the serialized object ID of the author of the social object. In contrast to the `authorID` attribute, the `authorObjectID` attribute

represents an ID that is used by WebSphere Portal Express rather than by Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's UID
authorUID
This attribute specifies the value of the UID attribute of the author of the social object. This value reflects the UID user attribute as defined by the user repository of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Body HTML
body HTML
The HTML content of this item.

Body plain
body plain
The plain content of this item.

Body content type
bodyContentType
This attribute specifies the content type identifier for the body attribute of this item.

Can create comment
canCreateComment
If the current user has permission to create comments to this wiki page, this attribute returns the string `true`. Otherwise, it returns `false`. This attribute is computed by the social rendering Digital Data Connector plug-in.

Download size
downloadSize
This attribute specifies the download size of this wiki page.

Feed entry
entry The original feed entry of this item.

ID
id The unique identifier for this item.

Modifier is active
isModifierActive
This attribute indicates whether the user who made the most recent update to this wiki page is currently available.

Label
label This attribute specifies the label of this wiki page.

Library title
libraryTitle
This attribute specifies the title of the library that contains this wiki page.

Library UUID
libraryUUID
This attribute specifies the UUID of the library that contains this wiki page.

Modifier's email
modifierEmail
This attribute specifies the email address of the user who made the most recent update to this wiki page.

Modifier's ID
modifierID
This attribute specifies the internal ID of the user who made the most recent update to this wiki page.

Modifier's Image URL
modifierImageLink
This attribute specifies the photo URL of the user who made the most recent update to this wiki page. This attribute is computed by the social rendering Digital Data Connector plug-in.

Modifier's name
modifierName
This attribute specifies the name of the user who made the most recent update to this wiki page.

Number of attachments
numberOfAttachments
This attribute specifies the number of documents that are attached to this wiki page.

Number of comments
numberOfComments
This attribute specifies the number of comments that were posted on this wiki page.

Number of Likes
numberOfLikes
This attribute specifies the number of people who like this wiki page.

Number of Versions
numberOfVersions
This attribute specifies the number of versions that exist for this wiki page.

Permissions
permissions
This attribute specifies the permissions that the current user was given on the wiki page.

Author's portal image URL
portalAuthorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL from which users can download the image from Connections by using the Ajax proxy of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's portal URL
portalAuthorLink
This attribute specifies the link for rendering the details view of the author of this item in the context of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Portal modifier's image URL
portalModifierImageLink
This attribute specifies the WebSphere Portal Express photo URL of the user who made the most recent update to this wiki page. This attribute is computed by the social rendering Digital Data Connector plug-in.

Portal modifier's URL
portalModifierLink

This attribute specifies the WebSphere Portal Express URL that points to the details view of the user profile of the user who made the most recent update to this wiki page. This attribute is computed by the social rendering Digital Data Connector plug-in.

Published date
published

This attribute indicates the time when the social object was published. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Author's raw Atom entry URL
rawAuthorEntryLink

Link to the Atom entry that represents the author of this item. This URL directly addresses the Connections server. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's raw image URL
rawAuthorImageLink

This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL for downloading the image directly from Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Portal body
portalBody

This attribute specifies the actual content of the wiki page when URL rewriting is performed. Links from that point to other wiki pages are rewritten into portal URLs so that clicking those links triggers a corresponding social object resolution. File attachment URLs are rewritten into Ajax proxy URLs. This attribute is computed by the social rendering bean list provider plug-in.

Raw body
rawBody

This attribute specifies the actual content of the wiki page. The content is served as is without any URL rewriting. This attribute is computed by the social rendering bean list provider plug-in.

Raw body fragment URL
rawBodyFragmentLink

This attribute specifies the Connections URL of the body fragment of this wiki page.

Raw comments URL
rawCommentsLink

This attribute specifies the Connections URL of the comments feed for this wiki page.

Raw download URL
rawDownloadLink

This attribute specifies the Connections URL of this wiki page. Users can use this URL to download this wiki page.

Raw edit URL
rawEditLink
This attribute specifies the Connections URL of the editable resource that represents this wiki page.

Raw Atom entry URL
rawEntryLink
This attribute specifies the Connections URL of the Atom resource that represents this wiki page.

Raw likes edit URL
rawLikesEditLink
This attribute specifies the Connections URL of the entry that makes it possible to edit the `liked` status of this wiki page.

null
rawLikesRemoveLink
null

Raw view URL
rawLink
This attribute specifies the Connections URL of the resource that holds the information whether the current user likes this wiki page. This resource can be modified.

Raw Modifier's entry URL
rawModifierEntryLink
This attribute specifies the URL of the Atom resource that represents the user profile of the user who made the most recent update to this wiki page. This attribute is computed by the social rendering Digital Data Connector plug-in.

Raw modifier's image URL
rawModifierImageLink
This attribute specifies the Connections photo URL of the user who made the most recent update to this wiki page. This attribute is computed by the social rendering Digital Data Connector plug-in.

Nested comments URL
rawNestedCommentsLink
This attribute specifies the Connections URL of the comments feed for this wiki page. This feed contains data about access rights and the `liked` status. This attribute is computed by the social rendering Digital Data Connector plug-in.

Raw attachments URL
rawWikiPageAttachmentsLink
This attribute specifies the Connections URL of the attachments feed for this wiki page. This attribute is computed by the social rendering Digital Data Connector plug-in.

Summary
summary
A summary of the content of this item.

Tags
tags This attribute specifies the tags that are associated with this item.

Title
title The title of this item.

Updated date
updated
This attribute indicates the time of the last update of the social object. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

User likes
userLikes
If this wiki page is liked by the current user, this attribute returns the string `true`. Otherwise, it returns `false`.

UUID
uuid This attribute specifies the UUID of this wiki page.

Version number
versionNumber
This attribute specifies the version number of this wiki page.

Version UUID
versionUUID
This attribute specifies the version UUID of this wiki page.

Visibility
visibility
This attribute specifies the visibility information of this wiki page.

In addition to these attributes, the wiki pages profile also provides access to the following list properties:

Updated date
updated
This list property indicates the time of the last update of the list that is represented by the feed at hand. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Raw URL to this feed
selfLink
This list property specifies the link to this feed.

Feed title
title This list property specifies the link to this feed.

Wiki page comments profile

The wiki page comments profile provides access to Connections wiki page comments feed data. It declares the following attribute names:

Author's email address
authorEmail
This attribute references the email address of the author of the social object.

Author's ID
authorID
This attribute references the internal ID of the author of the social object. It represents the author ID from the Connections server where the social object is stored.

Author's image URL
authorImageLink

This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL from which users can download the image either directly from Connections or by using the Ajax proxy of the portal. For more information about this attribute, read *Configuring globally how social object data is served*. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's name
authorName

You can use this attribute to include the name of the author of the social object in the design of a social list.

Author's object ID
authorObjectID

This attribute references the serialized object ID of the author of the social object. In contrast to the `authorID` attribute, the `authorObjectID` attribute represents an ID that is used by WebSphere Portal Express rather than by Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's UID
authorUID

This attribute specifies the value of the UID attribute of the author of the social object. This value reflects the UID user attribute as defined by the user repository of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Body HTML
body HTML

This attribute specifies the HTML content of this item.

Body plain
body plain

This attribute specifies the plain content of this item.

Body plain with new line replacement
bodyNewLine2BR

This attribute specifies the plain content of the item. In contrast to the `body plain` attribute, all new line characters are replaced by the appropriate HTML representation. For example the content might contain a line break:

```
Today is
a nice day.
```

The line break is replaced by the HTML line brake tag:

```
Today is<br>a nice day.
```

This attribute is computed by the social rendering bean list provider plug-in.

Body content type
bodyContentType

This attribute specifies the content type identifier for the body attribute of this item.

Can delete comment
canDeleteComment
If the current user has permission to delete the comment, this attribute returns the string `true`. Otherwise, it returns `false`. This attribute is computed by the social rendering Digital Data Connector plug-in.

Comment UUID
commentUUID
This attribute specifies the UUID of this wiki page comment.

Feed entry
entry The original feed entry of this item.

ID
id The unique identifier for this item.

Permissions
permissions
This attribute specifies the permissions that the current user was given on the wiki page comment.

Author's portal image URL
portalAuthorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL from which users can download the image from Connections by using the Ajax proxy of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's portal URL
portalAuthorLink
This attribute specifies the link for rendering the details view of the author of this item in the context of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Published date
published
This attribute indicates the time when the social object was published. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Author's raw Atom entry URL
rawAuthorEntryLink
Link to the Atom entry that represents the author of this item. This URL directly addresses the Connections server. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's raw image URL
rawAuthorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL for downloading the image directly from Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Raw edit URL
rawEditLink
This attribute specifies the Connections URL of the editable resource that represents this wiki page comment.

Raw edit media URL
rawEditMediaLink
This attribute specifies the Connections URL of the editable media resource that represents this wiki page comment.

Raw Atom entry URL
rawEntryLink
This attribute specifies the Connections URL of the Atom resource that represents this wiki page comment.

Raw view URL
rawLink
This attribute specifies the Connections URL of the details view of this wiki page comment.

Summary
summary
A summary of the content of this item.

Title
title The title of this item.

Updated date
updated
This attribute indicates the time of the last update of the social object. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Version number
versionNumber
This attribute specifies the version number of this wiki page comment.

In addition to these attributes, the wiki page comments profile also provides access to the following list properties:

Updated date
updated
This list property indicates the time of the last update of the list that is represented by the feed at hand. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Raw URL to this feed
selfLink
This list property specifies the link to this feed.

Feed title
title This list property specifies the link to this feed.

Total Number of Items
totalNumberOfItems
This list property specifies the total number of items that were found for this search.

Requested Number of Items
requestedNumberOfItems
This list property specifies the requested number of items for this search.

Start Index
startIndex
This list property specifies the start index for the paged list of items.

Raw first URL
rawFirstLink
This list property specifies the Connections URL of the first page of items that were found for this search.

Raw next URL
rawNextLink
This list property specifies the Connections URL of the next page of items that were found for this search.

Raw previous URL
rawPreviousLink
This list property specifies the Connections URL of the previous page of items that were found for this search.

Raw last URL
rawLastLink
This list property specifies the Connections URL of the last page of items that are found for this search.

Is truncated
isTruncated
If the list that represents a search result is truncated, this list property returns the string `true`. Otherwise, it returns `false`.

Wiki page attachments profile

The wiki page attachments profile provides access to Connections wiki page attachments feed data. It declares the following attribute names:

Author's email address
authorEmail
This attribute references the email address of the author of the social object.

Author's ID
authorID
This attribute references the internal ID of the author of the social object. It represents the author ID from the Connections server where the social object is stored.

Author's image URL
authorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL from which users can download the image either directly from Connections or by using the Ajax proxy of the portal. For more information about this attribute, read *Configuring globally how social object data is served*. This attribute is computed by the social rendering bean list provider plug-in.

Author's name
authorName
You can use this attribute to include the name of the author of the social object in the design of a social list.

Author's object ID
authorObjectID
This attribute references the serialized object ID of the author of the social object. In contrast to the `authorID` attribute, the `authorObjectID` attribute

represents an ID that is used by WebSphere Portal Express rather than by Connections. This attribute is computed by the social rendering bean list provider plug-in.

Author's UID
authorUID
This attribute specifies the value of the UID attribute of the author of the social object. This value reflects the UID user attribute as defined by the user repository of the portal. This attribute is computed by the social rendering bean list provider plug-in.

Body HTML
body This attribute specifies the HTML content of this item.

Body content type
bodyContentType
This attribute specifies the content type identifier for the `body` attribute of this item.

Body plain
bodyPlain
This attribute specifies the plain content of this item.

Attachment's size
byteSize
This attribute specifies the size of the attachment in bytes.

Download URL
downloadLink
This attribute represents a link to this attachment. This attribute contains a URL from which users can download the attachment either directly from Connections or by using the Ajax proxy of the portal. If the download link is not available, this attribute does not have a value. For more information about this attribute, read *Configuring globally how social object data is served*. This attribute is computed by the social rendering bean list provider plug-in.

Feed entry
entry This attribute specifies the original feed entry of this item.

File extension
fileExtension
This attribute carries the file extension of this attachment. For example, the file extension can be `.jpg`, `.mp3`, `.pdf`, or empty. This attribute is computed by the social rendering bean list provider plug-in.

File type
fileType
This attribute specifies the file type of this attachment. For example, the file type can be audio, code, compressed, contact, data, flash, graphic, pdf, presentation, text, video, wordProcessing, or dataDocs. The portal sets this attribute according to the value of the `fileExtension` attribute. The `fileType` attribute value is derived from the configured file type extension mappings of the WP Connections Integration Service resource environment provider. For more information, read *Configuring file type icon mappings*. This attribute is computed by the social rendering bean list provider plug-in.

ID
id This attribute specifies the unique identifier for this attachment.

Mime type
mimeType
This attribute specifies the mime type of this attachment.

Attachment's name
name This attribute specifies the name of this attachment.

Author's portal image URL
portalAuthorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL from which users can download the image from Connections by using the Ajax proxy of the portal. This attribute is computed by the social rendering bean list provider plug-in.

Author's portal URL
portalAuthorLink
This attribute specifies the link for rendering the details view of the author of this item in the context of the portal. This attribute is computed by the social rendering bean list provider plug-in.

Portal download URL
portalDownloadLink
This attribute represents a link to this attachment. This attribute contains a URL from which users can download the attachment by using the Ajax proxy of the portal. To download the attachment from Connections without using the Ajax proxy, use the `rawDownloadLink` attribute instead. If the download link is not available, this attribute does not have a value. This attribute is computed by the social rendering bean list provider plug-in.

Published date
published
This attribute indicates the time when the social object was published. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Author's raw Atom entry URL
rawAuthorEntryLink
This attribute specifies the link to the Atom entry that represents the author of this item. This URL directly addresses the Connections server. This attribute is computed by the social rendering bean list provider plug-in.

Author's raw image URL
rawAuthorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL for downloading the image directly from Connections. This attribute is computed by the social rendering bean list provider plug-in.

Raw download URL
rawDownloadLink
This attribute represents a link to this attachment. This attribute contains a URL from which users can download the attachment directly from Connections. To download the attachment from Connections by using the Ajax proxy of the portal, use the `portalDownloadLink` attribute instead. If the download link is not available, this attribute does not have a value.

Raw Atom entry URL
rawEntryLink
This attribute specifies the Connections URL of the Atom entry that represents this attachment.

Summary
summary
This attribute specifies a summary of the content of this social object.

Title
title This attribute specifies the title of this social object.

Updated date
updated
This attribute indicates the time of the last update of the social object. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

In addition to these attributes, the wiki page attachments profile also provides access to the following list properties:

Updated date
updated
This list property indicates the time of the last update of the list that is represented by the feed at hand. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Raw URL to this feed
selfLink
This list property specifies the link to this feed.

Feed title
title This list property specifies the link to this feed.

Total Number of Items
totalNumberOfItems
This list property specifies the total number of items that were found for this search.

Requested Number of Items
requestedNumberOfItems
This list property specifies the requested number of items for this search.

Start Index
startIndex
This list property specifies the start index for the paged list of items.

Raw first URL
rawFirstLink
This list property specifies the Connections URL of the first page of items that were found for this search.

Raw next URL
rawNextLink
This list property specifies the Connections URL of the next page of items that were found for this search.

Raw previous URL
rawPreviousLink
This list property specifies the Connections URL of the previous page of items that were found for this search.

Raw last URL
rawLastLink
This list property specifies the Connections URL of the last page of items that are found for this search.

Is truncated
isTruncated
If the list that represents a search result is truncated, this list property returns the string `true`. Otherwise, it returns `false`.

Related tasks:

"Configuring file type icon mappings" on page 2129
Social rendering provides two types of list appearance components for result lists: `simple` and `comprehensive`. Both list appearances components show file type-specific icons when they render list entries that refer to individual files. In this case, the file type is determined based on the file extensions of the individual files. You can configure the set of file types that you want to use in the WP Connections Integration Service resource environment provider. In the context of social lists, a file type is defined by a file type name and a list of file extensions. This list defines the mapping between individual files and your file types. In your social list appearance components, you can then access the file type name for a specific file by using the Web Content Manager `[AttributeResource attributeName="fileType"]` tag. That tag is defined in the IBM Digital Data Connector (DDC) for WebSphere Portal Express profile. You can then use the file type name to render the appropriate image, for example, by assigning a corresponding CSS class.

"Configuring globally how social object data is served" on page 2125
Among other features, social objects have different links that enable users to download related data. For example, users can use links to download a file or the profile image of the author. You can globally configure how social object data is served to the users. To do so, you use a setting in the WP Connections Integration Service resource environment provider in the WebSphere Integrated Solutions Console.

Common profile:

The common profile provides general aspects that are available in various feeds that are served by IBM Connections. You can use the common profile when you create your own custom profiles.

Note: You cannot use the common profile as is. Therefore, the common profile is not available to the Tag Helper.
For more information about how to extend this profile, read *XPath list-rendering profile keys*.

The common profile defines the following attributes:

Author's email address
authorEmail
This attribute references the email address of the author of the social object.

Author's ID
authorID
This attribute references the internal ID of the author of the social object. It represents the author ID from the Connections server where the social object is stored.

Author's image URL
authorImageLink

This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL where you can download the image either directly from Connections or by using the Ajax proxy of the portal. This attribute is computed by the social rendering IBM Digital Data Connector (DDC) for WebSphere Portal Express plug-in. For more information about this attribute, read *Configuring globally how social object data is served*.

Author's name
authorName

You can use this attribute to include the name of the author of the social object in the design of a social list.

Author's object ID
authorObjectID

This attribute references the serialized object ID of the author of the social object. In contrast to the **authorID** attribute, the **authorObjectID** attribute represents an ID that is used by IBM WebSphere Portal Express rather than by Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's UID
authorUID

This attribute specifies the value of the UID attribute of the author of the social object. This value reflects the UID user attribute as defined by the user repository of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Body HTML
body HTML

The HTML content of this item.

Body plain
body plain

The plain content of this item.

Body content type
bodyContentType

This attribute specifies the content type identifier for the body attribute of this item.

Feed entry
entry The original feed entry of this item.

ID
id The unique identifier for this item.

Author's portal image URL
portalAuthorImageLink

This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL where you can download the image from Connections by using the Ajax proxy of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's portal URL
portalAuthorLink
This attribute specifies the link for rendering the details view of the author of this item in the context of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Published date
published
This attribute indicates the time when the social object was published. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Author's raw Atom entry URL
rawAuthorEntryLink
Link to the Atom entry that represents the author of this item. This URL directly addresses the Connections server. This attribute is computed by the social rendering Digital Data Connector plug-in.

Author's raw image URL
rawAuthorImageLink
This attribute specifies the link to the profile image of the author of the social object. The Connections server that provides the social object also stores the profile image of the author. This attribute contains a URL for downloading the image directly from Connections. This attribute is computed by the social rendering Digital Data Connector plug-in.

Summary
summary
A summary of the content of this item.

Title
title The title of this item.

Updated date
updated
This attribute indicates the time of the last update of the social object. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

In addition to these attributes, the common profile also provides access to the following list properties:

Updated date
updated
This list property indicates the time of the last update of the list that is represented by the feed at hand. To display the date in the format of your choice, you can use all date format options that Web Content Manager provides.

Raw URL to this feed
selfLink
This list property specifies the link to this feed.

Feed title
title This list property specifies the link to this feed.

Related tasks:

"Configuring globally how social object data is served" on page 2125
Among other features, social objects have different links that enable users to download related data. For example, users can use links to download a file or the profile image of the author. You can globally configure how social object data is served to the users. To do so, you use a setting in the WP Connections Integration Service resource environment provider in the WebSphere Integrated Solutions Console.

Related reference:

"XPath list-rendering profile keys" on page 3277
The following list shows the set of list-rendering profile entry keys that are available in the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework.

Modifier support profile:

The modifier support profile provides general modifier aspects. These modifier aspects are available in various feeds that are served by IBM Connections. You can reuse them when you create your own profiles.

In contrast to the author information described in the *Common profile*, contributors of additional information to a social object who are not the initial authors are regarded as modifiers. As not all social objects have additional modifications, the modifier support profile is separate from the common profile.

The modifier support profile declares the following attribute names:

Modifier's email address
`modifierEmail`
: This attribute references the email address of the modifier of the social object.

Modifier's ID
`modifierID`
: This attribute references the internal ID of the modifier of the social object.

Modifier's image URL
`modifierImageLink`
: This attribute specifies the link to the profile image of the modifier of the social object. The Connections server that provides the social object also stores the profile image of the modifier. This attribute contains a URL from which users can download the image either directly from Connections or by using the Ajax proxy of IBM WebSphere Portal Express. For more information about this attribute, read *Configuring globally how social object data is served*. This attribute is computed by the social rendering bean list provider plug-in.

Modifier's name
`modifierName`
: You can use this attribute to include the name of the modifier of the social object in the design of a social list.

Modifier's object ID
`modifierObjectID`
: This attribute references the serialized object ID of the modifier of the social object. In contrast to the `modifierID` attribute, the `modifierObjectID` attribute represents an ID that is used by WebSphere Portal Express rather than by Connections. This attribute is computed by the social rendering bean list provider plug-in.

Modifier's UID
modifierUID
This attribute specifies the value of the UID attribute of the modifier of the social object. This value reflects the UID user attribute that is defined by the user repository of the portal. This attribute is computed by the social rendering bean list provider plug-in.

Modifier's raw Atom entry URL
rawModifierEntryLink
This attribute specifies the link to the Atom entry that represents the modifier of this item. This URL addresses the Connections server directly. This attribute is computed by the social rendering bean list provider plug-in.

Modifier's raw image URL
rawModifierImageLink
This attribute specifies the link to the profile image of the modifier of the social object. The Connections server that provides the social object also stores the profile image of the modifier. This attribute contains a URL for downloading the image directly from Connections. This attribute is computed by the social rendering bean list provider plug-in.

Modifier's portal image URL
portalModifierImageLink
This attribute specifies the link to the profile image of the modifier of the social object. The Connections server that provides the social object also stores the profile image of the modifier. This attribute contains a URL where you can download the image from Connections by using the Ajax proxy of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Modifier's portal URL
portalModifierLink
This attribute specifies the link for rendering the details view of the modifier of this item in the context of the portal. This attribute is computed by the social rendering Digital Data Connector plug-in.

Related tasks:

"Configuring globally how social object data is served" on page 2125
Among other features, social objects have different links that enable users to download related data. For example, users can use links to download a file or the profile image of the author. You can globally configure how social object data is served to the users. To do so, you use a setting in the WP Connections Integration Service resource environment provider in the WebSphere Integrated Solutions Console.

Tag support profile:

The tag support profile provides access to a list of tags that is associated with the social objects served by IBM Connections feed data.

It declares the following attribute name:

Tags
tags This attribute specifies the tags that are associated with this item.

OpenSearch support profile:

The OpenSearch support profile provides access to information that is defined in the OpenSearch specification. The OpenSearch specification is commonly used in the Atom feed format.

Specifically, this profile supports list properties that are required to make paging possible. Paging becomes necessary if the backend service does not return all available data records in a single feed and provides links for retrieving the next page of data records.

The OpenSearch support profile declares the following attribute names:

Total Number of Items
`totalNumberOfItems`
: This list property specifies the total number of items that were found for this search.

Requested Number of Items
`requestedNumberOfItems`
: This list property specifies the requested number of items for this search.

Start Index
`startIndex`
: This list property specifies the start index for the paged list of items.

Raw first URL
`rawFirstLink`
: This list property specifies the Connections URL of the first page of items that were found for this search.

Raw next URL
`rawNextLink`
: This list property specifies the Connections URL of the next page of items that were found for this search.

Raw previous URL
`rawPreviousLink`
: This list property specifies the Connections URL of the previous page of items that were found for this search.

Raw last URL
`rawLastLink`
: This list property specifies the Connections URL of the last page of items that are found for this search.

Is truncated
`isTruncated`
: If the list that represents a search result is truncated, this list property returns the string `true`. Otherwise, it returns `false`.

Using the business card:

You can integrate the business card and online status in a list of social objects by using live text micro format.

About this task

You can find a sample HTML component that includes this micro format in the IBM Connections library at the following location:

Social Lists 1.0 > **Components** > **Fragments** > **Business Card**. This HTML component is referenced by the default **Comprehensive** list appearance option for social objects that you can select when you customize social list definitions.

The business card sample identifies a user by the `authorObjectID` attribute of a social object that is returned by Connections.

If your IBM WebSphere Portal Express is configured to show the Connections business card information, use the `authorID` attribute instead of `authorObjectID` . For more information, read *Integrating with Connections*.

The following sample shows how you can map the different social object attributes to the microformat class attributes. This microformat works with both the WebSphere Portal Express business card and with the Connections business card. This means that the sample works, independent of whether WebSphere Portal Express is configured to show the portal business card or the Connections business card.

```
<span class='vcard X-sametime-resolve'>
   <span class="lotusMeta">
      <a target="" title="" href="javascript:SemTagMenu.a11y(event)"
                  class="fn lotusPerson" style="text-decoration:none;"
                  onclick="return false;">
                  [AttributeResource attributeName="authorName" separator=","]
      </a>
   </span>
   <span style="display: none;"
         class="userObjectId">[AttributeResource attributeName="authorObjectID"
         separator=","]</span>
   <span style="display: none;"
         class="x-lconn-userid">[AttributeResource attributeName="authorID"
         separator=","]</span>
</span>
```

For more information, read *Integrating the business card and online status in a custom portlet*.

Notes:

- CF03 If you integrate an Connections server that runs in the Smart Cloud for Social Business, support for the business card is currently not available. The Business Card HTML component that is contained in the Social Lists 1.0 library does not generate the business card microformat in this case. You can check whether you integrate an Connections server that runs in the Smart Cloud for social business in your own design components. To do so, use the `[Plugin:ConnectionsContext type="config" key="server.config"]`. For details, read *Configuring global settings for social rendering*.
- If you want to use the business card on a WebSphere Portal Express page, the page editor must add the `portal.livetext.hcard` theme capability to the page. To achieve this, the page editor can add the `wp_liveobject_framework` theme module to the current page profile. For more information, read *The module framework*.

Related concepts:

"The module framework" on page 2526
The module framework allows extensions to contribute to different areas of a page to provide flexibility, enhance the user experience, and maximize performance.

Related tasks:

Integrating with IBM Connections
Connections portlets give IBM WebSphere Portal Express users access to additional collaboration and social networking features such as Activities, Blogs, and Bookmarks. Users can also view Connections business card information and tags and link to Connections features directly from the portal server.

"Integrating the Business card and online status in a custom portlet" on page 3105
If IBM WebSphere Portal Express is configured to work with IBM Sametime, you can integrate the business card and online awareness in a custom portlet. Person names then appear with a dynamic status indicator. Users of the portlet can move the cursor over the name of an active person and then click **Click for Business Card**.

Customizing the CSS styles of social lists:

The lists of social objects rely on the availability of several CSS class definitions in the `wp_social_rendering_85` theme module.

About this task

Pages that render the sample items must use a theme profile that includes the `wp_social_rendering_85` theme module or another module that contains the same CSS class definitions. An example of such a theme profile is the Basic Content theme profile that is installed by default.

The `wp_social_rendering_85` theme module includes a single CSS file that is named `sr.css`. This style sheet is loaded from the following directory location: *`PortalServer_root`*`/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/modules/sr/v8.5/css/sr.css`.

The `wp_social_rendering_85` theme module defines the capability with the name `social_rendering_85` and the version `8.5`. No prerequisites are required to use this theme module.

Style sheet note: The lists of social objects use CSS styles that render differently, depending on the layout container that contains the social list web content viewer. Different classification CSS classes are assigned to these layout containers. As a result, the styles change when a social list web content viewer is moved from the center column into a narrower side column or vice versa. Furthermore, the styles are responsive to screen width and screen orientation. This responsiveness is achieved by using HTML5 and CSS3 media queries.

"Providing custom styles for social lists" on page 2226
The default CSS styles that are used to define the visual appearance of social lists are defined in the `wp_social_rendering_85` theme module. You can define your own styles.

"CSS class hierarchy for social lists" on page 2228
To define its visual appearance, the default markup for social lists uses several CSS classes. They are defined in the `wp_social_rendering_85` theme module. Learn about the purpose of the most important CSS classes.

"Using media queries to target mobile devices" on page 2229
The default social list CSS styles use CSS3 media queries to target specific mobile devices and implement responsive web design. Learn more about media queries.

Providing custom styles for social lists:

The default CSS styles that are used to define the visual appearance of social lists are defined in the `wp_social_rendering_85` theme module. You can define your own styles.

About this task

To create your own styles, proceed by the following steps:

Procedure

1. Create your own custom theme module. For information about how to create a new theme module, see the topic about *Define the module* in the WebSphere Portal Express product documentation.
2. Add an extension point to your theme module that adds CSS styles to the HEAD section of the page. As an example, the `wp_social_rendering_85` theme module uses the following extension point:

```
<extension point="com.ibm.portal.resourceaggregator.module" id="social_rendering_head" >
   <module id="wp_social_rendering_85">
      <capability id="social_rendering_85" value="8.5"/>
      <contribution type="head">
         <sub-contribution type="css">
            <uri value="res:{rep=WP GlobalThemeConfig;key=resources.modules.ibm.contextRoot}/modules/sr/v8.5/css/master_sr.css" />
            <uri type="rtl"
                 value="res:{rep=WP GlobalThemeConfig;key=resources.modules.ibm.contextRoot}/modules/sr/v8.5/css/master_sr_rtl.css"/>
            <uri deviceClass="smartphone"
                 value="res:{rep=WP GlobalThemeConfig;key=resources.modules.ibm.contextRoot}/modules/sr/v8.5/css/master_sr_mobile.css"/>
            <uri deviceClass="tablet"
                 value="res:{rep=WP GlobalThemeConfig;key=resources.modules.ibm.contextRoot}/modules/sr/v8.5/css/master_sr_mobile.css"/>
         </sub-contribution>
      </contribution>
   </module>
</extension>
```

   Make sure to leave the `social_rendering` capability ID and the `8.5` capability version number intact as it is, but change the module ID from `wp_social_rendering_85` to a value of your choice.
3. Make a copy of the theme profile provided with the portal, for example `profile_basic_content.json`. You can use that copied profile on your pages with social lists. If you update the default theme profile provided with the portal, your updates will be overwritten if you install a fix pack or otherwise upgrade your portal.
4. Add the new module ID from a previous step to your copy of the theme profile.
5. Copy the default CSS files from the social rendering theme into the new theme module. These files are in the following directory: *PortalServer_root*`/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/modules/sr/v8.5/css`

   The following CSS files are available:

Table 339. CSS files available in the social rendering theme

CSS file name	Purpose
`sr.css`	This file contains the basic social rendering list styles.
`sr_wptheme.css`	This file contains all styles that depend on `wptheme` sample CSS classes to detect wide, narrow, and thin columns.
`sr_rtl.css`	This file contains the style overrides for bidirectional languages.

Table 339. CSS files available in the social rendering theme (continued)

CSS file name	Purpose
sr_wptheme_rtl.css	This file contains the style overrides for wide, narrow, and thin column definitions in bidirectional languages.
sr_mobile.css	This file contains the style overrides and media queries for mobile devices.

The following table shows the master CSS files. They import the CSS files that are shown in the previous table.

Table 340. Master CSS files of the social rendering theme

CSS file name	Purpose
master_sr.css	This master CSS file links to the basic CSS styles for left-to-right languages.
master_sr_rtl.css	This master CSS file links to the basic CSS styles for bidirectional languages.
master_sr_mobile.css	This master CSS file links to the CSS styles for mobile devices and left-to-right and bidirectional languages.
master_sr_mobile_rtl.css	This master CSS file links to the CSS styles for mobile devices and bidirectional languages

6. Remove wp_social_rendering_85 from your new theme profile, and add the identifier of your new theme module instead. Repeat this process for all profiles that you are using in your website. By default, the wp_social_rendering_85 module is part of the following theme profiles:
   - profile_wab.json
   - profile_search_tag.json
   - profile_dojo_deferred.json
   - profile_dojo_basic_content.json
   - profile_basic_content.json
   - profile_deferred.json

   In the following code section, replace wp_social_rendering_85 by *your_theme_module_id*:

```
"wp_ic4_wai_resources",
 "wp_worklight_ext",
"wp_social_rendering_85",
"wp_sametime_proxy"
      ],
```

7. Save your changes.
8. Optional: If you updated your custom theme profile in a separate environment, copy your updated custom profile to the portal.
9. Optional: If you built your own custom theme and you do not use the wptheme sample CSS classes, make sure to modify the CSS rules in the CSS files sr_wptheme.css and sr_wptheme_rtl.css accordingly.
10. Optional: If you added more CSS files or renamed existing ones, make sure to update the master_*.css files, as they import one or more of the other sr_*.css files and make them available to the theme module.

11. Optional: If you add more `master_*.css` files or change their names, make sure to update the CSS contribution in the `plugin.xml` file of your theme module. See the following sample code snippet:

```
...
   <sub-contribution type="css">
      <uri value="res:{rep=WP GlobalThemeConfig;key=resources.modules.ibm.contextRoot}/modules/sr/v8.5/css/master_sr.css" />
      <uri type="rtl"
           value="res:{rep=WP GlobalThemeConfig;key=resources.modules.ibm.contextRoot}/modules/sr/v8.5/css/master_sr_rtl.css"/>
      <uri deviceClass="smartphone"
           value="res:{rep=WP GlobalThemeConfig;key=resources.modules.ibm.contextRoot}/modules/sr/v8.5/css/master_sr_mobile.css"/>
      <uri deviceClass="tablet"
           value="res:{rep=WP GlobalThemeConfig;key=resources.modules.ibm.contextRoot}/modules/sr/v8.5/css/master_sr_mobile.css"/>
      <uri ...additional contributions go here.../>
   </sub-contribution>
...
```

12. Optional: If you use social rendering with your own theme, modify the profiles of your custom theme accordingly.
13. Modify the CSS files that you copied to the new theme module in an earlier step as required. After you modified the CSS files, invalidate browser cache for your updates to take effect. To invalidate your cache, click **Theme Analyzer** > **Utilities** > **Control Center** > **Invalidate Cache**. CF06 Auto invalidation recognizes your changes automatically for WebDAV based themes. No further action is required. For more information, see Utilities.

Results

After you complete this procedure, you can use your own custom styles for your social lists.

Related tasks:

"Defining theme modules" on page 2547
You can define theme modules in XML or JSON.

CSS class hierarchy for social lists:

To define its visual appearance, the default markup for social lists uses several CSS classes. They are defined in the `wp_social_rendering_85` theme module. Learn about the purpose of the most important CSS classes.

Table 341. Social rendering CSS classes and their purposes

CSS class	Purpose
`srComponent`	This class is a top-level wrapper class that is set around the whole list.
`srSimple`	This class is a wrapper class that is set around the whole list in simple design within an `srComponent` container.
`srComprehensive`	This class is a wrapper class that is set around the whole list in comprehensive design within an `srComponent` container.
`srTable`	This class is set on the table that contains the list entries as row elements.
`srLeftColumn`	This class is set on the table cell that contains the entry icon.
`srMainColumn`	This class is set on the table cell that contains the entry title and metadata.
`srCountIcons`	This class is set on list items for the Likes and Comments icon.
`srLikeIcon`	This class is set on list item for the number of Likes.

Table 341. Social rendering CSS classes and their purposes (continued)

CSS class	Purpose
srCommentIcon	This class is set on list item for the number of Comments.
srShareIcon	This class is set on list item for access control status.
srMetaData	This class is a wrapper class that is set on container for date and tags metadata.
srMetaDate	This class is set on the date container of the entry.
srMetaTags	This class is set on the tags container of the entry.
srSummary	This class is set on the summary container of the entry.
srSocialBar	This class is set on the Social Bar component in each entry.
srRepliesList	This class is set on the outside container of the list of replies.
srViewSocialObject	This class is set on the part of a Social Bar that contains the View link.
srReply*	All classes that are used for styling replies in the forum topic details view are prefixed with srReply.

By using the CSS classes shown in this table, you can build specific rules to style every aspect of a social list entry individually. In the sr_wptheme.css style file, you can find a number of CSS rules that are used as overrides for basic CSS definitions on narrow and thin columns. To identify a narrow or thin column in the WebSphere Portal Express default Portal 8.5 theme, use CSS rules like the following one:

```
.wpthemeThin   .lotusui30 .srComponent .srSimple .srEntryTitle,
.wpthemeNarrow .lotusui30 .srComponent .srSimple .srEntryTitle {
    font-size: 0.63em;
}
```

Columns with the wpthemeThin CSS rule assigned to them are even smaller than columns with wpthemeNarrow assigned to them. If you want to define separate styles for narrow and thin columns, you can use the following CSS rules. The following code sample sets a smaller font size for the entry title on narrow columns and an even smaller font size for thin columns:

```
.wpthemeNarrow .lotusui30 .srComponent .srSimple .srEntryTitle {
    font-size: 0.63em;
}

.wpthemeThin .lotusui30 .srComponent .srSimple .srEntryTitle {
    font-size: 0.58em;
}
```

Using media queries to target mobile devices:

The default social list CSS styles use CSS3 media queries to target specific mobile devices and implement responsive web design. Learn more about media queries.

About this task

The media query that social rendering uses has the following format:

```
@media screen and (max-width: $valuepx) {
   ....
}
```

where `$value` is an integer value that defines the maximum screen width in pixels. The rules within this query block do not apply to screens with larger screen widths than specified. To define responsive styles, you can use multiple queries of this type with decreasing maximum screen widths. For example, you can start hiding elements within your markup on a tablet in portrait mode. You can then continue hiding even more elements on a smartphone in landscape mode, and show only a minimum data volume on smartphones in portrait mode.

Example

The default social list markup defines a social bar for each entry type, which takes up vertical space. To save vertical space on mobile devices, you can hide either the social bar View link or the complete social bar. In the following sample, you want to set up the following space arrangements for display on smartphones:

- In smartphone landscape mode, you want to hide the View link in the social bar of the social object.
- In smartphone portrait mode, you want to hide the social bar. This way, you reserve the complete vertical space for showing only the title and summary of the social object.

To achieve this reduction, you can use CSS3 media queries as shown herein the following code sample:

```
@media screen and (max-width: 480px) {
    .lotusui30 .srComponent .srViewSocialObject {
        display: none;
        width: 0px;
    }
}

@media screen and (max-width: 320px) {
    .lotusui30 .srComponent div.srSocialBar {
        display: none;
    }
}
```

The media queries in this sample work as follows:

1. The first media query makes sure that the View link in the social bar is hidden on displays that are 480 pixels or narrower.
2. The second media query makes sure that the social object social bar is hidden completely on displays that are 320 pixels wide or narrower.

For displays that are 321 to 480 pixels wide, only the first query applies. For screens that are 320 pixels wide or narrower, both queries apply.

Creating custom authoring templates for list definitions

Social rendering provides you a set of view definitions that you can use to add social data to your portal pages. These list view definition content items are created from the social list definition IBM Web Content Manager authoring template. You can create your own custom list view definitions by creating new

content items from this authoring template. To customize social rendering even further, you can also create your own authoring templates to extend the data set that makes up your list view definitions.

About this task

Your presentation components can then use the additional data that is provided by your custom authoring templates to control the rendering behavior for the lists. For example, you can add an option to have the page editor control whether the paging component for the social list is rendered or not. Social rendering can render content items that are created by such an extended authoring template in the same way as the default social lists do.

If you create a custom authoring template, make sure that it contains all the elements that are recognized by the `ListRenderingContext` rendering plug-in. This way you ensure that the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework can establish the appropriate query context when retrieving data from the remote IBM Connections server. To create a custom authoring template, proceed by the following steps:

Procedure

1. Copy the Social List Definition authoring template.
2. Keep all elements that the template already contains.
3. Add new elements as required.
4. Optional: You can also change the default for the existing elements or modify the authoring template user interface for those elements.

Implementing interactions with social objects

Site designers can implement design components that suppor interactions between the user and the social data. For example, a user can post a new reply to a forum topic or delete a previous reply.

About this task

To create such interactive designs, site designers need to implement an HTML form to send a POST action request to the Web Content Viewer portlet that displays the social list. The HTML form must include the URI of a data sink and parameters that identify the action that the designer wants to run. The portlet then dispatches to the target data sink that processes the form data and returns a data source that holds the result. For example, the result can be a JSON object that contains the status of the operation that is triggered with the POST action request and an error or success message. The portlet stores the result in the portlet session or as a private render parameter. To access the result in the formatting components, site designers can use the `SessionAttribute` rendering plug-in and the `RenderParam` rendering plug-in.

For more information about the action URL, render parameter, and session attribute rendering plug-ins, read *Utility rendering plug-ins*.

The following topics describe how you implement user interactions that are related to forum topics by using the social-rendering specific data sink. Additionally, you can implement further user interaction by using the generic XML Digital Data Connector data sink and the social rendering list-rendering profiles. For more information, read *The generic XML Digital Data Connector data sink* and *Digital Data Connector profiles for social rendering*.

"Interacting with forums"
WebSphere Portal Express provides a built-in data sink that supports different interactions with forums.

"Creating a reply" on page 2233
When you run a `createReply` action, the forums data sink uses a number of extra form fields.

"Deleting a reply" on page 2234
When you run a `deleteReply` action, the forums data sink uses a number of extra form fields.

Related concepts:

"Sending data to the Web Content Viewer portlet" on page 3260
You can use IBM Web Content Manager presentation components to create user interfaces for creating, modifying, and deleting external data. With this approach, you use Web Content Manager design components to generate specific HTML form markup.

"The generic XML Digital Data Connector data sink" on page 3263
The IBM Digital Data Connector (DDC) for WebSphere Portal Express framework contains a generic data sink. You can use the data sink in combination with HTML forms generated by your IBM Web Content Manager design components.

Related reference:

"The social rendering Digital Data Connector plug-in" on page 2237
To implement the IBM Connections data integration, the social rendering feature in IBM WebSphere Portal Express Version 8.5 uses a custom IBM Digital Data Connector (DDC) for WebSphere Portal Express plug-in.

"Digital Data Connector profiles for social rendering" on page 2147
Starting with Version 8.5, IBM WebSphere Portal Express includes a set of IBM Digital Data Connector (DDC) for WebSphere Portal Express profiles. You can use them with the social rendering DDC plug-in that is identified by the extension ID `ibm.portal.ddc.sr`.

Interacting with forums:

WebSphere Portal Express provides a built-in data sink that supports different interactions with forums.

About this task

To implement such an interaction, the HTML form that targets the Web Content Viewer portlet must include all of the following:

- The URI of the forums data sink. The URI must be specified in the `action.uri` form field
- The name of the action that you want to have performed
- Extra input data that is required for the specified action. The URI of the forums data sink must contain the identifier of the target social object, for example, a forum topic or a forum topic reply. You must add the identifier to the scheme-specific part of the URI after a separating colon, `sr:forums:$REPLY_ID`. The forums data sink also expects the `action` form field to be submitted along with the form POST request. It specifies the action that you want to have performed on the target resource that is identified by the URI. The data sink supports the following actions:

  `createReply:`
  The forums data sink creates a new reply to the target social object, which must be either a forum topic or another forum topic reply.

deleteReply
The forums data sink deletes the target social object, which must be reply.

Depending on the action, the data sink requires extra form data.

Note: The update operation for individual replies that Forum Topic Details appearance component supports is implemented as a post to *The generic XML Digital Data Connector data sink*.

Related concepts:

"The generic XML Digital Data Connector data sink" on page 3263
The IBM Digital Data Connector (DDC) for WebSphere Portal Express framework contains a generic data sink. You can use the data sink in combination with HTML forms generated by your IBM Web Content Manager design components.

Creating a reply:

When you run a `createReply` action, the forums data sink uses a number of extra form fields.

About this task

They are listed here:

parentEntryLink
Specifies the raw Atom entry link of the forum topic or forum topic reply to post a reply to. This link must point directly to the Connections server rather than to the Ajax proxy of the portal.

parentRepliesLink
Specifies the raw replies link of the forum topic or forum topic reply to post a reply to. This link must point directly to the Connections server rather than to the Ajax proxy of the portal. -

replyTitle
Specifies the optional title of the reply to create.

replyContent
Specifies the content of the reply to create.

The following code snippet shows a sample form for creating a reply. It uses the `AttributeResource` tag of Web Content Manager to set the ID, the entry link, and the replies link of the parent social object in the context of a social list.

```
<form method="POST"
    enctype="multipart/form-data" action="[Plugin:ActionURL action="post"
    param="resultSessionAttribute=myResult" param="resultRenderParameter=myResult"
        compute="always"]">
    <input type="hidden" name="_charset_"
           value="[Plugin:EvaluateEL value="${pageContext.response.characterEncoding}"
           compute="once"]"/>
    <input type="hidden" name="action.uri"
           value="forums:sr:[AttributeResource attributeName="id"]"/>
    <input type="hidden" name="action" value="createReply"/>
    <input type="hidden" name="parentEntryLink"
           value="[AttributeResource attributeName="rawEntryLink"]"/>
    <input type="hidden" name="parentRepliesLink"
           value="[AttributeResource attributeName="rawRepliesLink"]"/>
  Title:
    <input type="text" name="replyTitle"/><br/>
    <textarea name="replyContent" rows="3" cols="25">Please enter your reply.</textarea><br/>
    <input type="submit" value="Post"/>
</form>
```

If the createReply action is successful, the forums data sink returns a JSON object. Example:

```
{
  "status":"success",
  "message":"Your reply has been created.",
  "formData":{
      "action.uri":"forums:sr:8fd51c01-6505-4d78-b364-415edf649e91",
      "action":"createReply",
      "parentEntryLink":"https://.../forums/atom/topic?topicUuid=8fd51c01-6505-4d78-b364-415edf649e91",
      "parentRepliesLink":"https://.../forums/atom/replies?topicUuid=8fd51c01-6505-4d78-b364-415edf649e91"
  },
  "resultData":{
      "replyID":"urn:lsid:ibm.com:forum:815b1a6d-5a3c-4730-91d2-094075ff2e5d"
  }
}
```

If the createReply action fails, the forums data sink returns a JSON object as seen in the following sample:

```
{
  "status":"error",
  "message":"Your reply could not be created. If the problem persists, contact the system administrator.",
  "formData":{
      "action.uri":"forums:sr:8fd51c01-6505-4d78-b364-415edf649e91",
      "action":"createReply",
      "parentEntryLink":"https://.../forums/atom/topic?topicUuid=8fd51c01-6505-4d78-b364-415edf649e91",
      "parentRepliesLink":"https://.../forums/atom/replies?topicUuid=8fd51c01-6505-4d78-b364-415edf649e91",
      "replyTitle":"My Forum Topic Reply",
      "replyContent":"This is my reply to the forum topic."
  }
}
```

Note that the formData property contains the actual form input, which includes all additional form data that you submit. If the action is successful, the replyTitle and replyContent are not returned to the caller.

Deleting a reply:

When you run a deleteReply action, the forums data sink uses a number of extra form fields.

About this task

They are listed here:

replyEntryLink
Specifies the raw Atom entry link of the forum topic reply that you want to delete. This link must point directly to the Connections server rather than to the Ajax proxy of the portal.

deleteReason
Specifies the reason for deleting the reply. This field is optional.

The following code snippet shows a sample form for deleting a reply. It uses the AttributeResource tag of Web Content Manager to set the ID and the entry link of the reply in the context of a social list that contains forum topic replies:

```
<form method="POST" enctype="multipart/form-data"
    action="[Plugin:ActionURL action="post"
    param="resultSessionAttribute=myResult"
    param="resultRenderParameter=myResult" compute="always"]">
    <input type="hidden" name="_charset_" value="[Plugin:EvaluateEL
           value="${pageContext.response.characterEncoding}" compute="once"]"/>
    <input type="hidden" name="action.uri"
           value="forums:sr:[AttributeResource attributeName="id"]"/>
    <input type="hidden" name="action" value="deleteReply"/>
    <input type="hidden" name="replyEntryLink"
           value="[AttributeResource attributeName="rawEntryLink"]"/>
```

```
  Reason:
    <input type="text" name="replyDeleteReason"/><br/>
    <input type="submit" value="Post"/>
</form>
```

If the `deleteReply` action is successful, the forums data sink returns a JSON object. Example:

```
{
  "status":"success",
   "message":"The reply has been deleted.",
   "formData":{
      "action.uri":"forums:sr:urn:lsid:ibm.com:forum:815b1a6d-5a3c-4730-91d2-094075ff2e5d",
      "action":"deleteReply",
      "replyEntryLink":"https://.../forums/atom/reply?replyUuid=815b1a6d-5a3c-4730-91d2-094075ff2e5d"
  }
}
```

If the `deleteReply` action fails, the forums data sink returns a JSON object as seen in the following sample:

```
{
   "status":"error",
   "message":"The reply could not be deleted. If the problem persists, contact the system administrator.",
   "formData":{
      "action.uri":"forums:sr:urn:lsid:ibm.com:forum:815b1a6d-5a3c-4730-91d2-094075ff2e5d",
      "action":"deleteReply",
      "replyEntryLink":"https://.../forums/atom/reply?replyUuid=815b1a6d-5a3c-4730-91d2-094075ff2e5d",
      "replyDeleteReason":"This is my reason for deleting the reply."
  }
}
```

The `formData` property contains the actual form input, which includes all additional form data that you submit. If the action is successful, the `replyDeleteReason` is not returned to the caller.

Adding widgets to a community

As an owner of an IBM Connections community, you can define the set of widgets that are available in the community. For this purpose, you use the Customize option in Connections. For example, you might choose to add the Blog widget to a specific community. This way, community owners have control over the set of services available in the communities they own.

About this task

When you use portal pages as the front end for communities, the pages typically contain social content that interact with one or more widgets of the community. For example, the List of Community Blog Posts shows blog posts that were created by using the Blog widget. You can flag such social content items with Drag and Drop configurations to depend on specific sets of widgets that are available in the associated community. You set this flag by setting a specific portlet preference on the portlet entity of the associated Drag and Drop configuration. Whenever a page editor adds such portlet to a community page, the portal infrastructure tries to dynamically add the required widgets to the associated community.

To flag a drag and drop configuration to require a specific set of widgets, you set the following preference on the portlet entity:

`ibm.portal.instantiation.community.widgets=`*`comma separated list of widget definition IDs`*

You set this preference by using the Manage Portlets portlet, or the XML configuration interface (XMLAccess).

The following list shows the supported widget definition IDs

- `StatusUpdates`

- Forum
- Bookmarks
- Files
- Blog
- IdeationBlog
- Activities
- Wiki
- Calendar
- MediaGallery
- Feeds
- SubcommunityNav
- RelatedCommunities

Example: To flag a drag and drop configuration to require the Blog and Wiki widgets, set the preference as follows:

```
ibm.portal.instantiation.community.widgets=Blog,Wiki
```

By default, the portal tries a best effort to add the widget. If the operation fails for whatever reason, the portal still adds the social content item to the page, but it does not add the widget to the community. If you require a different behavior, you can set an additional portlet preference to generate an error message instead. To flag a drag and drop configuration to generate an error message if the required widgets cannot be added to the associated community, set the following portlet preference:

```
ibm.portal.instantiation.community.widgets.failonerror=[true|false]
```

The default value is `false`. You set this preference by using the Manage Portlets portlet, or the XML configuration interface (XMLAccess). With the `failonerror` portlet preference enabled, the portal does not add the social content item to the page, but shows an error message instead.

The drag-and-drop configurations for the community-focused list view definitions have the setting `failonerror` set to the value `false`. The following list shows these configurations for your reference:

- List of Community Forum Topics:
  `ibm.portal.instantiation.community.widgets=Forum`
- List of Community Blog Posts:
  `ibm.portal.instantiation.community.widgets=Blog`
- List of Community Events:
  `ibm.portal.instantiation.community.widgets=Calendar`
- List of Community Content: `ibm.portal.instantiation.community.widgets=Forum,Bookmarks,Files,Blog,Activities,Wiki,Calendar`
- List of Community Files: `ibm.portal.instantiation.community.widgets=Files`

Note: The portlet preferences that are described here are ignored on non-community pages, that is on pages that are not associated to a specific community.

Extending social lists by using the digital data connector

The social rendering feature in IBM WebSphere Portal Express Version 8.5 is implemented as a IBM Digital Data Connector (DDC) for WebSphere Portal Express plug-in.

About this task

This approach makes it possible to use the extensibility features of this framework to extend the social rendering feature with extra capabilities:

- Realize custom queries against the IBM Connections server.
- Retrieve data from Connections services other than the search service.
- Make it possible for portal site visitors to contribute social information through user interfaces that are generated by IBM Web Content Manager features.

  "The social rendering Digital Data Connector plug-in"
  To implement the IBM Connections data integration, the social rendering feature in IBM WebSphere Portal Express Version 8.5 uses a custom IBM Digital Data Connector (DDC) for WebSphere Portal Express plug-in.

  "Implementing custom Connections queries" on page 2238
  IBM Connections provides a comprehensive remote API. You can use this API to access social data through Atom feeds.

  "Dynamic Connections source URL construction" on page 2239
  To dynamically construct IBM Connections query URLs, you can use the `[Plugin:ConnectionsContext]` and `[Plugin:ResourceURL]`.

  "Organizing your custom Connections queries" on page 2239
  For the different query URLs that you want to use, you can create multiple dynamic or static IBM Connections query URL components. For convenient use of these query URL components, you can create a generic authoring template for all list definitions that you want to build from those queries.

Related concepts:

"IBM Digital Data Connector (DDC) for WebSphere Portal Express" on page 3255
You can use the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework to integrate data from external data sources on your portal pages by using IBM Web Content Manager presentation components. External data means that the data does not need to be stored directly in IBM Web Content Manager. For example, you can use DDC to render social data that you have on your IBM Connections server or on other social platforms in the context of your portal pages. Other possible data sources include news feeds, task lists, product catalog information, to name just a few.

Related information:

The IBM Connections context rendering plug-in

Using the list-rendering cache

The social rendering Digital Data Connector plug-in

To implement the IBM Connections data integration, the social rendering feature in IBM WebSphere Portal Express Version 8.5 uses a custom IBM Digital Data Connector (DDC) for WebSphere Portal Express plug-in.

You identify this social rendering DDC plug-in by using the DDC plug-in extension ID `ibm.portal.ddc.sr`.

The social rendering DDC plug-in delegates the data loading and XML transformation to the generic XML DDC plug-in. As a result, you can control this data transformation by XPath based list-rendering profiles. WebSphere Portal Express Version 8.5 includes a set of such profiles for a subset of the available Connections services. For more detailed information, read *Digital Data Connector profiles for social rendering*. You can use these profiles as is, or extend them by

creating custom extensions of the profiles, or create your own list-rendering profiles. The social rendering DDC plug-in adds various computed attributes to the profiles.

The social rendering DDC plug-in supports the `source` list-rendering context attribute. You can use this attribute to directly specify the source URL that serves the Connections XML data. To dynamically construct those source URLs, you can use the `[Plugin:ConnectionsContext]` and `[Plugin:URLParam]` tags. For more detailed information, read *Dynamic IBM Connections source URL construction*.

Related reference:

"Dynamic Connections source URL construction" on page 2239
To dynamically construct IBM Connections query URLs, you can use the `[Plugin:ConnectionsContext]` and `[Plugin:ResourceURL]`.

"Digital Data Connector profiles for social rendering" on page 2147
Starting with Version 8.5, IBM WebSphere Portal Express includes a set of IBM Digital Data Connector (DDC) for WebSphere Portal Express profiles. You can use them with the social rendering DDC plug-in that is identified by the extension ID `ibm.portal.ddc.sr`.

Implementing custom Connections queries

IBM Connections provides a comprehensive remote API. You can use this API to access social data through Atom feeds.

About this task

For more information about the API, read *IBM Connections 4.5 API Documentation*.

You can build custom queries for social data by constructing corresponding query URLs. To render the result of such a query on your portal pages by using IBM Web Content Manager design components, proceed as follows:

Procedure

1. Identify or create the list-rendering profile for the data format that your query URL serves. For example, your query might use the Connections Forums remote API to query for a specific set of forum topics. In this case, you can use the forum topics `ibm.portal.sr.forums.topics` profile to transform the result documents that this URL serves.
2. Establish the list-rendering context for this query. To do so, specify your query URL as a source attribute, and address the appropriate IBM Digital Data Connector (DDC) for WebSphere Portal Express plug-in that can handle your request. You have two choices:
   - If the profile that you use is based on the profiles that are included with IBM WebSphere Portal Express Version 8.5, you can use the social rendering DDC plug-in.
   - If you use a custom list-rendering profile, you can use either the generic XML DDC plug-in or a custom DDC plug-in that you deployed.

   Example: To retrieve the list of all public communities, you establish the list-rendering context as follows:

   ```
   [Plugin:ListRenderingContext action="set"
      extension-id="ibm.portal.ddc.sr"
      profile="ibm.portal.sr.communities"
      attribute="source=https://www.ibm.com/connections/communities/service/atom/catalog/public"]
   ```
3. To render the result, you must render an appearance component of Digital Data Connector. In that component, you extract the individual pieces of information.

You do so by using the [AttributeResource] tag and the attribute names that the list-rendering profile that you use supports. Example: In the community query that is shown in the previous step, you can render a link to a detail view of the community. You do so by adding the following markup to your Result Design section of the DDC appearance component:

```
<a href="[AttributeResource attributeName='link']">
    [AttributeResource attributeName='title']
</a>
```

4. After the rendering of the appearance component is completed, make sure to remove the list rendering context again. This is especially important if you want to nest list rendering contexts to render nested lists. You remove the list rendering context as follows:

   ```
   [Plugin:ListRenderingContext action="remove"]
   ```

   For more information, read *Setting the list-rendering context*.

Related information:

The IBM Connections context rendering plug-in

 IBM Connections 4.5 API Documentation

Dynamic Connections source URL construction

To dynamically construct IBM Connections query URLs, you can use the [Plugin:ConnectionsContext] and [Plugin:ResourceURL].

The [Plugin:ConnectionsContext] plug-in serves relevant Connections context and configuration information. This includes the service base URLs, the UUID of the community that is associated to the current page, and the source URL of social Atom object entries that a user clicked. For more detailed information about this plug-in, read *The connection context rendering plug-in*.

You can use the [Plugin:ResourceURL] to construct complex query URLs. You do so by adding one or more query parameters to an existing Connections service base URL. For more detailed information about this plug-in, read *The resource URL plug-in*.

Example: To dynamically construct the query URL that serves the information about the members of the community that associated to the current page, you create an HTML component that contains the following markup:

```
[Plugin:ResourceURL param="role=all" param="format=full"
   param="sortBy=created" param="sortOrder=desc"
   url="[Plugin:ConnectionsContext type='service'
   key='communities']/service/atom/community/members"
   compute="always"]
```

Related information:

The IBM Connections context rendering plug-in

Organizing your custom Connections queries

For the different query URLs that you want to use, you can create multiple dynamic or static IBM Connections query URL components. For convenient use of these query URL components, you can create a generic authoring template for all list definitions that you want to build from those queries.

About this task

When you work with IBM Digital Data Connector (DDC) for WebSphere Portal Express, the generic authoring template must comprise short text elements for identifying the following elements:

- The DDC plug-in ID and the profile name. For example, they can be elements named `provider` and `profile`.
- Component references to the query URL HTML component and the appearance component. For example, they can be elements named `sourceref` and `design`.

You can now create individual list definition content items from this authoring template for your Connections queries. To set up the query-specific list-rendering context and to render the query-specific appearance component, you can use a generic presentation template for all those queries as follows:

```
[Plugin:ListRenderingContext compute="always"
   extension-id="[Element context='current' type='content' key='provider']"
   profile="[Element context='current' type='content' key='profile']"
   attribute="source=[Element context='current' type='content' key='sourceref']"]
   [Element context="current" type="content" key="design" compute="always"]
```

For example, a list definition content item for the query that was defined in *Dynamic Connections source URL Construction* specifies the following settings:

- It sets the `provider` field to `ibm.portal.ddc.sr`.
- It sets the `profile` field to `ibm.portal.sr.communities.members`.
- The `sourceref` component reference points to the query HTML component. For a description of that component, read *Dynamic Connections source URL construction*.
- The design component reference points to an appearance component that generates the list of community member markup. The markup is generated based on the attributes that are defined in the `ibm.portal.sr.communities.members` list-rendering profile.

Related reference:

"Dynamic Connections source URL construction" on page 2239
To dynamically construct IBM Connections query URLs, you can use the `[Plugin:ConnectionsContext]` and `[Plugin:ResourceURL]`.

Setting up marketing campaigns

Use these tools and features to setup marketing campaigns for your website.

"Personalization"
Portal Personalization provides automatic customization of website content for individual users and user groups.

"Social Media Publisher" on page 2421
The Social Media Publisher for Web Content Manager is an extension to Web Content Manager that allows businesses to promote their web content on social networks, and provide some basic statistics about the promoted content.

Personalization

Portal Personalization provides automatic customization of website content for individual users and user groups.

Personalization can recognize a specific user based on a profile or can determine characteristics of a user based on previous purchases, products or pages viewed, and so forth. Personalization then selects content that is appropriate for that profile. If a person has a high salary range, Personalization can be configured to

retrieve information about a commercial website premium product. If an individual belongs to a particular geographic region, content specific to that region may be targeted to the individual. The page is assembled with the proper personalized information, and the user sees her personalized page.

Personalization is composed of

- Personalization browser - The Personalization user interface:
  - registers resource collections
  - authors rules, campaigns, and content spots
  - maps rules into content spots for a particular campaign

  Since objects are authored through the Personalization server, the Personalization browser can display rules in production as well as view rules in a staging environment.
- Rules engine - The rules engine executes rules created in the Personalization browser. A programming interface exists for Personalization to invoke rules, Personalization rules may be invoked through the Personalized List portlet, or rules may be invoked through Web Content Manager Personalization components. Rules associated with pages or portlets through Portal Administration are also automatically triggered.
- LikeMinds Recommendation engine - The recommendation engine evaluates recommendation rules created in the Personalization browser.
- Resource engine - The resource engine resolves the queries produced by rules into content pieces to be returned. Content for Personalization is created and approved using whatever content management tool you choose, or may come from an LDAP or any other database. Content is accessed via a set of Resource Collection classes.
- A logging framework - The logging framework is used to record information about website usage to the feedback database and the recommendation engine. It is entirely up to the site developers to decide what information is logged.

The engines are sometimes collectively referred to as the Personalization run time server.

The engine identifies the particular user. Personalization retrieves that person's profile. For example, a person may have a salary range included in her profile. Personalization then selects content that is appropriate for that profile. If a person has a high salary range, region code, or other information, Personalization can be configured to retrieve information about a commercial website premium product. The page is assembled with the proper personalized information. The user sees her personalized page.

Types of Personalization

There are three types of Personalization:

Simple filtering
: A site displays content based on predefined groups of site visitors. For example, if a site visitor is in the Human Resources department, the site provides access to URLs containing Human Resources policy manuals.

Rules engines
: In a rules based system, the site owner defines a set of business rules which determine what category of content is shown when a certain profile type visits the site. An example would be: Display all four wheel drive SUVs to visitors in the northeast in the 21 to 35 age group.

This approach has the advantage of driving the site's behavior with the business objectives of the site owner. The site owner is usually the owner of a marketing campaign or some other business manager.

Collaborative filtering
A site visitor rates a selection of products, explicitly or implicitly. Those ratings are compared with the ratings offered by other visitors. Software algorithms detect similarities. For example, a visitor receives book recommendations based on the similar purchases of others.

Rules versus collaborative filtering

When complex filtering is required, a rule-based system may work better than collaborative filtering, and vice versa. The following table details examples where one type of personalization is better than the other.

Table 342. When to use rules filtering versus collaborative filtering.

Scenario	Which filtering type to use	Reason
If the number of items offered and users who purchase them are rather low.	Rules	Very little room to compute user similarity necessary for collaborative filtering.
If price points are high or purchasing frequency is low.	Rules	Finite, limited arenas - collaborative filtering fails because of the inherent lack of diversity.
If there is a pre-existing dependency between items. Example: Disability policy required for homeowner	Rules	Recommending a disability policy just because collaborative filtering says many others "like this user" also bought a policy is incorrect--one must have the homeowner policy first.
If number of items offered and users who purchase them are rather high.	Collaborative	Cannot write rules covering all items.
If price points are low, all quite dissimilar, or the products offered have a wide range of user appeal.	Collaborative	The wide variance fits the collaborative filtering approach. Collaborative filtering also lowers the risk of making "bad" recommendations.
When not much information is gathered about the user, but the user can be identified, possibly by a login or cookie.	Collaborative	In this case, user attributes on which to base rules may be lacking. Collaborative filtering can compare the user's experiences on the site to other users.

1. “How a site is personalized” on page 2244
   Use this topic to understand how to define a personalized list of new articles for a website, such as a section of an intranet site for targeted employee bulletins or where the content of the site is tailored to the particular user.
2. “Personalization terms” on page 2245
   The concepts and principles for working with Portal Personalization require an understanding of terminology.

3. “The Personalization interface” on page 2296
   The Portal Personalization user interface consists of three portlets: the Personalization Navigator, Personalization Editor, and the Personalized List.
4. “Publishing personalization rules overview” on page 2298
   WebSphere Portal Express Personalization sends published rules across HTTP to a servlet which resides on each personalization server. This servlet can receive publishing data or initiate new publishing jobs. When a user begins a publishing job from the personalization authoring environment, the local servlet is provided with the set of information necessary to complete the job. The local servlet contacts the destination endpoint servlet (which could be the same servlet) and sends its data to it. The destination servlet reports success or failure.
5. “The Web Content resource collection” on page 2308
   The Web Content resource collection is installed and configured out of the box. This predefined collection allows you to write rules that select lists of content from IBM Web Content Manager. Rules specify which Web content to show in a Portal Personalization component in Web Content Manager.
6. “The Portal User resource collection” on page 2311
   Portal Personalization comes with a Portal User resource collection. This collection uses public APIs provided by IBM WebSphere Portal Express to access user information.
7. “LikeMinds Recommendations” on page 2313
   Personalization contains a dynamic recommendation system based on LikeMinds. LikeMinds is software that is used with your e-commerce applications. LikeMinds analyzes user interactions that occur on your Web site and generates real time predictions and recommendations to your Web site users.
8. “Feedback and analytics” on page 2344
   Personalization provides a complete logging framework for collecting data on how visitors are using your Web site. If Feedback is enabled, data is automatically collected about each Personalization rule that is fired. In addition, development tools enable Web sites to collect a variety of data related to visitors' actions and behavior. By default this data is logged to a standard database schema for later analysis and reporting. The framework is also extensible, allowing Web sites to customize and supplement the way data is collected and stored to more fully meet their needs.
9. “Developing a personalized portlet” on page 2385
   This exercise demonstrates how to use Personalization features of WebSphere Portal and Rational Application Developer to build your first personalized portlet. Your final result is a working portlet that uses Personalization rules and content spots to display personal news based on user attributes (or profiles).
10. “Personalization programming reference” on page 2402
    IBM WebSphere Portal Express provides the programming model, processes, and APIs for the Personalization rules and resource engines.

Related concepts:

“Developing a personalized portlet” on page 2385
This exercise demonstrates how to use Personalization features of WebSphere Portal and Rational Application Developer to build your first personalized portlet. Your final result is a working portlet that uses Personalization rules and content spots to display personal news based on user attributes (or profiles).

How a site is personalized

Use this topic to understand how to define a personalized list of new articles for a website, such as a section of an intranet site for targeted employee bulletins or where the content of the site is tailored to the particular user.

Developing a personalized portlet or website

1. A web developer defines an area of a site that needs Personalization. This area may be a personalized list of new articles appearing in a website, a place on an intranet site for targeted employee bulletins such as changes to benefit plans, a product list on a commerce website, or any other place where the content of the site should be tailored to the particular user.
2. A set of Resource Collections and Application Objects are defined. Together, these objects make up the business vocabulary that represents the terms and objects upon which Personalization decisions are based. These objects may be defined in the web page by pointing to an IBM Java Content Repository item type. These objects may be generated through a set of Personalization wizards provided with IBM Rational Application Developer; or they may be developed according to a set of public programming interfaces.
3. The Resource Collections and Application Objects are registered to the Personalization server through the Personalization browser by importing .hrf files. These files define Resource Collections. The developer can also manually create the Resource Collection and Application Object references in the Personalization Browser.

   Note: The classes used for the Resource Collections and Application Objects must be on the classpath for both the application being personalized and for the Personalization browser web application. One way to achieve this task is to use a shared library placed either on the application server, or on the individual web applications. If you are using stock resource collections provided with IBM WebSphere Portal Express, such as the Portal User Resource Collection or the Java Content Repository Resource Collection, these classpaths are already registered properly.
4. Programmers now use the objects and terms defined through the Resource Collections and Application Objects to write rules and map those rules to content spots using campaigns.
5. Portlet and website developers may either configure a Personalization Rule Display portlet to show the rule or content spot or may call into the Personalization programming interfaces to execute the appropriate rules or content spots which the programmers have defined.

Note: The Content Spot provides a way for the site developers to develop personalized pages before the rules are authored as well as a way to more loosely tie a particular rule to a page. It is then up to a programmer to "map" the Content Spot to a Rule using a Rule Mapping within a Campaign in a Personalization browser.

Using Personalization Rules

1. A user navigates to the page containing Personalization rules or content spots.
2. The application invokes Personalization to find content or make decisions.
3. Personalization identifies the correct rules to execute. If a Content Spot with the name given to execute is not found, a rule is looked for directly with this name.
4. The Personalization server searches for all Rule Mappings for the Content Spot. The server looks for Rule Mappings which have started, but not yet ended.

5. The Rule Mappings are ordered based on the priority and split values. The rules associated with each mapping are executed in turn until a rule returns content.

   Note: It is possible to configure Personalization so that only the first rule (higher priority) will get executed.
6. For each rule executed, Personalization retrieves the user's profile and evaluates the rule to select the content which meets the conditions of the rule. The rules engine will invoke the resource engine as necessary to retrieve content pieces.
7. The content is returned to the web page, and displays for the user.

Related concepts:

"Developing a personalized portlet" on page 2385
This exercise demonstrates how to use Personalization features of WebSphere Portal and Rational Application Developer to build your first personalized portlet. Your final result is a working portlet that uses Personalization rules and content spots to display personal news based on user attributes (or profiles).

Personalization terms

The concepts and principles for working with Portal Personalization require an understanding of terminology.

1. "Resources, resource instances, and resource collections" on page 2246
   Before you can personalize IBM WebSphere Portal Express resources, you need to understanding the terms for portal objects stored in the content repository.
2. "User resources" on page 2247
   Your Web site visitors want to quickly access the Web content that meets their needs and interests. The needs and interests of the site visitors are stored as properties in the user profile data store.
3. "Content resources" on page 2248
   Web content resources consist of data that is formatted and displayed on Web pages. Examples include news articles, products, statistics and educational materials.
4. "Attribute Based Administration" on page 2248
   Attribute based administration provides a facility to customize the layout of a page for individual authenticated users by using rules to show or hide pages or portlets. This implementation tells the portal to show or hide pages and portlets based on dynamic characteristics that are determined at runtime. Attribute Based Administration is only available for authenticated users. For anonymous users, all pages are shown.
5. "Rules" on page 2250
   Rules are used to define how your Web site interacts with individual and groups of Web site visitors. Rules are composed of easy-to-read logic statements that, in their final form, specify how to evaluate various conditions and what actions to take based on those conditions.
6. "Content spots" on page 2271
   A content spot is a placeholder or slot for a rule on a Web page. When the page is viewed, the content spot uses its rule mapping to determine which rule to execute. When the rule is executed, any actions defined within the rule take place. Each content spot has a unique name. A content spot's content type must be defined when it is created and should not be changed.
7. "Rule spot mappings" on page 2272
   For a rule to be used on your Web site, it must be mapped to an existing content spot. A rule spot mapping is merely an association between a content

spot and a rule. Changing the rule that is executed in the run-time environment is as easy as mapping a different rule to a content spot.

8. "Campaigns" on page 2272
   Campaigns are a means of organizing and implementing sets of personalization behavior. A useful analogy is an advertising campaign, which targets specific audiences with high-priority information for a specified period of time. Campaigns achieve this by allowing you to preferentially display campaign-related content in the content spots of a Web site. To accomplish such a goal, a campaign contains a set of rule-to-content spot mappings, start dates, and stop dates.
9. "Application object" on page 2273
   An application object is a java object existing at a known location in the request context. Defining an application object involves specifying the object's class name (as a Java class), and specifying a key (string key into a session attribute) to find it in the request context. Personalization calls methods on these objects during rule execution and uses their results in its decision making. Application Objects that implement the SelfInitializingApplicationObject interface are automatically instantiated as needed by Personalization.
10. "Request Context" on page 2295
    This is the interface used to access various attributes for rules. For HTTP contexts, it provides access to the HttpRequest and HttpSession attributes. For non-HTTP contexts, it provides the same interface to a surrogate for the request and session.
11. "Query framework" on page 2296
    The query framework is an object representation of a query. The framework is not specifically oriented toward querying either object or relational databases. A set of common operators is defined, but what an attribute represents is determined by the interpreter of the query.

Resources, resource instances, and resource collections:

Before you can personalize IBM WebSphere Portal Express resources, you need to understanding the terms for portal objects stored in the content repository.

A *resource* is a Java class that defines the properties of a user or content object. In database terms, it is analogous to the database schema that defines the column names and types for a database table. Resource classes must implement the com.ibm.websphere.personalization.resources.Resource interface

A *resource instance* is an instance of the resource class. Again, using a database analogy, the resource instance is similar to a row of a database table because it contains actual values for each property defined by the resource.

A *resource collection* is a Java class that represents and allows access to a collection of resource instances. It is similar to a database table with a fixed schema and a number of rows. Resource collection classes must implement the com.ibm.websphere.personalization.resources.ResourceDomain3 interface. Rational Application Developer provides a wizard that can create resource collections that store data in SQL databases or LDAP repositories. The classes that can make up the resource collection are as follows:

Resource Class
An instance of `com.ibm.websphere.personalization.resources`.

Resource Manager Class
An instance of
`com.ibm.websphere.personalization.resources.ResourceManager3`.

Domain Class
An instance of
`com.ibm.websphere.personalization.resources.ResourceDomain3`.

Translator Class
An instance of
`com.ibm.websphere.personalization.resources.AuthIDTranslator`.

For more details, refer to the Javadoc API documentation for Personalization APIs You can provide your own implementation of these classes or use the RAD Personalization Wizard to generate classes that query against SQL or LDAP repositories.

While resources, resource instances, and resource collections are easily mapped to familiar database concepts, it is important to note that the actual content store they refer to does not have to be a database table. It can be a file system, an LDAP repository, an XML store, or virtually any content store accessible by Java.

Related information:

"Using the Personalization APIs" on page 2409
Personalization provides open APIs that enable the Personalization run-time environment and Rational Application Developer to access user and content data in customer data stores.

User resources:

Your Web site visitors want to quickly access the Web content that meets their needs and interests. The needs and interests of the site visitors are stored as properties in the user profile data store.

Many sites obtain the user needs and preferences by using an HTML input form. The input form includes information that seldom changes (such as the user name and address) and information that must be updated over time (such as gift wish lists). In addition to explicit information provided by the site visitor in the HTML input form, a business might also programmatically update the user profile data store with information obtained from other sources. These sources could include an analysis of a user's actions on the Web site and data from a user's non-Web interactions with the business.

Although users are unique, personalizing Web content does not often require creating totally unique content for each user. Users can be profiled or grouped into categories that facilitate personalization. For example, on an internal Web site, certain information might be appropriate for managers, while other content may be suitable for salespersons.

Note: The Portal User Collection resource collection included with Personalization cannot currently be used to select lists of users in select content rules. The resource collection may be used to profile the user in a profiler rule. This applies to both the user resource collection which is installed with Personalization as well as any custom resource collections created with the manager class com.ibm.websphere.personalization.resources.wps.UserManager.

Content resources:

Web content resources consist of data that is formatted and displayed on Web pages. Examples include news articles, products, statistics and educational materials.

The content is displayed when a user requests a JSP or servlet that dynamically generates and formats a Web page in the appropriate format, such as HTML or XML. When personalizing your Web site, you only have to customize the content you want to selectively display (or suppress) based on the user. For example, a given page might consist of 40% static content and 60% variable content based on particular user characteristics or business conditions, while other pages might be 100% static content.

Content resources can be file content or structured content.

Attribute Based Administration:

Attribute based administration provides a facility to customize the layout of a page for individual authenticated users by using rules to show or hide pages or portlets. This implementation tells the portal to show or hide pages and portlets based on dynamic characteristics that are determined at runtime. Attribute Based Administration is only available for authenticated users. For anonymous users, all pages are shown.

Portal Personalization rules can be used to control whether a page is displayed in the site navigation; this is managed by choosing a rule appropriate for the user attribute you want to enable to see the page. If the rule returns true, the page or portlet will be shown, otherwise, it will be hidden. If Personalization is not installed or is not enabled in the properties settings, you will not see this option.

Access Control and Visibility Rules

Access Control and Visibility Rules both impact what appears on a portal page or portlet. Access Control determines what a user is allowed to see on a page or in a portlet. In order to see pages and portlets, a user must be explicitly defined as a member of the group that has access. The groups are arranged in a hierarchy and each group is assigned roles such as administrators or editors.

Visibility rules determine what a user will see, or what has been targeted towards a user, and Access Control is based on group membership, visibility rules use any type of information, including LDAP attributes, or time of day. For example if you want to hide a portlet for an individual in a certain geography, store the location as an attribute in LDAP, and assign a visibility rule hiding the portlet. For example, a user may have access to the revenue figures for all divisions for the entire year, but these figures should not be displayed prominently except when they are first released. For a week after the figures are first released, the figures for the employee's division should show prominently on their home page. The visibility rule hides figures for divisions the employee is not in and only shows the employee's figures the week after they are released.

Access Control takes precedent over visibility rules. You must have access to a page or portlet prior to applying visibility rules. Access Control also determines if a page or portlet will be returned in a search; if a user does not have access, he will not be able to see the portlet or page in search results. If a user has access to a

portlet or page, but has the visibility rule set to hide the page or portlet, it will show up in search results.

"Assigning attribute based administration rules to pages and portlets"
Attribute based administration rules can be assigned manually in the portal in the Edit Properties and Edit Layout portlets, or through the XML configuration interface. The rule must be present on the system in order to assign it to a page or portlet. You can usually use Personalization Publish to accomplish this.

"Changing the error condition behavior" on page 2250
If an error occurs when locating or using a rule assigned to a page or portlet, by default that page or portlet will be hidden. If a user or expected application does not exist, the system will continue. This behavior may be appropriate, but in a development environment, you may need to change this behavior for testing purposes. Update the PersonalizationService.properties file to override this behavior globally.

Assigning attribute based administration rules to pages and portlets:

Attribute based administration rules can be assigned manually in the portal in the Edit Properties and Edit Layout portlets, or through the XML configuration interface. The rule must be present on the system in order to assign it to a page or portlet. You can usually use Personalization Publish to accomplish this.

Before you begin

- In order to map rules, non-administrator users must be given at least user access to the rule that is being mapped and edit access to the page or portlet where the rule is being mapped.
- Pages or portlets on derived pages show an inherited visibility rule if no rule is defined for them. You cannot clear the inherited visibility rules on derived pages.

About this task

Using the XML configuration interface, you can assign a rule to a page or portlet. Set the parameter **com.ibm.portal.navigation.rule** to indicate the rule to assign to the page or portlet. The value of the parameter should be the unique id or UUID of the rule. The UUID can be found by exporting the rule in the Personalization user interface and inspecting the exported XML for the **jcr:uuid** parameter.

Procedure

Example

For example, to assign a rule with the UUID `7ce9d5004ee31f41b0d3b944c9f7965c` to a page or portlet, add the following parameter to the content-node in the XML access script:

```
<parameter name="com.ibm.portal.navigation.rule" type="string"
update="set"><![CDATA[7ce9d5004ee31f41b0d3b944c9f7965c]]</parameter>
```

Related concepts:

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Changing the error condition behavior:

If an error occurs when locating or using a rule assigned to a page or portlet, by default that page or portlet will be hidden. If a user or expected application does not exist, the system will continue. This behavior may be appropriate, but in a development environment, you may need to change this behavior for testing purposes. Update the PersonalizationService.properties file to override this behavior globally.

Before you begin

About this task

The property rulesEngine.visibilityDefault specifies whether a page or portlet renders if there is a problem with the assigned rule. Changing the property value to show means the page or portlet will display even if the assigned rule cannot be found or if there is a problem with the rule.

The property rulesEngine.throwObjectNotFoundException specifies what happens if an object such as a user is not found when needed during rule execution. This may occur when Personalization cannot find the current user or when an expected application object does not exist on the session or request at the expected key. When set to false, a null user or object is not treated as an error but is instead only printed to the logs as a warning. Personalization will continue as if the requested attribute of the null object is itself null.

For example, if no user object is found and rulesEngine.throwObjectNotFoundException is set to false, a rule such as `Show page or portlet when user.name is null` would return `show`. A null user is treated as if the user name is null. However, if no user object was found and rulesEngine.throwObjectNotFoundException is set to true, this same rule would throw an exception. If this rule was used to determine the visibility of a page or portlet, the ultimate result would depend upon the value of the rulesEngine.visibilityDefault property, which would decide what occurs if an exception is thrown during processing of a rule in attribute based administration.

Procedure

1. Open the PersonalizationService.properties file. This file is located in the following directory:*wp_profile_root*`/PortalServer/config/config/services/PersonalizationService.properties`.
2. Find the rulesEngine.visibilityDefault property.
3. Set the value of this property to `show` to enable portlets or pages to render if an error occurs.
4. Fine the rulesEngine.throwObjectNotFoundException property.
5. Set the value of this property to `true` to throw an exception if the object is not found.

Rules:

Rules are used to define how your Web site interacts with individual and groups of Web site visitors. Rules are composed of easy-to-read logic statements that, in their final form, specify how to evaluate various conditions and what actions to take based on those conditions.

"Actions" on page 2251
Actions use simple evaluation statements to select content to use or display, or

to set information. Basic arithmetic calculations (addition, subtraction, multiplication, division) that follow order of operations can be used on either side of the evaluation; parentheses are not supported. Actions can be combined with profilers into bindings.

"Profilers" on page 2253
Profilers are typically used to categorize an individual (usually the current site visitor) according to his or her user properties.

"Bindings" on page 2256
Bindings combine actions and profilers to specify actions to perform when defined conditions are encountered. Returned content can be sorted or filtered prior to display or use. Actions can be set to be performed by *always*, *exclude*, and *otherwise* blocks. In addition, the total number of items used can be limited.

"Recommend Content" on page 2259
You use Recommend Content rules (also referred to as recommendation rules) to recommend content to your Web site visitors. Recommendation rules, powered by LikeMinds, recommend content based on users' past interactions with your Web site.

"Visibility Rules" on page 2260
Visibility rules determine what a user will see, or what has been targeted towards a user. Visibility rules can be assigned to pages and portlets and will be triggered automatically by the portal as needed.

"Rule elements" on page 2261
Learn about the options in the rule editor for the different types of rules.

"Email" on page 2269
View the prerequisites for creating an email action or promotion within the Personalization workspace.

"Email administration" on page 2270
View the steps you need to complete before your run time server can send personalized email.

Actions:

Actions use simple evaluation statements to select content to use or display, or to set information. Basic arithmetic calculations (addition, subtraction, multiplication, division) that follow order of operations can be used on either side of the evaluation; parentheses are not supported. Actions can be combined with profilers into bindings.

The following action items are supported:

Select actions
Select data or content. Select actions retrieve data from a data store, typically for display on a web page. Select actions also can be used within bindings to exclude certain content, that is, filtering a subset of returned content from a superset.

Update actions
Update content or objects on the request. Update actions cannot select content. Update actions are used to store content or data in the user profile, an application object, or other data stores.

Email actions
Send email messages using a web page as the body. Email actions cannot select content. An email action sends an email message to a recipient or list of recipients. An email action assigned to a content spot sends an email

message when the content spot is triggered. For example, at the time a website visitor views a page with the content spot.

The email action editor allows the fields that identify the recipients (primary, copied and blind copied), the subject line, and the sender to be identified by either explicit text or by resources attributes. The email body is a separate file, such as a text file, an HTML file, or a JSP, that must be accessible from the email server via a URI. An emailed JSP can contain content spots for personalizing the email message for the user who triggers the content spot.

Although email actions differ somewhat from email promotions, both have prerequisites that must be in place and working before email can be sent.

"Example: Simple select content action"
View an example for a simple select content action.

"Example: Simple update action"
View an example of an update action that is part of a Web site that allows visitors to manage certain information and preferences about themselves.

"Example: Simple email action" on page 2253
This email action rule example is typical of one that might be used after a website visitor submits a form indicating interest in an item or service.

Example: Simple select content action:

View an example for a simple select content action.

The select content action shown here, `Get Bank News By Role`, queries all records within a content resource entitled News and returns those marked as being for the current user's role.

The content resource YourCoNews represents news articles in the data store. Each record has several different fields (for example, Title, Abstract, Author, Body), including a field entitled `Role`. In the data store, this field is marked to indicate the role of the visitor to whom it applies.

Simple select content action

```
Select NewsArticle
       whose Role is current User.Role
        order as is
```

Example: Simple update action:

View an example of an update action that is part of a Web site that allows visitors to manage certain information and preferences about themselves.

When executed, the following update action will write to the fields (`Income Group`, `Role`) in the data store for the record associated with the current user (the current Web site visitor), using data contained in the current user's session variables, such as `incomeGroup` and `role`.

Update action

```
Update
        current Portal Users.Income Group set to current Session.incomeGroup
        current Portal Users.Role set to current Session.role
        current Portal Users.Last Name set to current Session.lastName
        current Portal Users.Title set to current Session.title
```

Example: Simple email action:

This email action rule example is typical of one that might be used after a website visitor submits a form indicating interest in an item or service.

The email action rule can be attached to a content spot that, when triggered by the visitor viewing the page with the spot, sends the email indicated by the **bodyURI** field to that visitor. This email is also blind copied to someone within the sample company.

Email action rule

```
To: current Portal Users.Email Address
 From: Rates@YourCo.com
 bcc: Mortgage_Broker@YourCo.com

 Subject: Today's Mortgage Rates
 Body URI /email/mortgage-rates.jsp
```

Related information:

"Email administration" on page 2270
View the steps you need to complete before your run time server can send personalized email.

Profilers:

Profilers are typically used to categorize an individual (usually the current site visitor) according to his or her user properties.

Besides a user's properties, profilers can be used to define other conditions based on such factors such as the current date and time, the type of browser the visitor uses, or other implicit and explicit application object properties. Profilers can also make decisions based on the current user's session attributes and request attributes and parameters, along with category and action counts.

Profilers can be constructed to define the conditions of arbitrarily named profiles, or can be defined in terms of other profilers. For example, you can create a profiler that will evaluate as true if a profile is in any, all, or none of a group of other profiles.

"Example: Simple profiler" on page 2254
View an example of a simple profiler that determines whether confidential company news articles will be shown to the current Web site visitor.

"Example: Nested profiler" on page 2254
View an example of a nested profiler.

"Example: Category Count (implicit profiling)" on page 2254
Get an overview of implementing category counts from a profiler that will contain profile definitions for movies, cooking, and sports; this overview sets the profile for the category with the highest count.

"Example: Browser capability" on page 2255
The browser capability stock object allows you to profile the browser the current Web visitor is using to view your site. View an example profiler that checks the browser version to determine whether it is supported.

"Example: "Count of" (quantifiable condition)" on page 2255
View an example demonstrating the use of "Count of".

"Example: Request attributes and session attributes" on page 2256
Request values passed to the JSP or stored in the HTTP session can be referred to from rules. To refer to them in a rule, you must know the variables and their possible values.
"Example: Arithmetic operation" on page 2256
View an example of a profiler that uses arithmetic operations.

Example: Simple profiler:

View an example of a simple profiler that determines whether confidential company news articles will be shown to the current Web site visitor.

The profiler `User Clearance` is based on a user resource called `Personnel`. When the profiler was created, the name `User Clearance` and the settings, `Confidential` and `Regular`, were arbitrarily defined for later reference within bindings. One side of the comparison line (`current Personnel.Role`) refers to a user resource named `Personnel` created from a Personnel table in a database. `Role` is a mapped value, defined when the resource was created, that points to the Personnel.Role column in the Personnel table. The values in the Role column in the database are either Employee, Executive, or Manager.

This completed profiler is used within a binding as a means of determining whether confidential company news articles will be shown to the current Web site visitor.

Simple profiler

```
User Clearance is
       Confidential when
           current Personnel.Role is Executive or
           current Personnel.Role is Manager
      Otherwise Regular
```

Example: Nested profiler:

View an example of a nested profiler.

A nested profiler is true if any, all, or none of the profilers within the profiler are true. You can categorize a condition as a combination of other conditions. For example, a Web site visitor can be profiled as a young male if a pre-existing gender profiler classifies the visitor as male, and a pre-existing age profiler classifies as the visitor as being in his teens or twenties.

Nested profiler

```
AgeGenderProfiler is
       YoungMale when
           GenderProfiler is Male and
           AgeProfiler is any of Teenager or Twenties
      Otherwise NotInTargetAudience
```

Example: Category Count (implicit profiling):

Get an overview of implementing category counts from a profiler that will contain profile definitions for movies, cooking, and sports; this overview sets the profile for the category with the highest count.

For the Category Count example, assume a repository of articles on sports, movies, and cooking is available for the user to view. Each time the user views an article, a

record is logged to show his or her preference for that topic. For this to occur, each article must be a JSP that implements category beans. For example, the following code would appear on a sports article:

```
<jsp:useBean class="com.ibm.wcp.analysis.beans.Category" id="category"
scope="session">
</jsp:useBean>
<% category.log(request, "Articles/Sports"); %>
```

These values were typed into the Attribute text field during the creation of this profiler after selecting current Category Count.

Note: A complete version of this profiler will contain profile definitions for movies and cooking for the cases where those category counts are greater than sports and each other.

Category count

```
ArticlePreference is
        Sports when
            current Category Count.Articles.Sports is greater than current Category Count.Articles.Cooking and
            current Category Count.Articles.Sports is greater than current Category Count.Articles.Movies
```

Example: Browser capability:

The browser capability stock object allows you to profile the browser the current Web visitor is using to view your site. View an example profiler that checks the browser version to determine whether it is supported.

When using the browser capability stock object, there is a finite list of attributes available, but you must type the values for one side of the evaluation. The browser capability script files and the SinglePixel.gif image must be properly configured in the Web application.

The following example profiler checks the browser version to determine whether it is supported. Other possible checks include available plug-ins, whether Java is enabled, and whether cookies are enabled.

```
Check Browser is
        supported when
           (
             (
              current Browser Capability.BrowserType is Netscape and
              current Browser Capability.FullVersion is greater than or equal to 6.2
             ) or
             (
              current Browser Capability.BrowserType is Internet Explorer and
              current Browser Capability.FullVersion is greater than or equal to 6.0
             )
           )
       Otherwise unsupported
```

Example: "Count of" (quantifiable condition):

View an example demonstrating the use of "Count of".

A quantifiable condition profiler is similar to implicit profiling because a profile is defined according to numbers of items. Quantifiable condition profilers do not require the use of logging beans, but require attributes of resources that will be quantified to be organized uniformly.

In the following profiler, counts are made for items in the session object `shoppingCart` used by the user resource Shopper. Here, `shoppingCart` is analogous to a table in a database and `color` would be a column of data. Each item within the table would be a row. For example:

Table 343. Counts made for items in the session object `shoppingCart` used by the user resource Shopper

Item	quantity	size	color	price
Gadget	1	L	red	$1.99
Gizmo	3	S	green	$0.95

Quantifiable condition profiler for the session object `shoppingCart`

```
ColorPreference is
Red when number of items matching (Shopper.shoppingCart.color is red) is greater than 5
Green when number of items matching (Shopper.shoppingCart.color is green) is greater than 5
```

Example: Request attributes and session attributes:

Request values passed to the JSP or stored in the HTTP session can be referred to from rules. To refer to them in a rule, you must know the variables and their possible values.

The following example profiles news articles as read or unread by the current user, based on the page request generated by a form on a JSP.

Request attributes and session attributes

```
User News is
        ReadNews when
            current Request.ProcessNews is Read
        UnreadNews when
            current Request.ProcessNews is Unread
        UnreadAllNews when
            current Request.ProcessNews is UnreadAll
```

Example: Arithmetic operation:

View an example of a profiler that uses arithmetic operations.

To build an operation, select a combination of **Resource.Attribute** and other operands. You can do this on either side of the evaluation.

Order of operations is in effect but parentheses cannot be used

```
 AgeProfiler is
       Youth when
           current Date.Year - current User.BirthDate.Year is less than or equal to 25
       Adult when
           current Date.Year - current User.BirthDate.Year is between 25 and 65
       Senior when
           current Date.Year - current User.BirthDate.Year is greater than or equal to 65
```

Bindings:

Bindings combine actions and profilers to specify actions to perform when defined conditions are encountered. Returned content can be sorted or filtered prior to display or use. Actions can be set to be performed by *always*, *exclude*, and *otherwise* blocks. In addition, the total number of items used can be limited.

The always and exclude blocks will always be performed no matter what the outcome of the profiler execution, but the otherwise block will only be run if none of the profilers match. An otherwise block works the same as an always blocks in that it adds items to the results, but for an exclude block, you actually choose an action that defines items that you don't want to include in the results. For example, on a shopping site, you may want to exclude items that have already been purchased.

"Example: Simple binding"
Because bindings couple the conditional processing of a profiler with the functional power of an action, the simplest form of a binding works like a conditional "if-then" clause.

"Example: Multiple profilers and optional actions"
View an example that demonstrates the use of a conditional "if-then" with an additional clause.

"Example: Nested bindings (simple)" on page 2258
When creating a binding, it is possible to use a binding in any of the `do action` areas. This is known as a nested binding.

"Example: Nested bindings (advanced)" on page 2259
View an example that demonstrates advanced nested bindings.

Example: Simple binding:

Because bindings couple the conditional processing of a profiler with the functional power of an action, the simplest form of a binding works like a conditional "if-then" clause.

Consider a simple binding example. If the current user is not a previous customer, then show a limited number of current offers. If the current user is a known customer, then show offers appropriate to their status level.

For example, the profiler `Customer Type` is used to check whether the current user is a known customer. If the customer is not a known customer, the action `Get Limited Number of Offers` is run. If the profiler indicates that the user is a known customer with a status of Gold or Platinum, then a different action is run and different offers are retrieved for display.

Simple binding

```
When Customer Type is
        Not A Customer
           do Get Limited Number of Offers
        Gold or
        Platinum
            do Get Offers For User
        order as is
```

Example: Multiple profilers and optional actions:

View an example that demonstrates the use of a conditional "if-then" with an additional clause.

Consider an example: If the current user does not have Confidential status, then the action `GetNonConfidentialNews` is executed. The same results could be achieved in this example by placing the `GetNonConfidentialNews` action under `Otherwise` because these are the only two profiles possible within this profiler.

The action field under Otherwise remains as is. Since the UserClearance profiler places every user into one of two categories (Regular or Confidential), any action placed here would never be executed.

The GetSiteNews rule will always be executed. Any content the rule retrieves from the data store is added to the total return set.

The GetNewsAlreadyRead action works like any other action because it retrieves content from the data store. However, when the action is placed under Exclude, any content retrieved by this action is removed from the total return set.

Note: It must be possible to indicate an article has been read by a given user. When you click the Select Action menu, you will only see rules that are assigned a Select Action type. Binding rules are also Select Action type rules. Once a resourceCollectiontype is set for the binding, all of the action rules will be locked and will use the same collection type.

The order of the total return set is randomized and the number truncated to no more than 10 items. This effect takes place each time the rule is executed, so the news articles displayed on the Web page will change from page view to page view.

Conditional "if-then" with an additional clause

```
When UserClearance is
        Confidential
           do GetConfidentialNews
        Regular
           do GetNonConfidentialNews
       Always
            GetSiteNews
       Exclude
            GetNewsAlreadyRead
        order randomly
        show 10 items
```

Example: Nested bindings (simple):

When creating a binding, it is possible to use a binding in any of the do action areas. This is known as a nested binding.

In this example, the Always action (Get Top Products) in Get Products By Location is actually another binding. When the nested binding is placed with Always, it has the effect of the boolean operator or.

For example: The total rule returns content that meets the conditions in the earlier part of the binding or that meets the conditions in the later part.

Note: It is possible for a nested binding to contain nested bindings.

Get Products By Location:

Nested binding

```
When User Location is
        Lab
           do Get Test Products
        Factory
           do Get Available Products
        Field
           do Get All Products
       Otherwise
```

```
            do Get Future Products
        Always
            Get Top Products
         order as is

Get Top Products:
   When User Role is
         Manager
            do Get Top Selling Products
         Exec
            do Get Top Selling Products
         Employee
            do Get Top Overstocked Products
         order as is
```

Example: Nested bindings (advanced):

View an example that demonstrates advanced nested bindings.

This example is similar to the Nested bindings (simple) example, except for the addition of the binding `Get Top Products` to the actions done when the `User Location` profiler is `Lab`. Multiple actions can be grouped by selecting multiple actions or bindings simultaneously within the rule editor. Selecting multiple actions or bindings here has the effect of the boolean operator "and", which returns the intersection of the data sets.

For example: the current user location must be in the lab and an executive. Therefore, in addition to test products, executives and managers in the lab receives information about the best selling products, and employees at the lab gets the most popular overstocked items. Factory or field workers will not see either best selling or most popular overstocked products.

Modified `Get Products By Location`:

Advanced nested bindings

```
When User Location is
         Lab
            do Get Top Products and
            do Get Test Products
         Factory
            do Get Available Products
         Field
            do Get All Products
        Otherwise
            do Get Future Products
         order as is
```

Recommend Content:

You use Recommend Content rules (also referred to as recommendation rules) to recommend content to your Web site visitors. Recommendation rules, powered by LikeMinds, recommend content based on users' past interactions with your Web site.

When creating a Recommend Content rule, you specify one of three recommendation methods. The recommendation methods are:

how the current user navigated the site
: This method is associated with the LikeMinds Clickstream Engine.

preferences explicitly expressed by the user

Use this recommendation method to generate recommendations based on users' ratings of items. This method is associated with the LikeMinds Preference engine.

Items map to a piece of content and are represented by resources and resource collections. Your Web site captures ratings using the Rating bean. The Rating bean collects the rating, the item resource, and resource collection, and then logs the data for LikeMinds to use later.

association with content returned from a rule

Use this recommendation method to generate recommendations based on market-basket analysis. This method associates items that a current Web site user has interest in (such as an item in their shopping cart) with items that others users have had interest in or have purchased. This method is associated with the LikeMinds Item Affinity engine.

Item affinity rules make use of the LikeMinds transaction data being collected. They offer a method for generating recommendations from a known set of resources (actually the results of another rule returning the same resource type).

Notes:

- Before using Recommend Content rules, check with your system administrator to see which LikeMinds engines are configured and running on the production run-time server.
- To preview results, the production LikeMinds database must contain data, including items, users, and transactions (ratings or actions). The problem of initial data priming is commonly called coldstart.

Visibility Rules:

Visibility rules determine what a user will see, or what has been targeted towards a user. Visibility rules can be assigned to pages and portlets and will be triggered automatically by the portal as needed.

Visibility rules use any type of information, including LDAP attributes, time of day, or session information. For example, if you want to hide a portlet for an individual in a certain geography, store the location as an attribute in LDAP, and assign a visibility rule hiding the portlet. A user may have access to the revenue figures for all divisions the entire year, but these figures should not be displayed prominently except when they are first released. For a week after the figures are first released, the figures for the employee's division should show prominently on their home page. The visibility rule hides figures for divisions the employee is not in and only shows the employee's figures the week after they are released.

Visibility rules can be assigned to pages and portlets and will be triggered automatically by the portal as needed. Through the APIs, visibility rules behave like profiler rules where the only two possible profiles are **show** and **hide**. This allows visibility rules to be invoked programmatically and used in any custom application just as you would call a profiler rule. Visibility rules only apply to authenticated users.

"Example: Show page or portlet" on page 2261
View an example of a visibility rule, `Show Page` that shows the specified page or portlet only during the specified time period, and only to users in the Midwest. For all other dates and users, the page or portlet is hidden.

Related concepts:

"Attribute Based Administration" on page 2248
Attribute based administration provides a facility to customize the layout of a page for individual authenticated users by using rules to show or hide pages or portlets. This implementation tells the portal to show or hide pages and portlets based on dynamic characteristics that are determined at runtime. Attribute Based Administration is only available for authenticated users. For anonymous users, all pages are shown.

Example: Show page or portlet:

View an example of a visibility rule, `Show Page` that shows the specified page or portlet only during the specified time period, and only to users in the Midwest. For all other dates and users, the page or portlet is hidden.

Visibility rule

```
Show page or portlet when
        current Date.Date is between December 12, 2006 and December 19, 2006
        current LdapUsers.Geography is Midwest Region
     Otherwise hide
```

Rule elements:

Learn about the options in the rule editor for the different types of rules.

"Arithmetic expressions" on page 2263
Arithmetic expressions allow you to perform mathematical operations on resource attributes as part of your rule. When you choose this option, you can select multiple resource attributes, values, and operators (addition, subtraction, multiplication, or division) to use between them.

"Count of (quantifiable conditions)" on page 2263
Create an evaluation based on a count or tally of attributes that meet your criteria.

"Current Action Count or Action Name" on page 2263
Current action count, like **current category count**, is a way to base a profiler or rule on the number of times a Web site visitor has performed certain actions. These actions must be logged using the action logging beans in order to be accessible by the rule. **Current action name** inspects the names of the actions logged.

"Current Browser Capability" on page 2263
Browser Capability is an application object that allows you to profile a Web site visitor based on the attributes or capabilities of the browser being used. When applicable, it appears in the rule editor as an option when you select **Resource.Attribute**.

"Current Date" on page 2264
The **current Date** resource contains several attributes you can use for comparison (date, day, month, time, timestamp, weekday, and year). To set values, you may enter a specific value or reference the value of a specific attribute of the same type.

"Current Request Attributes" on page 2264
Use **current Request Attributes** to inspect request attributes which can be set on the current JSP. You must know the name of the request attribute to use it in a rule. This request is the request passed into the content spot executing the rule. For example, you would use the portlet request to set the current Request attribute for portlets. The portlet request is not shared among portlets. For jsps directly within a Web application, the current Request Attribute is the HTTP request of the Web application.

Arithmetic expressions:

Arithmetic expressions allow you to perform mathematical operations on resource attributes as part of your rule. When you choose this option, you can select multiple resource attributes, values, and operators (addition, subtraction, multiplication, or division) to use between them.

An example use of an arithmetic expression is a profiler that profiles Web site visitors according to age. In the data you record for each visitor, it is more practical to store date of birth (which does not change), than to store age. In the evaluation in the profiler, you can use an arithmetic expression to calculate the visitor's age by subtracting the current user's year of birth from the current year (`current Date.year`).

Arithmetic expressions are calculated according to traditional order of operations (multiplication and division are calculated before addition and subtraction. For example, 3+2*2–1/2 evaluates to 6.5). It is not possible to group expressions using parentheses.

Count of (quantifiable conditions):

Create an evaluation based on a count or tally of attributes that meet your criteria.

In the Specify a Resource Attribute drop down, you have the option to select **Use Number of Items in List**. Selecting this option allows you to create an evaluation based on a count or tally of attributes that meet your criteria. When you select this option, you must select a resource and the attribute of that resource that are to be tallied when the rule executes.

Note: This option can only be used in profiler and visibility rules.

Because the tally is made at the time the rule is triggered, it could produce different results at different times during the session if used on an application object or a current resource. For example, you might profile a visitor's color preference as red by creating an evaluation that checks to see if the number of red items in the user's shopping cart is greater than 5. The rule syntax for this evaluation could be:

```
Count of (shoppingCart.item.color is red) is greater than 5
```

Although you can make counts of any data type, the tally must be compared against a value or resource attribute that has a data type of number, decimal number, or integer.

Current Action Count or Action Name:

Current action count, like **current category count**, is a way to base a profiler or rule on the number of times a Web site visitor has performed certain actions. These actions must be logged using the action logging beans in order to be accessible by the rule. **Current action name** inspects the names of the actions logged.

Current Browser Capability:

Browser Capability is an application object that allows you to profile a Web site visitor based on the attributes or capabilities of the browser being used. When applicable, it appears in the rule editor as an option when you select **Resource.Attribute**.

Purpose

Browser Capability currently supports these attributes:

AcceptLanguage
Returns the value of the header 'accept-language' from the request object.

AcceptMimeTypes
Returns the value of the header 'accept' from the request object.

Agent Returns the value of the header 'user-agent' from the request object. This is a lowercase string that contains information about the client software, usually the browser name or version.

BrowserType
Returns the browser type. Choices are available for supported browsers.

FullVersion
Returns the version of the browser to one point of precision. For instance, 6.1 and 6.1.1 are both returned as 6.1.

MajorVersion
Returns the first digit of the browser version. For instance, 6.0, 6.1 and 6.1.1 are all returned as 6.

Current Date:

The **current Date** resource contains several attributes you can use for comparison (date, day, month, time, timestamp, weekday, and year). To set values, you may enter a specific value or reference the value of a specific attribute of the same type.

Current Request Attributes:

Use **current Request Attributes** to inspect request attributes which can be set on the current JSP. You must know the name of the request attribute to use it in a rule. This request is the request passed into the content spot executing the rule. For example, you would use the portlet request to set the current Request attribute for portlets. The portlet request is not shared among portlets. For jsps directly within a Web application, the current Request Attribute is the HTTP request of the Web application.

Consider the following code that can be inserted into a JSP to set a request attribute:

```
<%
 request.setAttribute("user", userObject);
%>
where userObject is of any Object type
```

An example rule condition constructed to evaluate the previous example might be:

```
when current Request Attributes.user is equal to rob
```

All data types are supported.

Current Request Parameters:

Use **current Request Parameters** to inspect data contained within the query string (the variables and values that appear after a question mark on a URL).

To understand current Request parameters, consider the example URL `http://hostname/page.jsp?var1=rob&var2=expert`. In this example, the request parameters are `var1` and `var2`. Typically these are passed by GET and POST methods associated with forms. You must know the name of the request parameter that is passed by the page to use it in a rule.

Given the following <jsp:forward> command:

```
<jsp:forward page="/servlet/login">
 <jsp:param name="user" value="rob" />
</jsp:forward>
```

an example rule condition constructed to evaluate this example might be:

```
when current Request.user is rob
```

Only data types Text and List are supported.

Current Session Attributes:

Use **current Session Attributes** to inspect parameters stored within the current session object for the Web site visitor. You must know the name of the parameter to use it in a rule. All data types are supported. The current session object is the session associated with the request passed into the content spot executing the rule. For example, the current session object is the portlet session, which is unique to the portlet. For jsps within a WAR, the current session object is the the http session.

do Action:

Within a binding, you can couple actions with profilers so certain tasks are performed when certain conditions are met. You can also indicate actions to be done under other conditions. Use the **Do action**, **Otherwise do action**, **Always do action**, and **Exclude do action** elements.

Do action allows you to choose one or more actions in your project. You can also select another profiler and profile to define a combination of conditions to evaluate. These actions run when the condition in the preceding profile (or set of profiles) are met.

Note: If there are multiple actions in a binding, they must all work with resources of the same type.

Otherwise do action allows you to choose one or more actions that run when none of the preceding conditions in the profile (or set of profiles) are met. Within the otherwise clause, you can also select another profiler and profile to define a combination of conditions to evaluate.

Always do action allows you to choose one or more actions in your project that will execute whether or not any of the preceding conditions are met.

Exclude do action allows you to identify one or more actions in your project that will execute, and whose results returned will be removed from the result set generated by the other actions in the binding.

Note: Exclude takes precedence over Always.

Include Only:

Include Only is a choice within the Select Action and Recommendation Rule structures. You can select action or binding rule to be used as the include only cause. When the content is selected for the main select action or recommended rule, it is only returned if it would be selected by the Include Only clause's action rule.

is:

Select **is** to evaluate the relationship between two sides of a conditional statement.

When using **is**, either side of the conditional statement can typically be the content returned by a resource attribute, value, or arithmetic expression. If the resource attribute is of the data type List (array, vector, or enumeration), the available evaluations become `includes` and `includes any of`. Otherwise, the choices are:

- includes
- includes any of
- is between
- is between but not equal to
- is empty
- is
- is greater than
- is greater than or equal to
- is included in
- is less than or equal to
- is less than
- is not empty
- is not

Is Empty/Is Not Empty

The evaluations `is empty` and `is not empty` allow a rule to check for the existence of a null value or an empty list. When using either of these evaluations, one side of the evaluation is unnecessary and is removed.

Table 344. Examples of `Is Empty` or `Is not Empty` evaluations

One side of Evaluation	Evaluation	Result
Resource Attribute (non-list type)	is empty	true if attribute is null, otherwise false
Resource Attribute (non-list type)	is not empty	false if attribute is null, otherwise true
Resource Attribute (list type)	is empty	true if list is empty, otherwise false
Resource Attribute (list type)	is not empty	false if list is empty, otherwise true
Request Attributes or Session Attributes (non-list type)	is empty	false if attribute/parameter exists and value is not null; true if attribute/parameter does not exist or value is null

Table 344. Examples of `Is Empty` or `Is not Empty` evaluations (continued)

One side of Evaluation	Evaluation	Result
Request Attributes or Session Attributes (non-list type)	is not empty	true if attribute/parameter exists and value is not null; false if attribute/parameter does not exist or value is null
Request Attributes or Session Attributes (list type)	is empty	true if attribute/parameter does not exist or list is empty; false if attribute/parameter exists and list has data
Request Attributes or Session Attributes (list type)	is not empty	false if attribute/parameter does not exist or list is empty; true if attribute/parameter exists and list has data

Profiler evaluations

If you choose to evaluate a profiler instead of a resource attribute in the Specify a Resource Attribute window, the available evaluations are:

- is
- is all of
- is any of
- is not
- is not any of

On one side of the evaluation, the possible choices are the profiles that are defined within that profiler. You may select one or more profiles for the result of the evaluation.

order as is:

Order as is is used to specify the order you want selected content to be returned and used. The default, order as is, will return data in the order it is stored in the repository. By clicking the order as is link, you can also choose **order randomly** or **order by**.

Order by allows you to sort content by any of its attributes, sort by more than one attribute (and specify the order the attributes are used to sort), and specify whether you want each attribute in ascending or descending order. Order randomly returns data in a different order each time the rule executes.

Profile:

Profile is an arbitrary name (of your choice) that provides information about the Web site visitor, the date and time the visit occurs, or other circumstances or conditions.

To better understand profiles, consider an example. Suppose you want to differentiate your Web visitors according to whether they are able to view confidential information. You might use two profiles in this scenario: Confidential and Regular.

When you create a profile within a profiler, type an arbitrary, but descriptive name. Be as accurate as possible to avoid duplication or confusion with other profiles.

When selecting a profile (for example, within a binding), you choose from a list of available profile names that match the profiler rule.

Profiler:

Profiler is a choice within the binding rule structure, and within the Specify a Resource Attribute window when constructing a profiler. Within a binding, you identify specific actions to perform that are based on the one profile within a profiler that evaluates as true.

You may also select quick profiler and specify a simple evaluation in-line instead of using a separate profiler.

Selecting **Profiler** within a profile allows you to identify other profilers (in effect, nesting profilers within one another) in order to create a profiler that can evaluate as true when multiple profilers or other evaluations are true.

Quick Profiler:

Quick profilers are created within bindings to perform simple evaluations. By using a quick profiler, you can avoid creating simple profilers as separate rules.

When you select the profile link within a binding and choose the quick profiler option, the structure for the line will change to an evaluation. The subsequent line of the binding will be the action to perform if the quick profiler evaluates as true.

sender:

sender must be a valid email address, list of email addresses (comma separated), or a resource attribute containing valid email addresses to whom the email will be sent.

set to:

Learn about the **set to** action and alternatives to **set to**.

Set to is the default action within an update action rule. Set to will modify the attribute of a resource, request object, or session object according to the value you specify in the expression.

Alternatives to set to include:

- append
- decrement by
- divide by
- increment by
- multiply by
- prepend
- remove
- remove all

value:

`Value` is the placeholder for the result of an evaluation. This value can be one you enter, the value of another resource attribute, or an arithmetic expression.

The value must be compatible with the data type of the other side of the expression or evaluation. For example, if you are evaluating an attribute that has the type Number, you can only compare it to resource attributes of the type Number or Decimal Number. The rule editor prevents you from choosing other resource attributes with incompatible types.

Note: Making comparisons against resources in a database respects the column type and size. Therefore, to compare a value to a column typed as CHAR(10), you must include all 10 characters. For example, assume you have a table with a column named DAY that is typed as CHAR(10). A row in the table has the value of 'Monday ' rather than 'Monday' in the DAY column because DAY is compared against a profiler condition, and must have all 10 characters defined. However, if the column is typed as VARCHAR, the value in the profiler condition can be 'Monday' (without the four additional blanks).

Mapped values

Resources may be created using mapped values instead of actual values specified in the data store. This facilitates the creation of rules that are easier to understand. For example, if a column in the database held the integer values of 1, 2, or 3 indicating Yes, No, or Maybe, the resource can be configured to map integer values to words. If mapped values have been created for a resource, the mapped values will be used in the rule editor instead of the actual values. For more information on creating value mappings, refer to the documentation in Rational Application Developer for creating resources using the Personalization resource wizard.

Dynamic properties

In addition to predefined resource properties, you can enter properties of a resource that are not in the list. If you know the resource to handle dynamically, specify the name of the property. If the resource manages properties dynamically, the values are retrieved when the rule is evaluated.

Email:

View the prerequisites for creating an email action or promotion within the Personalization workspace.

An email promotion is an email message automatically sent to a defined list of recipients by a running rule. Email promotions can be sent once or repeatedly on regular intervals. The body of the email message is derived from a file on the server, such as a text file, an HTML file, or even a JSP containing content spots for rules. This file can be chosen from the authoring repository. The list of recipients can be derived dynamically from a rule. Email promotions are implemented by using a rule event to trigger an email rule on a schedule.

Before you can create an email action or promotion within the Personalization workspace, you need the following:

- A user resource: This resource contains information about potential email recipients, and can be created with the Personalization resource wizard in Rational Application Developer. The resource must include a string that represents the email address of the recipients (in the form "username@domain"). The string must be published to the workspace server and to the run time environment.

- An email body: The body of the email is a flat text, HTML or JSP file that must exist on a server accessible from the run time environment. Typically, this file is created in Rational Application Developer. The location of the file in the run time environment must be specified as a URI when you create the email rule. The email body HTML or JSP page must be on the server, which is sending the email.
- An email rule: The email rule specifies to whom the email must be sent, who is sending it, and identifies the email body as a URI.
- A rule to determine recipients (for emails that are triggered on a schedule or periodically, optional for email actions): This rule can be a select content action or a binding, but must return a collection of recipients from the user resource that was previously mentioned. The action or rule you create becomes an option of the **To** list. If your actions or rules are not properly defined, the **To** list displays "No Matching Rules".

  It is possible to create email actions or promotions that are sent to a predefined address or list of addresses. This task is done by typing them into the **To** field when you create the email action or promotion.
- A rule event (for emails that are triggered on a schedule or periodically): The rule event binds the email rule to the rule, which determines the recipients. The rule event says that at a given time or times in the future, a specific rule must be run once for each user in a list. That list is determined by the rule to determine recipients.

For more information about configuring email activities, see *configuring email activity accounts*.

Email administration:

View the steps you need to complete before your run time server can send personalized email.

Before your run time server can send personalized email:

- Verify that the `pznscheduler.ear` is as an Enterprise Application.
- Have a properly configured and operating SMTP email server.

  JavaMail provides the SMTP required to send email. You can manage email responses from customers and outgoing error conditions (such as an unknown email address) using a standard email client.
- Configure a Mail Provider using the WebSphere Integrated Solutions Console.
  1. Click **Resources** > **Mail** > **Mail Providers**.
  2. Create a mail session to use with Personalization. By default, Personalization looks for a mail session in `mail/personalizationMailSession/jndi`. The JNDI name used is configurable in the `PersonalizationService.properties` file if you want to use Personalization with an existing mail session you have configured. If you are creating a new mail session, you must specify a Mail Transport Host. This is the mail server Personalization uses to send email. If your mail server is secured, you must specify a Mail Transport User ID and Mail Transport Password.
  3. Restart the server on which the email rules execute for the changes to take effect.
- Additional configuration is available through the `PersonalizationService.properties` file. Using this file, you configure how

often Personalization checks for rule events that run scheduled or repeating emails. You can specify what port an email rule will try to connect to when retrieving the bodyuri of the rule. You can also configure the mail session being used.

Content spots:

A content spot is a placeholder or slot for a rule on a Web page. When the page is viewed, the content spot uses its rule mapping to determine which rule to execute. When the rule is executed, any actions defined within the rule take place. Each content spot has a unique name. A content spot's content type must be defined when it is created and should not be changed.

Content spots are created by developers using the Content Spot wizard in Rational Application Developer and also in the Personalization workspace by selecting **New > Content Spot**. After creating the spot, the developer can place it on a JSP, or invoke it programmatically from any Java class.

To make the spot available to the Personalization engine, it should be on the classpath of any application which invokes it. If you are using a portlet or Web project in Rational Application Developer, the classpath information updates automatically when you deploy the application.

To make the content spot available to the Personalization authoring portlets, the content spot must be created in the workspace by selecting **New > Content Spot**. The name given to the content spot in the authoring portlet should match the display name given to it in the Rational Application Developer wizard or the name by which it is invoked using the com.ibm.websphere.personalization.ContentSpot programming interface. Content spots may be placed into folders by either using a display name which fully qualifies this folder, or by setting the execution scope to match the folder at runtime. So, if you want your content spot to be called `MyDataSpot` in a folder called `ProductData`, then the content spot's display name should be specified in the wizard as `ProductData/MyDataSpot`.

Users of the Personalization workspace specify which rule to place in a content spot. This is also known as mapping the rule to the content spot, or creating a rule mapping. When finished, a workspace user with authority to publish rules and rule mappings publishes them from the workspace server to the run-time environment. Publishing is optional, and is used to move objects between servers. Content spots, rules and all other objects created in Personalization are live as soon as they are created. Rule mappings can be changed at any time and are effective immediately upon publication, or upon the rule mapping start date, whichever comes later. Rule mappings expire on the rule mapping end date.

Content spots can be accessed in the workspace through the Personalization authoring portlets. You can see a list of all the content spots in the project, along with their content type and the name of their mapped rules, by navigating the browser view.

Use of content spots is optional. The com.ibm.websphere.personalization.ContentSpot class can be used to directly execute a rule by the rule name.

Rule spot mappings:

For a rule to be used on your Web site, it must be mapped to an existing content spot. A rule spot mapping is merely an association between a content spot and a rule. Changing the rule that is executed in the run-time environment is as easy as mapping a different rule to a content spot.

You create rule spot mappings within the Personalization workspace. There are two views for rule mappings: **Rule Mappings by Campaign** and **Rule Mappings by Content Spot**. The **Rule Mappings by Campaign** view shows all the rule mappings under a selected campaign from a drop-down menu, and their mapped content spots and rules. This view includes the Default Mappings option which simply shows all the default mappings of each content spot. The **Rule Mappings by Content Spot** view shows all the rule mappings under a particular content spot, including the default mapping and any mappings under campaigns. You can change the personalization behavior of your Web site by mapping a different rule to a given content spot.

Although only one rule is executed when a content spot bean is invoked, you can have multiple rules simultaneously mapped to a content spot by using campaigns. When you create a campaign, you can create a separate set of rule spot mappings for any or all of the content spots in your project.

When multiple campaigns are simultaneously active, campaign priorities and splits are used to determine the rule to execute. When multiple active campaigns have the same priority, the splits are used to calculate a percentage chance that one mapping will be used instead of the others.

Splits can be changed for each rule spot mapping.

Rule spot mappings can be duplicated and moved from one campaign to another. The start and end date of the rule spot mapping may both be modified if they fall outside the range of dates for the campaign to which the spot mapping is moved. Multiple mappings can be added to the same spot within a campaign.

Campaigns:

Campaigns are a means of organizing and implementing sets of personalization behavior. A useful analogy is an advertising campaign, which targets specific audiences with high-priority information for a specified period of time. Campaigns achieve this by allowing you to preferentially display campaign-related content in the content spots of a Web site. To accomplish such a goal, a campaign contains a set of rule-to-content spot mappings, start dates, and stop dates.

Users can create and manage campaigns through the Personalization Authoring Portlet. Campaigns are live as soon as their start date is reached and they may be published to other servers together with rules. To create a campaign, select **New** > **Campaign**. To add rule mappings to a campaign, select the campaign, and select **New** > **Rule Mapping**.

When a campaign is active in the run-time environment, the rule mappings take precedence over the default rule mappings for content spots the campaign references. For example, a seasonal campaign might contain certain rule mappings that result in the display of special offers to a Web site visitor. A campaign can contain rule mappings for some or all of the content spots on a site.

It is possible to have multiple campaigns active simultaneously. When this happens, the priority settings of the active campaigns dictate which campaign's rule mapping will be used. The campaign with the highest priority 'wins' and its rule mappings are used. In the event that multiple active campaigns have the same priority setting, the rule mapping used for a given content spot is determined randomly according to the relative split ratios.

Application object:

An application object is a java object existing at a known location in the request context. Defining an application object involves specifying the object's class name (as a Java class), and specifying a key (string key into a session attribute) to find it in the request context. Personalization calls methods on these objects during rule execution and uses their results in its decision making. Application Objects that implement the SelfInitializingApplicationObject interface are automatically instantiated as needed by Personalization.

"Device application object" on page 2274
The Device application object displays content by device class or by location (city, country, latitude, and longitude).

"Referrer application object" on page 2284
You can target your content to users based on what site the user came from before the user accessed your site. You can also target your content by the search parameters that are entered by users in a search engine.

"Public Render Parameters application object" on page 2288
The Public Render Parameter application object provides read and write access to public render parameters. Public render parameters allow portlets to share navigational state information and preserve this information across requests. Use this object in rules when you need your site to store a user's selection as a public render parameter during the action phase and have other portlets read this public render parameter during the render phase.

"Shared Data application object" on page 2291
The Shared Data application object is used in rules to share complex data between web applications and IBM WebSphere Portal Express in a user's session.

Installed application objects:

The Device, Referrer, Public Render Parameter, and Shared Data application objects are installed by default. By using installed application objects, you can skip the process of defining and registering application objects.

When you use application objects that are not installed, you must define the application objects by using a set of Personalization wizards that are provided with IBM Rational Application Developer or develop application objects according to a set of public programming interfaces. After you define the application objects, the application objects are registered to the Personalization server through the Personalization browser. When you use the installed application objects, you do not have to define or register the application objects.

The Device, Referrer, Public Render Parameter, and Shared Data application objects are installed and enabled with 8.5.

Optional: To enable the Content Targeting Dialog in virtual portals, use XML access to manually create the hidden Content Targeting Dialog page for each virtual portal. From the *wp_profile_root*/ConfigEngine directory, run the following task:

- Linux : ./xmlaccess.sh -in *PortalServer_root*/pzn.ui/wp.pzn.ui.actions/config/templates/DeployPages.xml -url http://*localhost*:10039/wps/config/*your_virtual_portal_context* -user *admin_user_id* -password *admin_password*
- IBM i: xmlaccess.sh -in *PortalServer_root*/pzn.ui/wp.pzn.ui.actions/config/templates/DeployPages.xml -url http://*localhost*:10039/wps/config/*your_virtual_portal_context* -user *admin_user_id* -password *admin_password*
- Windows: xmlaccess.bat -in *PortalServer_root*\pzn.ui\wp.pzn.ui.actions\config\templates\DeployPages.xml -url http://*localhost*:10039/wps/config/*your_virtual_portal_context* -user *admin_user_id* -password *admin_password*

The following topics contain an overview of the installed application objects. This overview includes descriptions and examples of using the application objects in rules that you create. To select the attributes that are used in the examples, you must enable the application objects. Instructions for enabling the location attributes associated with the Device application are also included in this section.

Device application object:

The Device application object displays content by device class or by location (city, country, latitude, and longitude).

"Device application object: Device class attribute"
Site content can display differently for a smartphone, desktop, or other device by using the Device application object in rules. You do not have to restrict your content to fit across all devices. Use this object to enable your site to recognize the device that accesses your site and to target content that best displays for that device.

"Device Application Object: Location attributes" on page 2279
You can use location to tailor the content that displays. The Device application object contains five location attributes that you can use in rules that you create: latitude, longitude, country, state, and city.

Device application object: Device class attribute:

Site content can display differently for a smartphone, desktop, or other device by using the Device application object in rules. You do not have to restrict your content to fit across all devices. Use this object to enable your site to recognize the device that accesses your site and to target content that best displays for that device.

"Defining your device classes"
When you target content to a segment based on device classes, ensure that the device classes are installed and enabled in portal.

"Example: Creating a segment by device class" on page 2277
Learn more about the Device application object through an example. In this example, you are creating two segments by using a profiler rule: Smartphone User and Tablet User. Each segment views content through different devices.

Defining your device classes:

When you target content to a segment based on device classes, ensure that the device classes are installed and enabled in portal.

About this task

By default, the smartphone and tablet device classes are already defined for you. You can use these device classes in rules that you create. If your mobile device is not recognized by using the default smartphone or tablet device classes, you can add your device as a client to these default classes. To add your device go to **Add new client**. Click the **Administration menu** icon. Then, click **Portal Settings** > **Supported Clients**. Then, click **Add new client**. You can also create an XML script to add your device.

When you add your device to the default classes, your device inherits and uses the responsive web design theme when it displays content. You can import the XML file. Click the **Administration menu** icon. Then, click **Portal Settings** > **Import XML**. You can also run the XML access directly to import the XML file.

You can also create custom device classes rather than using the default classes. Clients that are assigned to custom classes use the desktop theme. For custom classes to have a responsive theme based on the type of device, you must develop custom themes for your class. For more information, see the *Responsive Web Design* section.

To add a client to a device class, you need, at a minimum, the following information about your device:

- User-agent pattern
- Manufacturer
- Model
- Client capability

Use the table to learn more about the options available to you for defining device classes.

Table 345. Define device classes to use later with the Device application object in rules that you create. Select one of the following options to define device classes.

Option	Description
Add client definitions to already defined device classes. The smartphone and tablet device classes are examples of device classes that are provided to you by default.	1. Click the **Administration menu** icon. Then, click **Portal Settings** > **Supported Clients**.Then, click **Add new client** to enter information about the client. **Note:** Refer to the product help for more detailed instructions for adding new clients. 2. In the **User Agent** field, enter the name of the client. For example, enter `*.useragent.*` 3. In the **Manufacturer** field, enter the name of the company that manufactured the client. 4. In the **Model** field, enter the model number or name of the client. 5. In the **Capabilities** field, enter the specific capabilities of the client, and click **Add**. To add tablet capabilities, enter the following information: `com.ibm.portal.devicesupport.deviceclass=tablet HTML_4_0` To add smartphone capabilities, enter the following information: `com.ibm.portal.devicesupport.deviceclass=smartphone HTML_4_0` 6. From the **Position** menu, select the order in which you want this client to be entered in the client registry. It is recommended to place the most specific user agent patterns at the beginning of the list. 7. Click **OK** to save the new client that you added.
Create client definitions, and add to existing device classes.	You can also create client definitions through an XML file that contains the device information. For more information, see the *Device classes* topics in the *Developing Themes and Skins* section for more information and examples.
Create custom class definitions, and add client information to your class definitions.	Class definitions might also be created through an XML file that contains the class information. After you create your device class, you must add appropriate clients by using XML. You might need to create a custom theme for this class. For more information, see the *Device classes* topics in the *Developing Themes and Skins* section for more information and examples.

Example: Creating a segment by device class:

Learn more about the Device application object through an example. In this example, you are creating two segments by using a profiler rule: Smartphone User and Tablet User. Each segment views content through different devices.

About this task

Procedure

1. Click **Applications** > **Personalization** > **Business Rules**.
2. From the Personalization Navigator, click **New** > **Rule**.
3. For Rule Type, select **Profiler**. The profiler rule groups a user into one or more segments. Segments are profiles.
4. The numbered screen capture, along with the corresponding table, provides the values and selections that are used in this example. Use these example values and selections to guide you in creating a profiler rule by using the device class attribute.

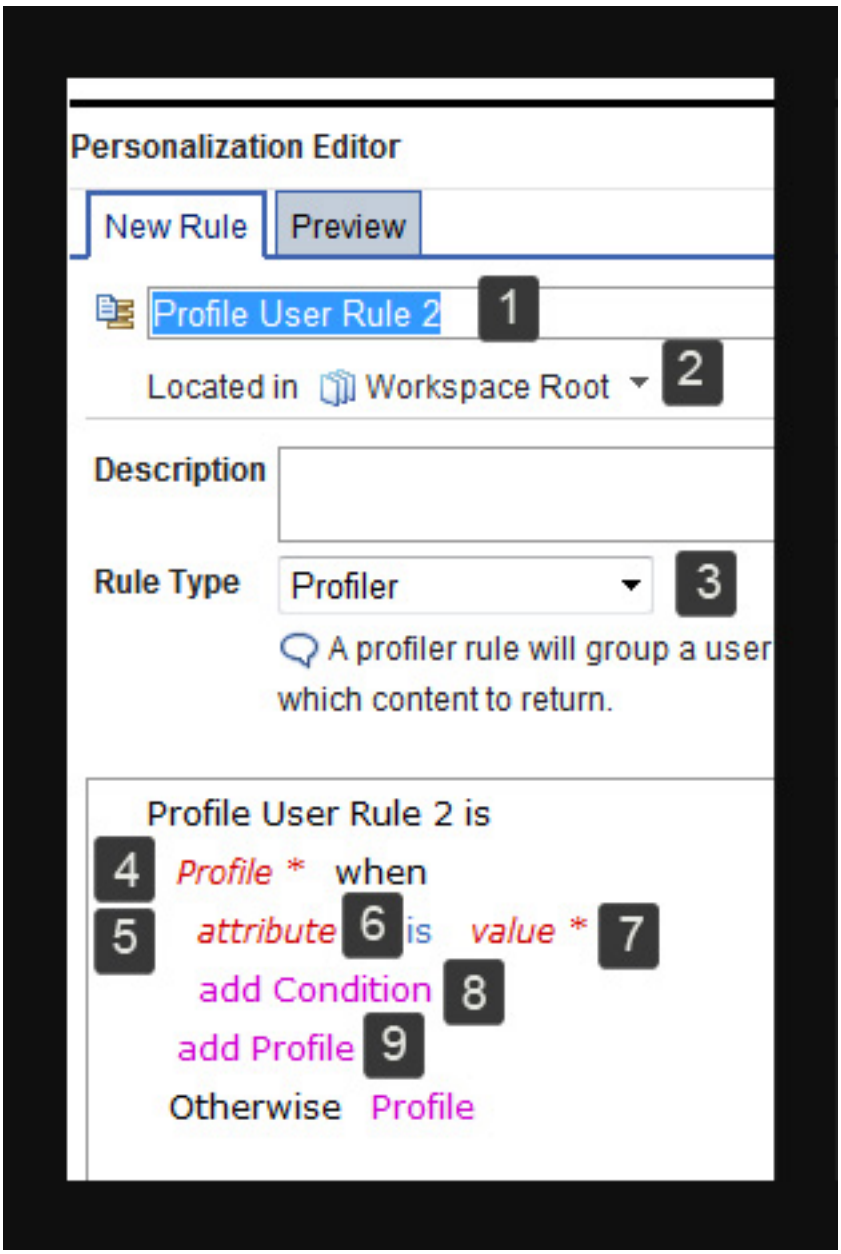


Table 346. Creating segments (profiles) in a profiler rule

Numbered item in screen capture	Description
1	Type `User by device` as the name for your rule.
2	By Located in, use the default folder to store your rule for this example.
3	For Rule Type, select **Profiler**. A segment is another word for profile in the profiler rule.
4	Click **Profile**, and enter `Smartphone User` as the name for your segment. Click **Submit**.

Table 346. Creating segments (profiles) in a profiler rule (continued)

Numbered item in screen capture	Description
5	In this example, you are defining a segment that is based on device class used. Define attributes and values for the Smartphone User segment. Click **attribute**, and select **Device** > **Device Class**. The attribute label changes to current Device.Device Class.
6	In this example, continue to use **is** as the comparison operator.
7	Click **value**, and select **smartphone** as the value for the condition.
8	In this example, you are not setting up another condition for the Smartphone User. Skip to add Profile.
9	Click **add Profile** to add another segment to the profiler rule. Enter `Tablet User` as the name for your segment. Click **Submit**.
Not shown in screen capture	Define attributes and values for the Tablet User segment. Click **attribute**, and select **Device** > **Device Class**. The attribute label changes to current Device.Device Class.
Not shown in screen capture	In this example, continue to use **is** as the comparison operator.
Not shown in screen capture	Click **value**, and select **tablet** as the value for the condition.

5. Click **Save**.

Results

The segments in this saved profiler rule appear in the Add Segment view when you create targeting rules. You can add these segments to content in a targeting rule or use these segments in other types of rules.

Device Application Object: Location attributes:

You can use location to tailor the content that displays. The Device application object contains five location attributes that you can use in rules that you create: latitude, longitude, country, state, and city.

About this task

To receive latitude and longitude information from a user's device:

- You must enable the geolocation theme module on IBM WebSphere Portal Express.
- The user that accesses your site must use a device that is location aware and selected to share location information about their device to your site.
- The client software on the device (the browser) must support HTML5.

Country, state, and city attributes are resolved from the latitude and longitude coordinates of the user's device or from the IP address that is obtained by using a reverse geocoding service. A reverse geocoding service is not provided with WebSphere Portal Express. Use the geolocation interface and registration mechanism that is provided with WebSphere Portal Express to define how to deploy the service you use.

"Enabling the geolocation theme module"
Unlike the other application objects that are installed by default, the geolocation features of the Device application object are not enabled after you run the ConfigEngine **enable-content-targeting-extension** task. To enable the geolocation features, you can modify the JSON profiles to include the geolocation theme module, provide a resolver JAR, or use both methods.

"Retrieving the location of users or devices" on page 2281
To implement the geolocation interface, you must provide methods to resolve coordinates or IP address to one or more of the location attributes (country, state, or city). You can determine the location of a device by using reverse geocoding or by using service providers. Reverse geocoding converts latitude and longitude coordinates into a location (such as country, state, or city) by using a geolocation JAR that you provide or by using an external geolocation service provider.

"Example: Creating a segment that uses the state attribute" on page 2282
Learn more about the Device application object through an example that uses a location attribute (state). In this example, you are creating two segments by using a profiler rule: North Carolina and California. Each segment views content that is based on the location of the device.

Enabling the geolocation theme module:

Unlike the other application objects that are installed by default, the geolocation features of the Device application object are not enabled after you run the ConfigEngine **enable-content-targeting-extension** task. To enable the geolocation features, you can modify the JSON profiles to include the geolocation theme module, provide a resolver JAR, or use both methods.

About this task

The new geolocation theme module adds JavaScript to every portal page to check that the HTML5 geolocation API is available on the client. When the HTML geolocation API is available, the latitude and longitude of the device that accesses portal is returned to the portal.

Procedure

1. Connect to WebDav client, and go to the `/profiles` directory. For example, follow this path to the `/profiles` directory: `http://server_name/wps/mycontenthandler/dav/themelist/ibm.portal.85Theme/profiles/`
2. By default, four JSON profiles are in this directory. Add the Geolocation theme module ID to the moduleIDs section of every JSON file. The value for the Geolocation theme module ID is `wp_pzn_geolocation`. For example:

```
"moduleIDs": [
  "wp_theme_portal_85",
  "wp_pzn_geolocation",
  "wp_portlet_css",
  "wp_one_ui",
```

```
"wp_one_ui_dijit",
"wp_legacy_layouts",
"wp_client_ext",
...
```

3. Use the Theme Opt Analyzer portlet to invalidate the cache. Click the **Administration menu** icon. Then, click **Portal Analysis** > **Theme Analyzer**. Then, click **Utilities** > **Control Center** > **Invalidate cache**. CF06 Auto invalidation recognizes your changes automatically for WebDAV based themes. No further action is required. For more information, see Utilities. You do not need to restart the server.

Retrieving the location of users or devices:

To implement the geolocation interface, you must provide methods to resolve coordinates or IP address to one or more of the location attributes (country, state, or city). You can determine the location of a device by using reverse geocoding or by using service providers. Reverse geocoding converts latitude and longitude coordinates into a location (such as country, state, or city) by using a geolocation JAR that you provide or by using an external geolocation service provider.

About this task

Key concepts for determining the location of a device:

- The new geolocation theme module checks to see that the HTML5 geolocation API is available on the client. When the HTML geolocation API is available, the latitude and longitude of the device that accesses your site is returned to the portal.
- When latitude and longitude information is returned to your portal, Personalization calls the API method `getAddressFromCoords()` to return the corresponding country, state, or city information.
- If latitude and longitude are not returned to portal, Personalization calls the API method `getLocation()`. This method typically resolves the IP address to a country, and other location attributes are not set. Examples of when latitude and longitude are not available are as follows:
  - The client does not support HTML5. For the location-based services to work correctly, the clients (browsers) must support HTML5.
  - The user does not want to share location information about their device.
  - The device does not have location awareness.
- You can also use the geolocation theme to add JavaScript. When the latitude and longitude coordinates are received on the client, these coordinates are resolved to country, state, or city location information that uses a call from the client to an external resolver service. The location attributes (latitude, longitude, country, state, or city) are returned to your portal. The `getLocation()` method resolves the request parameters and returns these parameters.

Procedure

1. Enable the reverse geocoding class in the `PersonalizationService.properties` file to receive the latitude and longitude coordinates of the client that accesses your portal.
   a. Locate the `PersonalizationService.properties` in the following directory: *wp_profile_root*`/PortalServer/config/config/service`
   b. Create a backup copy of the `PersonalizationService.properties` file.
   c. Add the class name to the key. For example:
   `pzn.externalGeolocation=com.acme.geolocation.ReverseGeolocationExample`

2. Create a JAR file, and place this file on the portal server class path. For example, place your JAR file in the following directory: *PortalServer_root*/shared
3. Restart the WebSphere Portal Express server.

Example: Creating a segment that uses the state attribute:

Learn more about the Device application object through an example that uses a location attribute (state). In this example, you are creating two segments by using a profiler rule: North Carolina and California. Each segment views content that is based on the location of the device.

About this task

Procedure

1. Click **Applications** > **Personalization** > **Business Rules**.
2. From the Personalization Navigator, click **New** > **Rule**.
3. For Rule Type, select **Profiler**. The profiler rule groups a user into one or more segments. Segments are profiles.
4. The numbered screen capture, along with the corresponding table, provides the values and selections that are used in this example. Use these example values and selections to guide you in creating a profiler rule that uses the device class attribute.

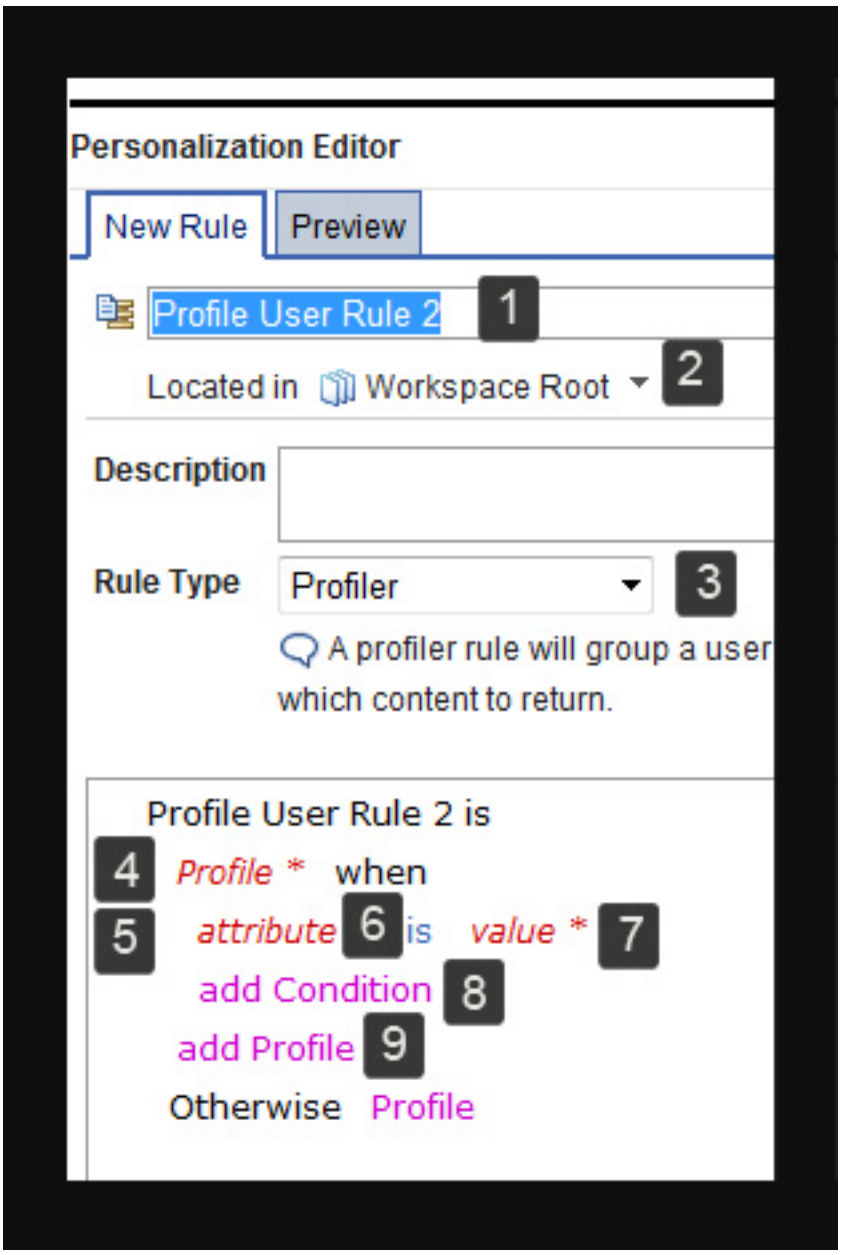


Table 347. Creating segments (profiles) in a profiler rule

Numbered item in screen capture	Description
1	Type `Device segments by state` as the name for your rule.
2	By Located in, use the default folder to store your rule for this example.
3	For Rule Type, select **Profiler**. A segment is another word for profile in the profiler rule.
4	Click **Profile**, and enter `North Carolina` as the name for your segment. Click **Submit**.

Table 347. Creating segments (profiles) in a profiler rule (continued)

Numbered item in screen capture	Description
5	In this example, you are defining a segment that is based on the state the device is located in. Define attributes and values for the North Carolina segment. Click **attribute**, and select **Device** > **State**. The attribute label changes to current Device.State. **Note:** • You can also create other segments that are based on location by selecting Country, Latitude, or Longitude.
6	In this example, continue to use **is** as the comparison operator.
7	Click **value** to enter `North Carolina` as the value for the condition. Click **Submit**.
8	In this example, you are not setting up another condition for the North Carolina segment. Skip to add Profile.
9	Click **add Profile** to add another segment to the profiler rule. Enter `California` as the name for your segment. Click **Submit**.
Not shown in screen capture	Define attributes and values for the California segment. Click **attribute**, and select **Device** > **State**. The attribute label changes to current Device.State.
Not shown in screen capture	In this example, continue to use **is** as the comparison operator.
Not shown in screen capture	Click **value** to enter `California` as the value for the condition. Click **Submit**.

5. Click **Save**.

Results

The segments in this saved profiler rule appear in the Add Segment view when you create targeting rules. You can add these segments to content in a targeting rule or use these segments in other types of rules.

Referrer application object:

You can target your content to users based on what site the user came from before the user accessed your site. You can also target your content by the search parameters that are entered by users in a search engine.

About this task

The Referrer application object supports the following search engines:

- Google
- Yahoo
- AOL

- Ask
- Bing

  "Example: Creating a visibility rule that uses the referral host and search attributes"
  Learn more about the Referrer application object through an example. In this example, you are creating a visibility rule that uses the referral host and search attributes. These attributes are only used in rules to display ski related vacation information for users that search on ski trips in Google.

Results

Example: Creating a visibility rule that uses the referral host and search attributes:

Learn more about the Referrer application object through an example. In this example, you are creating a visibility rule that uses the referral host and search attributes. These attributes are only used in rules to display ski related vacation information for users that search on ski trips in Google.

About this task

This rule prevents other types of information, such as beach or mountain vacation information, from displaying when users search on ski trips.

Procedure

1. Click **Applications** > **Personalization** > **Business Rules**.
2. From the Personalization Navigator, click **New** > **Rule**.
3. For Rule Type, select **Visibility Rule**.
4. The numbered screen capture, along with the corresponding table, provides the values and selections that are used in this example. Use these example values and selections to guide you in creating a visibility rule that uses the referral host and search attributes.

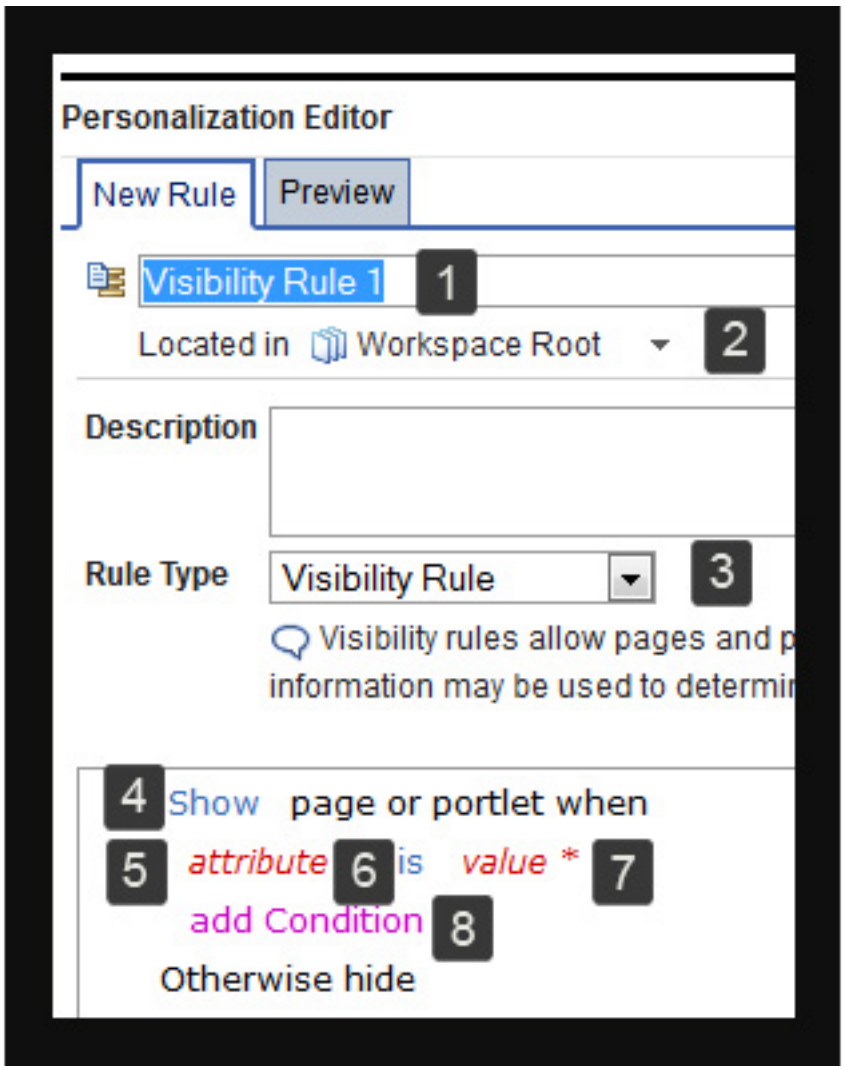


Table 348. Creating a visibility rule that uses the referral host and search attributes for the Referrer application object

Numbered item in screen capture	Description
1	Type `Ski vacation information` as the name for your rule.
2	By Located in, use the default folder to store your rule for this example.
3	For Rule Type, select **Visibility Rule**.
4	Continue to use **Show** in this example to show a page or portlet that is based on the conditions that you define in this rule.
5	Click **attribute** > **Referrer** > **Referral Host**. The attribute label changes to current Referrer. Referral Host. **Note:** The Referral URL attribute is another attribute available to you with the Referrer application object. Use the Referral URL attribute when a user is referred to your site from a search engine other than the supported search engines. Use the Referral Host attribute, as shown in this example, when a user is referred to your site from the supported search engines.
6	In this example, continue to use **is** as the comparison operator.

Table 348. Creating a visibility rule that uses the referral host and search attributes for the Referrer application object (continued)

Numbered item in screen capture	Description
7	Click **value** * to enter `www.google.com` as the value for the condition. Click **Submit**.
8	Click **add Condition** > **Referrer** > **Search Keywords**. The attribute label changes to current Referrer. Search Keywords. **Note:** You can provide a value for the Referrer. Search Keywords only when a user is arriving to your site from one of the supported search engines.
Not shown in screen capture	Click **value** * to enter `skiing` as the value for the condition. Click **Submit**.

5. Click **Save**.

You successfully created your visibility rule. You can add your rule to a page or portlet from the Manage Pages area of your site.

Note: Visibility rules do not hide pages or portlets for anonymous users.

6. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
7. Click **Content Root**.
8. Go to the page that you want to use your rule. For example, you might have a page that is named Ski Vacations in the Home area of your site. To locate the Travel page, click **Home** > **Ski Vacations**
9. After you locate your Travel page, follow the instructions for adding your rule to a page or for adding your rule to a specific portlet.

Option	Description
Instructions for adding your rule to a page	1. Click **Edit Page Properties**. 2. Expand **Advanced options**. 3. Click the arrow by Show or Hide page rule, and click **Select Rule**. 4. Select the check box by a rule you want to add to your page. Click **OK**. 5. Click **Done**
Instructions for adding your rule to a portlet	1. Click **Edit Page Layout**. 2. Make sure that Show portlet rule mapping is enabled. When this option is enabled, you see a Hide Portlet Rule Mappings link. 3. Click the arrow by **No rule mapped** for a portlet on this page, and click **Select Rule**. 4. Select the check box by a rule you want to add to your portlet. Click **OK**. 5. Click **Done**.

Public Render Parameters application object:

The Public Render Parameter application object provides read and write access to public render parameters. Public render parameters allow portlets to share navigational state information and preserve this information across requests. Use this object in rules when you need your site to store a user's selection as a public render parameter during the action phase and have other portlets read this public render parameter during the render phase.

About this task

You can write public render parameters from preprocessors and portlets (JSR 168, JSR 286, and legacy IBM portlets) during the action phase. Render parameters are read from anywhere, including preprocessors, themes, dynamic assembly transformation, and portlets, during all phases.

Let's apply an example to this definition to highlight how you can use the public render parameters application object in rules that you create on your site. For example,

- You select a travel destination from an airline website.
- Your selection is saved by a portlet as a public render parameter during the action phase.
- Portlets can read your destination during the render phase, and your destination is available in bookmarks that you create. For example, you have a portlet that provides ticket rates that are based on the destination selected.
- A visibility rule that uses the Public Render Parameters application object displays content in the site that is based on your destination selection.

Note: You cannot write public render parameters from themes or from update rules.

Use the table to learn more about the Public Render application object

Table 349. Feature highlights for the Public Render application object

Description	Public Render Parameter application object
Type of data stored	Strings and arrays of strings. Since these parameters are encoded in your portal URLs, it is recommended that you use string values that are not long in length.
Location of stored data	Public render parameters are encoded in all portal URLs. Public render parameters remain available when a user remains in the same browser tab or opens a new tab by using a link from the original tab.
Duration of storage	Data is not connected to a user session. Data is available in the same browser tab, window, or bookmarks.
Supported user types	You can use this object for all users. For example, supported user types include authenticated users and anonymous users, with and without sessions

Note: The Shared Data application object is also used in rules to share data between portal web applications. Use the Shared Data application object, instead of the Public Render Parameters application object, when you need to share complex data.

"Example: Creating a visibility rule that uses travel destination as your public render parameter"
Learn more about the Public Render Parameter application object through an example. In this example, create a visibility rule that uses your travel destination as the public render parameter used in the rule.

Example: Creating a visibility rule that uses travel destination as your public render parameter:

Learn more about the Public Render Parameter application object through an example. In this example, create a visibility rule that uses your travel destination as the public render parameter used in the rule.

Procedure

1. Click **Applications** > **Personalization** > **Business Rules**.
2. From the Personalization Navigator, click **New** > **Rule**.
3. For Rule Type, select **Visibility Rule**.
4. The numbered screen capture, along with the corresponding table, provides the values and selections that are used in this example. Use these example values and selections to guide you in creating a visibility rule that uses a public render parameter.

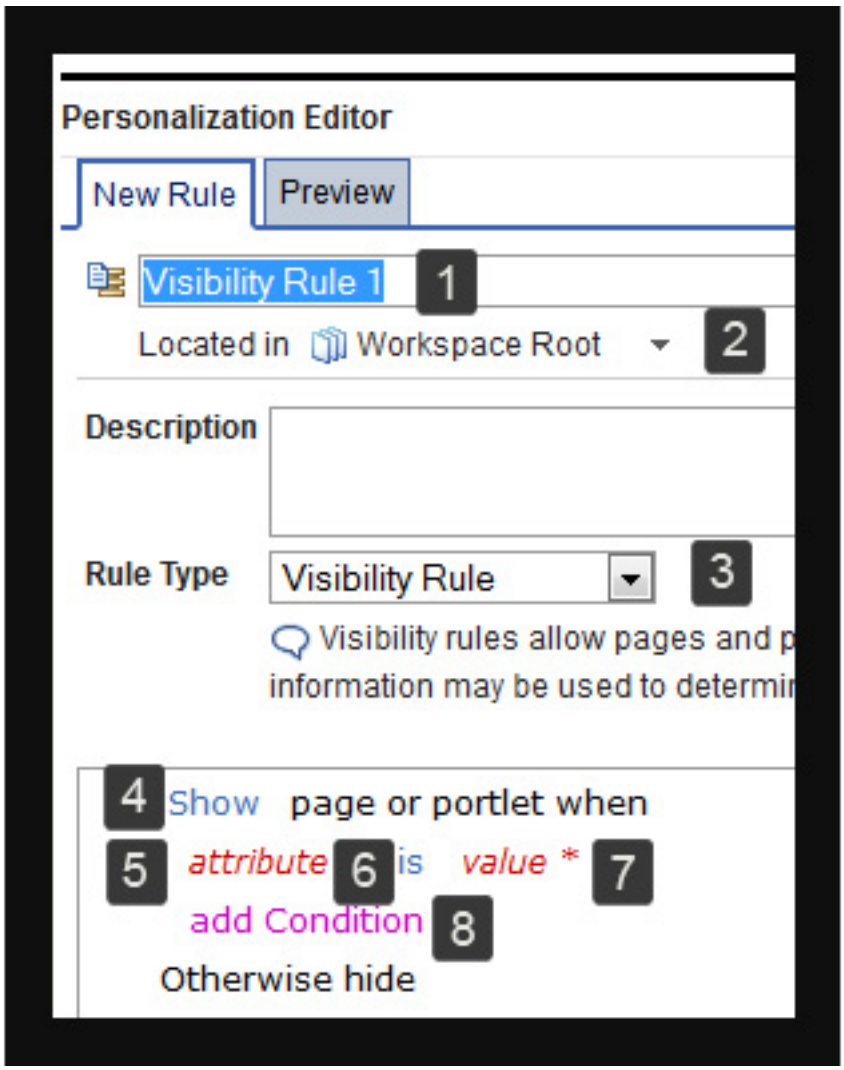


Table 350. Creating a visibility rule that uses travel destination as your public render parameter

Numbered item in screen capture	Description
1	Type `Show Shops in Paris` as the name for your rule.
2	By Located in, use the default folder to store your rule for this example.
3	For Rule Type, select **Visibility Rule**.
4	Continue to use **Show** in this example to show a page or portlet that is based on the conditions you define in this rule.
5	Click **attribute** > **Public Render Parameters** > **Travel Destination**. **Note:** • Travel Destination must exist as a parameter to select this option. • If Travel Destination does not exist, click **attribute** > **Public Render Parameters** > **Manage Properties** to create this parameter. • You must know the name of the parameter that the portlet or preprocessor is using to set the value for public render parameters in Manage Properties.

Table 350. Creating a visibility rule that uses travel destination as your public render parameter (continued)

Numbered item in screen capture	Description
6	In this example, continue to use **is** as the comparison operator.
7	Travel Destination is the public render parameter that is read by portlets on the site during your session and is used to display targeted content. Click **value** * to enter `Paris` as the value for the condition. Click **Submit**.
8	In this example, you are not setting up another condition.

5. Click **Save**.

You successfully created your visibility rule. You can add your rule to a page or portlet from the Manage Pages area of your site.

Note: Visibility rules do not hide pages or portlets for anonymous users.

6. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
7. Click **Content Root**.
8. Go to the page that you want to use your rule. For example, you might have a page that is named Travel to contain portlets that use the Travel Destination public render parameter in the Home area of your site. To locate the Travel page, click **Home** > **Travel**
9. After you locate your Travel page, follow the instructions for adding your rule to a page or for adding your rule to a specific portlet.

Option	Description
Instructions for adding your rule to a page	1. Click **Edit Page Properties**. 2. Expand **Advanced options**. 3. Click the arrow by Show or Hide page rule, and click **Select Rule**. 4. Select the check box by a rule you want to add to your page. Click **OK**. 5. Click **Done**
Instructions for adding your rule to a portlet	1. Click **Edit Page Layout**. 2. Make sure that Show portlet rule mapping is enabled. When this option is enabled, you see a Hide Portlet Rule Mappings link. 3. Click the arrow by **No rule mapped** for a portlet on this page, and click **Select Rule**. 4. Select the check box by a rule you want to add to your portlet. Click **OK**. 5. Click **Done**.

Shared Data application object:

The Shared Data application object is used in rules to share complex data between web applications and IBM WebSphere Portal Express in a user's session.

About this task

Use the table to learn more about the Shared Data application object.

Table 351. Feature highlights for the Shared Data application object

Description	Shared Data application object
Type of data stored	Any type of data.
Location of stored data	Encoded in all portal URLs.
Duration of storage	Data is tied to a user's session and is available only for the user's session.
Supported user types	This object is only used to store data for users with an active session.

Note: The Public Render Parameters application object is also used in rules to share data between portal web applications. Use the Public Render Parameters application object, instead of the Shared Data application object, when you are sharing string data that is not restricted to a user's session.

"Example: Creating a visibility rule to track your order history"
Learn more about the Shared Data application object through an example. In this example, create a visibility rule that uses your order history as the parameter used in the rule. A user would need to sign in to the site to view a summary of their order history and know the tracking number.

Example: Creating a visibility rule to track your order history:

Learn more about the Shared Data application object through an example. In this example, create a visibility rule that uses your order history as the parameter used in the rule. A user would need to sign in to the site to view a summary of their order history and know the tracking number.

Procedure

1. Click **Applications** > **Personalization** > **Business Rules**.
2. From the Personalization Navigator, click **New** > **Rule**.
3. For Rule Type, select **Visibility Rule**.

   Note: In this example, a parameter is set in an application and read in a Visibility rule. You can also use an Update rule to set your parameter for a Visibility rule to reference.
4. The numbered screen capture, along with the corresponding table, provides the values and selections that are used in this example. Use these example values and selections to guide you in creating a profiler rule that uses the device class attribute.

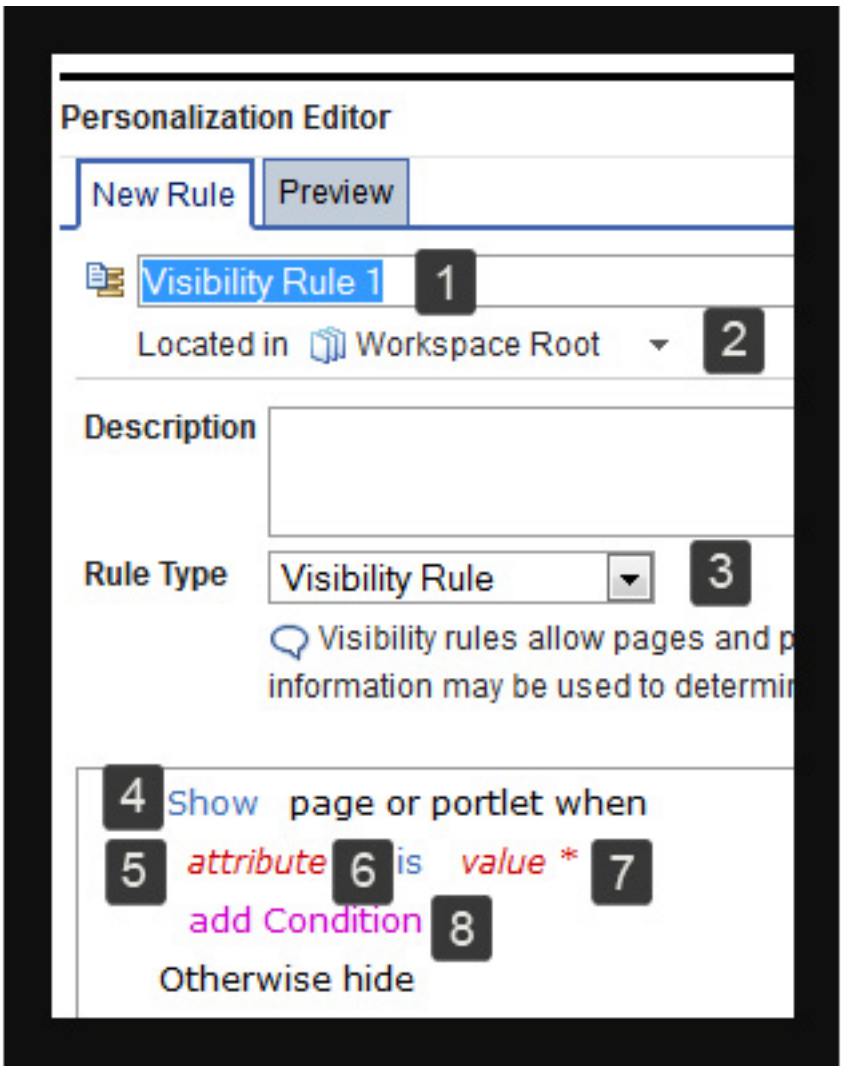


Table 352. Creating a visibility rule that uses order history as parameter used in the Shared Data application object.

Numbered item in screen capture	Description
1	Type `Show Order Tracking` as the name for your rule.
2	By Located in, use the default folder to store your rule for this example.
3	For Rule Type, select **Visibility Rule**.
4	Continue to use **Show** in this example to show a page or portlet that is based on the conditions you define in this rule.
5	Click **attribute** > **Shared Data** > **Order Number**. **Note:** • Order Number must exist as a parameter to select this option. • If Order Number does not exist, click **attribute** > **Shared Data** > **Manage Properties** to create this parameter. • You must know the name of the parameter that the portlet or preprocessor is using to set the value for parameters in Manage Properties.

Table 352. Creating a visibility rule that uses order history as parameter used in the Shared Data application object. (continued)

Numbered item in screen capture	Description
6	In this example, continue to use **is** as the comparison operator.
7	In this example, Order Tracking is the parameter that is read by portlets on the site during your session and is used to display targeted content. Click **value** * to enter `248761` as the value for the condition. Click **Submit**. **Note:** In this example, `248761` is your order number for a recent purchase.
8	In this example, you are not setting up another condition.

5. Click **Save**.

You successfully created your visibility rule. You can add your rule to a page or portlet from the Manage Pages area of your site.

Note: Visibility rules do not hide pages or portlets for anonymous users.

6. To open the **Manage Pages** portlet, click the **Administration menu** icon. Then, click **Portal User Interface** > **Manage Pages**.
7. Click **Content Root**.
8. Go to the page that you want to use your rule. For example, you might have a page that is named Orders in the Home area of your site. To locate the Orders page, click **Home** > **Orders**
9. After you locate your Travel page, follow the instructions for adding your rule to a page or for adding your rule to a specific portlet.

Option	Description
Instructions for adding your rule to a page	1. Click **Edit Page Properties**. 2. Expand **Advanced options**. 3. Click the arrow by Show or Hide page rule, and click **Select Rule**. 4. Select the check box by a rule you want to add to your page. Click **OK**. 5. Click **Done**
Instructions for adding your rule to a portlet	1. Click **Edit Page Layout**. 2. Make sure that Show portlet rule mapping is enabled. When this option is enabled, you see a Hide Portlet Rule Mappings link. 3. Click the arrow by **No rule mapped** for a portlet on this page, and click **Select Rule**. 4. Select the check box by a rule you want to add to your portlet. Click **OK**. 5. Click **Done**.

Request Context:

This is the interface used to access various attributes for rules. For HTTP contexts, it provides access to the HttpRequest and HttpSession attributes. For non-HTTP contexts, it provides the same interface to a surrogate for the request and session.

The request context, and any request values accessed via the request context, are only valid for the life of the request.

The Request Context string is used in caching lookups. The lookup is created by using a user-specified string combined with query values as the lookup key. The user-specified string should uniquely represent the current Request Context. This key is stored under ibm.wcp.cache.user.key as a request attribute or as a session attribute with the request attribute taking precedence.

Accessing the Request Context

The Request Context provides the Personalization rules engine with the data and environmental information needed for rules processing. In other words, the Request Context contains all the input to execute Personalization content spots and rules. This includes simple inputs like request and session data, and more advanced input like the user object.

You can access the Request Context from a content spot by first using the setRequest method of the content spot to back the content spot with a request, and then by calling getContext to retrieve the context. You can also use the Request Context to call directly into the ResourceDomain3 and ResourceManager3 APIs.

The Request Context allows you to retrieve session, request, portlet attribute, date, cookie, and other data and environmental information from the resource layer.

The Request Context includes:

- Session

  The session information identifies the HttpSession object that is associated with the current user.
- Request date

  This request date is the date the HTTP request was received. This information supports rules that have date-dependent actions.

Since Personalization uses the Request Context to contain all the rule input, the Request Context must be set onto the content spot prior to rule execution. The code with the content spot's useBean tag must be similar to:

```
<jsp:useBean id="gold_promo_bean"
class="yourco.goldpromo.BannerSpot" />
<% gold_promo_bean.setRequest (request); %>
```

In the previous section, the jsp:useBean tag constructs the yourco.goldpromo.BannerSpot class and stores an instance of that class into the local variable gold_promo_bean. The next line calls setRequest to put the HttpServletRequest or PortletRequest onto the newly constructed content spot bean. The content spot then implicitly constructs a Request Context which is backed by the given HttpServletRequest or PortletRequest. This Request Context then provides access to that request's parameters and attributes and attributes of the session through the com.ibm.websphere.personalization.RequestContext interface.

In some cases, it may be useful to call into a content spot without having access to an HttpServletRequest or PortletRequest. The interface com.ibm.websphere.personalization.PznRequestObjectInterface can be used in these situations. A implementation of this class called com.ibm.websphere.personalization.PznRequestObjectImplementor is provided for convenience.

Query framework:

The query framework is an object representation of a query. The framework is not specifically oriented toward querying either object or relational databases. A set of common operators is defined, but what an attribute represents is determined by the interpreter of the query.

Since the framework makes no assumptions about how the name of an attribute is translated to the object it represents, object trees and graphs, relational models, or almost any other data structure could be queried with the framework. The query framework is used to pass query information from the runtime engine into the resource engine. It is up to the resource collection to interpret the query. Traditionally, this interpretation of the query object is done through a callback class, which is essentially a translator from the query framework into a protocol-specific query language.

The Personalization interface

The Portal Personalization user interface consists of three portlets: the Personalization Navigator, Personalization Editor, and the Personalized List.

These portlets are automatically installed with IBM WebSphere Portal Express. Although personalization services can be used in portlets supporting remote WSRP services, the personalization portlets do not support being used as a remote WSRP service.

Personalization Navigator

The Personalization Navigator is the main navigation interface. Users explore the repository with a tree structure--directories display on one side of the window, and elements within a selected directory display in the other part of the window. The view can be modified to eliminate the side of the window with the tree structure, and instead list all directories and elements in a hierarchical list. A drop-down list enables users to show all elements in Personalization, or filter the view to display a single element type (such as rules or campaigns). The following table shows which views are available:

Table 353. Available views in the Personalization Navigator

List view	Description	Properties shown
Browse Personalization Resources (tree)	This tree view will show only artifacts created by Personalization. For example, rules, campaigns, resource collections.	Name, Creator, Last Modified, Node Type
Browse Rules (tree)	This tree view will show only Personalization rules. This view will also show the type of each rule; for example, Select Action or Profiler.	Name, Creator, Last Modified, Return Type, Rule Type

Table 353. Available views in the Personalization Navigator (continued)

List view	Description	Properties shown
Campaigns (list)	Campaigns will be a specialized list view. In the action bar for the view, there will be a scope drop down. The scope drop down will allow the user to pick an execution scope. The campaigns from the scope context of that execution scope will be shown. When the scope setting is global as by default, there will be no drop down and all campaigns will be shown. As with all views, the Edit Mappings action will be available when a campaign is selected.	Name, Priority, Split, Start Date, End Date
Rule Mappings by Campaign (list)	This view is launched either by the drop down for selecting a view, or by the Edit Mappings action for a campaign. The view shows a list of mappings for a given campaign.	Spot Name, Rule Name, Content Type, Split, Start Date, End Date
Collections (list)	This view will allow a user to see all resource collections and application objects created under a collection.	Collection Name, IBM Java Content Repository(True / False), Collection Type (User / Content)
Events	This view will allow a user to see all rule events on the system.	

In addition to navigation, the Personalization Navigator is where users control the repository content. Users can move, copy, import and delete elements, and create new elements. When users create a new element (by selecting **New** from the Personalization Navigator menu), they are automatically taken to the Personalization Editor.

You can also access the Resource Permissions portlet to set access control on individual personalization items from the Personalization Navigator by clicking **Edit Access** > **Extra Actions**.

Personalization Editor

This portlet is where users edit element content or information. Selecting a new element from the Personalization Navigator activates the edit mode, where users enter data, depending on the element chosen. To edit an existing element, users highlight the item in the Personalization Navigator. They return to the Personalization Editor and click **Edit** on the Edit item tab.

Personalized List

The Personalized List portlet allows a user to display personalized content without having to build a custom JSP portlet. Each portlet can display a list of resources and show details for each returned resource. Groups of related resources may be categorized for easy viewing. When a more detailed view of a piece of content is required, a custom detail JSP may be specified. Different instances of the portlet may be used across WebSphere Portal Express to quickly and easily deploy customized information to users.

Publishing personalization rules overview

WebSphere Portal Express Personalization sends published rules across HTTP to a servlet which resides on each personalization server. This servlet can receive publishing data or initiate new publishing jobs. When a user begins a publishing job from the personalization authoring environment, the local servlet is provided with the set of information necessary to complete the job. The local servlet contacts the destination endpoint servlet (which could be the same servlet) and sends its data to it. The destination servlet reports success or failure.

1. "Publishing considerations"
   If you are publishing personalization rules in a clustered server configuration, to an IPv6 host, or using Web Content resource collection classes, there are unique considerations that you should be familiar with.
2. "Publishing personalization rules" on page 2300
   After developing personalization rules, you publish the rules.
3. "Publishing and deleting personalization rules using a script" on page 2303
   You can use a WebSphere Portal Express Personalization provided script, pznload, to export, publish, and delete Personalization rules on local or remote servers. You can script the delivery of rules and campaigns from staging to production, or the offline publishing between disconnected systems (such as when production servers are secured behind a firewall). You can use this function to quickly revert production servers to an earlier state.
4. "Publishing personalization rules over SSL" on page 2305
   WebSphere Portal Express Personalization uses the built-in SSL capabilities of WebSphere Application Server to provide secure publishing across unprotected networks. Your personalized portal can benefit from the full range of authentication repositories supported by WebSphere Application Server security.
5. "Monitoring the status of publishing" on page 2307
   After you start a job to publish a personalization rule, you can monitor the status to make sure the publish completes successfully.

Publishing considerations:

If you are publishing personalization rules in a clustered server configuration, to an IPv6 host, or using Web Content resource collection classes, there are unique considerations that you should be familiar with.

Clusters

Publishing to or from a clustered environment requires no special configuration. The specific cluster member that will perform the publishing task is chosen by the same rules that apply to incoming Web requests (because the publishing mechanism uses HTTP messages). At the end of a successful publishing job, Personalization flushes its caches for that workspace to ensure that any subsequent personalized content will be as current as possible.

When you first used the Personalization authoring portlets on a cluster to publish objects, the Publish Status dialog (accessed through **More Actions** > **Publish** > **View Status**) only shows information about the publish jobs initiated on that cluster member. To make all publishing jobs visible, set the **pzn.publishServlet.url** parameter described previously to be a specific cluster member. Set the URL to point to a single machine at the internal HTTP port: The default port number for HTTP is *10040*, and the default port number for HTTPS is *10035*.

For example, supposed the cluster head is visible at `http://intranet.yourco.com`, and the cluster members are accessible at `http://intranet01.yourco.com` and `http://intranet02.yourco.com`. If you set the publish servlet URL parameter to `http://intranet01.yourco.com:10040/wps/pznpublish/pznpublishservlet` you force all publishing requests to run on this single machine.

Note: Publish to a single node in the cluster as opposed to the cluster head, making sure you do not pass through a Web server.

IPv6 hosts

The server which initiates the publish command must have the IPv6 protocol stack installed and available. When publishing from the command line using pznload to an IPv6 host, you may need to set the system environment variable IBM_JAVA_OPTIONS to a value of *-Djava.net.preferIPv4Stack=false -Djava.net.preferIPv6Addresses=true* on the system where pznload is run.

Resource collection classes

You use resource collections in Personalization to access LDAP, IBM Content Manager, the Portal user object, or other custom sources of data.

The DB2 Content Manager run-time edition and the WebSphere Portal Express user resource collection classes are installed in the Personalization shared library. Therefore, you do not need to move these classes between systems because they are already installed with Personalization.

For LDAP resources, Rational Application Developer provides a wizard to generate classes which implement the resource collection interfaces.

To use the authoring portlet, all resource collection classes must be in the class path of the Personalization authoring portlet. The rule editor uses these classes to display the list of attributes belonging to the collection. If the resource collection classes are not found by the rule editor, you could see the following message in a JavaScript alert.

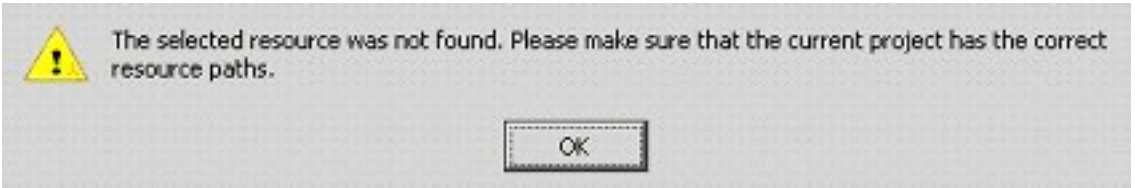


Figure 22. Message displayed when resource classes cannot be found

The resource collection classes must also exist on the class path of the application invoking the Personalization rules. The Personalization rules engine finds the

resource collection classes using the class path of the application which invokes the rules. If you use the Personalized List portlet to display rule results, this application is the Personalized List application `pznruleportlet.war` in the Personalization `Lists.ear`.

So, the classes should be accessible to both the rule editor and the personalized application. An application server shared library is the easiest way to accomplish this. You can configure the shared library using the WebSphere Integrated Solutions Console. For more information, see the sections on the shared library in the WebSphere Application Server Information Center.

You handle updates and additions to the resource collection classes just as you would handle updates to any application binary or JSP. These classes are not affected by Personalization publishing. The definition of the resource collection which Personalization uses to associate a resource collection with its classes is stored in the Content Manager repository. Initially represented by the .hrf file, this definition is published along with the rules and campaigns.

Publishing personalization rules:

After developing personalization rules, you publish the rules.

Procedure

1. To begin publishing personalization rules, you create an object in the authoring environment which describes the target endpoint. This endpoint definition is referred to as a publish server and is created and managed in a manner similar to creating and managing rules and campaigns.

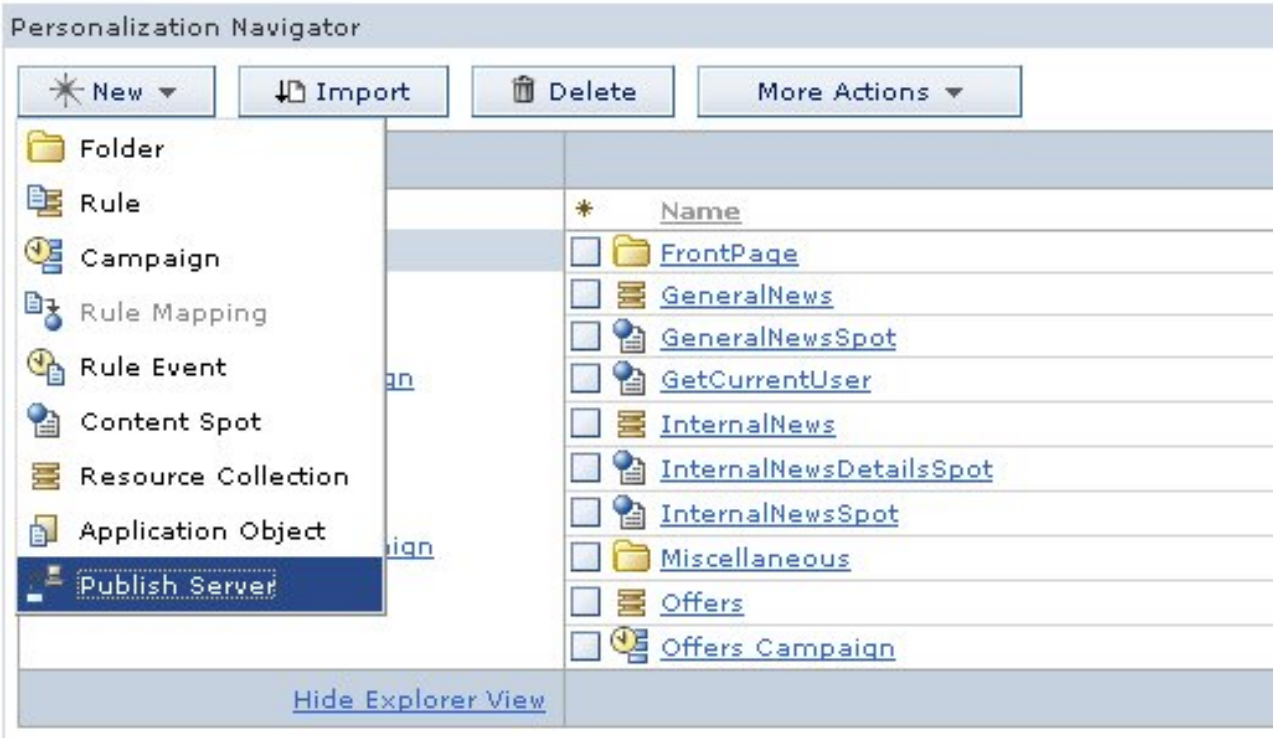


Figure 23. Creating a new publish server

The server requires one field, which is the URL associated with the publish servlet for that endpoint. The publish server may also define which workspace

will receive publishing data. Personalization operates in the default Content Manager run-time edition workspace after installation. If the target workspace field is empty, then the publish server uses the default workspace. (You need to set the workspace field if you are configuring scenario three described previously.)

The last option is whether or not to delete remote objects that have been deleted on the local system. The default is Smart Delete, which simply removes items that are no longer present. If you do not have delete permission on the remote server you could select the Leave deleted resources on server option.

2. After you create a publish server, you can publish either the entire workspace or a set of objects within it. You specify either of these options by selecting the **More Actions** > **Publish submenu**

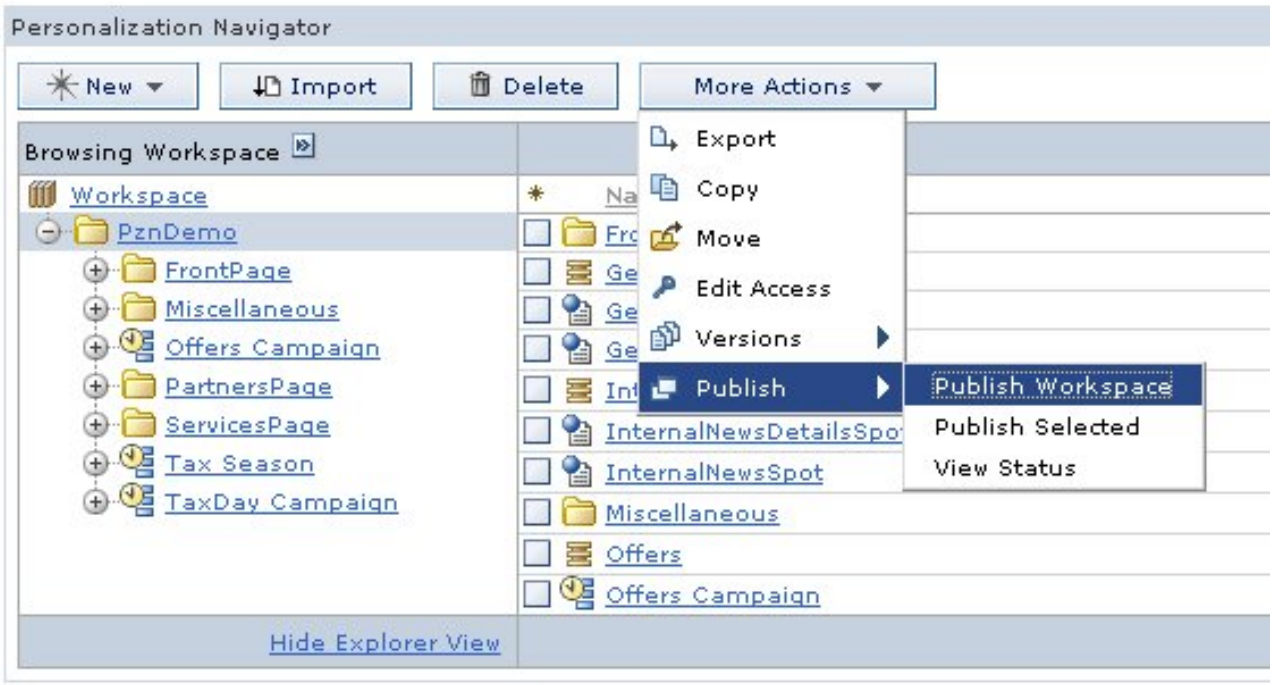


Figure 24. Start a publish job

The Publish page displays what will be published. This page requires the user to choose a destination publish server and any necessary authentication information. If the remote system is secured and is not a member of the current server's Single Sign-On domain you can enter a user name and password in the provided fields. The values for user and password are stored in the WebSphere Portal Express credential vault and are not accessible to any other user.

3. Finally, click **Publish** to launch the publish job.

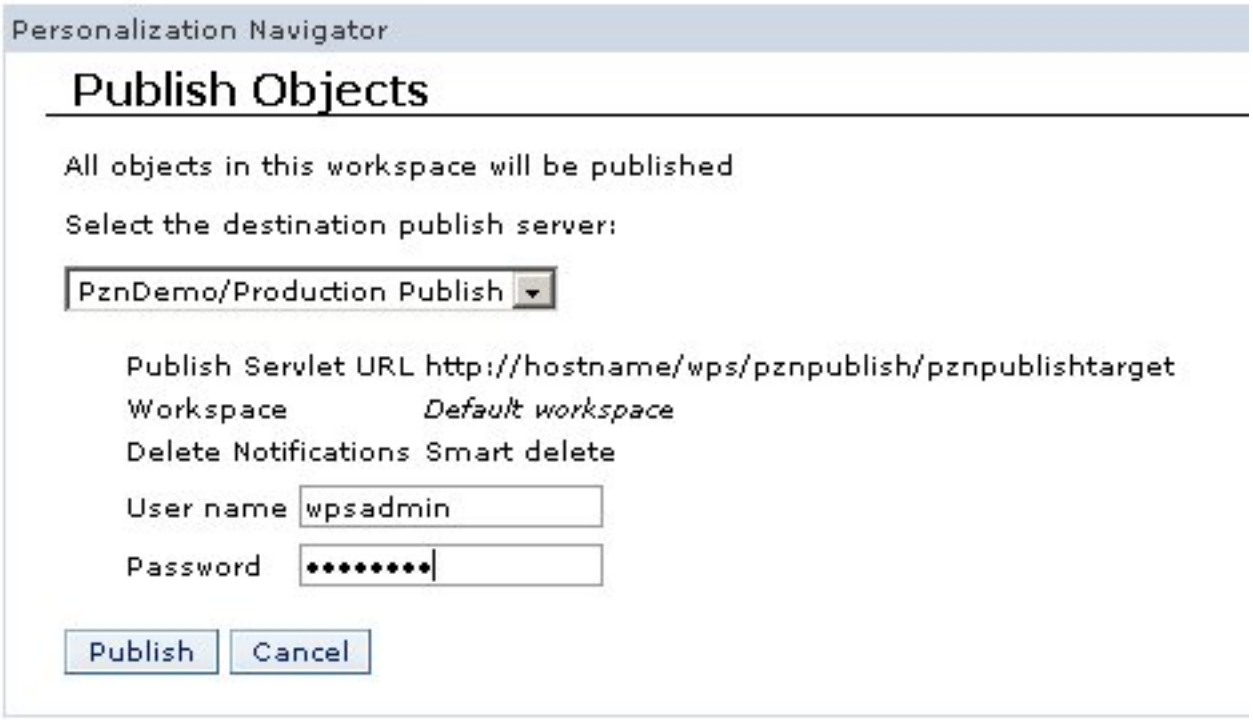


Figure 25. Publish page

If the local system is able to locate and authenticate with the remote publish server, you are returned to the main navigator view, and you see the Personalization message `EJPVP20001I`. Then, the publish job runs as a background process on the local server.

4. Click the **View the details of this job** link to open the publish status window to see information about the progress and success or failure of the publish job.

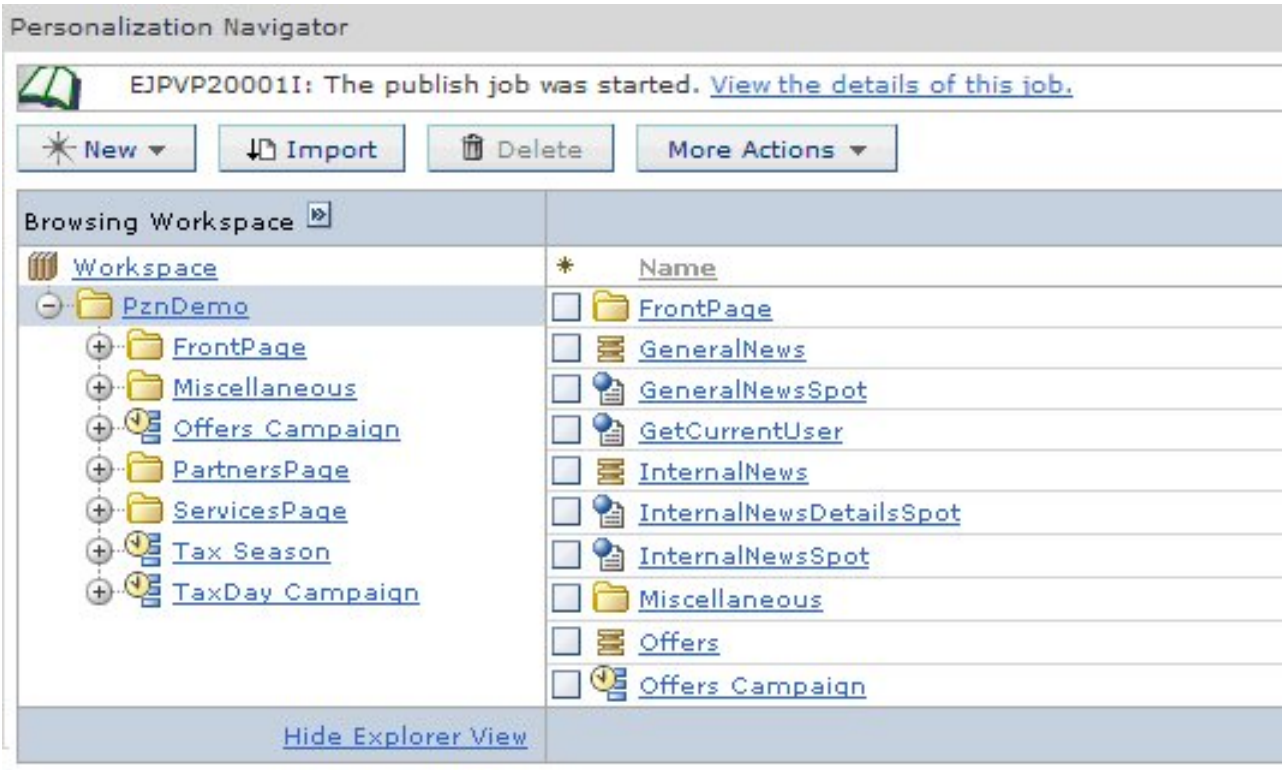


Figure 26.

Publishing and deleting personalization rules using a script:

You can use a WebSphere Portal Express Personalization provided script, pznload, to export, publish, and delete Personalization rules on local or remote servers. You can script the delivery of rules and campaigns from staging to production, or the offline publishing between disconnected systems (such as when production servers are secured behind a firewall). You can use this function to quickly revert production servers to an earlier state.

Before you begin

In order to run the pznload script, you must first run the `setupCmdLine` batch file to set up the WebSphere Application Server parameters. The `setupCmdLine.bat` file is in the `wp_profile\bin` directory.

About this task

Publishing through the command line is a two-step process. First, you export the personalization rules that you want to transfer from the authoring environment to a remote system. After you are done exporting and saving the required objects, you use the pznload script to send this data to the appropriate server.

Tip: For extra help on using the pznload script, run the `--help` command.

Procedure

1. You can export the personalization rules on the site or you can run the pznload command. Select one of the following methods to export personalization objects from the site:

- Click **More Actions** > **Export** in the Personalization Navigator. You are prompted for a location to save a nodes file. This file contains an XML representation of all the currently selected personalization objects. You can export entire folders.

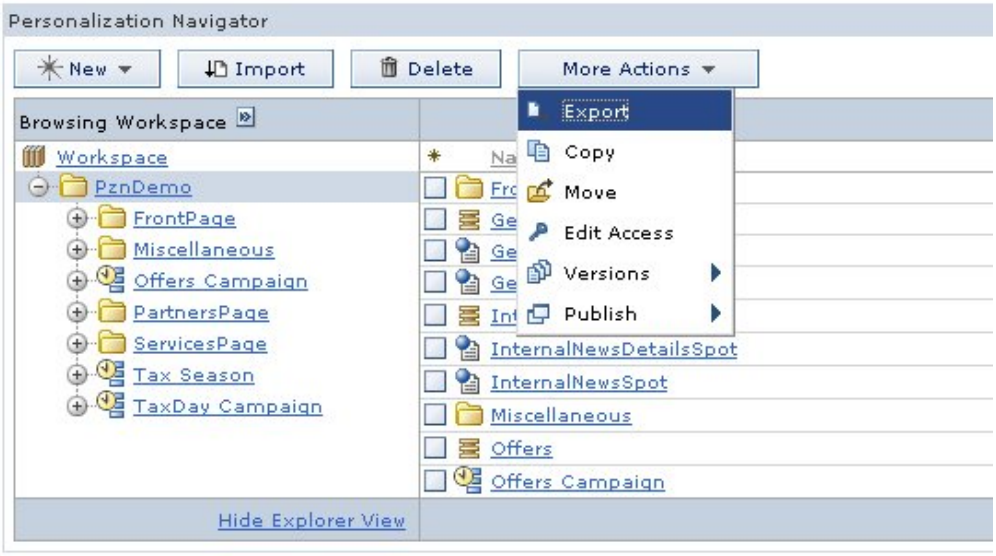


Figure 27. Exporting a folder to the file system

- Open a command prompt and run the following command, where `out` designates the location of the exported data on your local system and `targetpath` is the object (and child) that is exported:

  `pznload --export --out filename --serverurl` *`url`* `--targetpath` *`path`* `--targetworkspace` *`workspace`* `--username` *`username`* `--password` *`password`*

2. Choose the appropriate option to send this data to the appropriate server:

   The **`pznload`** script is in the *PortalServer_root*`/pzn/prereq.pzn/publish/` directory.

   Tip: This program accepts a number of command-line options and a set of nodes files to publish. Start `pznload` with the - help option to see a list of all options. The most important arguments are described here:

   serverurl
   The URL of the remote publish servlet. If you do not specify a value, the program attempts to connect to a WebSphere Portal Express server that is running on the local machine.

   targetworkspace
   The name of the workspace to publish to. The default workspace name on all IBM Content Manager run time edition installations is *ROOTWORKSPACE*.

   targetpath
   The location in the target workspace, which is the parent for the published nodes. The target path must exist before publishing. Example: If the export function was used on the folder `/Projects/HR` website, then the target path is specified as `/Projects`. So that the published resources are once again in `/Projects/HR` website.

username
A valid user on the target system with sufficient access rights.

password
The password for the user.

- Linux: `./pznload.sh --serverurl url --targetpath path --targetworkspace workspace --username username --password password`
- IBM i: `pznload.sh --serverurl url --targetpath path --targetworkspace workspace --username username --password password`
- Windows: `pznload.bat --serverurl url --targetpath path --targetworkspace workspace --username username --password password`

3. To delete objects, run the following command where `targetpath` is the object (and all associated child) that is deleted.

   `pznload --delete --username username --password password --targetpath path`

   Note: To force the deletion of a personalization rule without having to respond to a confirmation message, insert the `--force` flag after the delete command. For example, `pznload --delete --username username --password password --targetpath path --force`

Results

After a publish is started, you see status messages in the command console.

Publishing personalization rules over SSL:

WebSphere Portal Express Personalization uses the built-in SSL capabilities of WebSphere Application Server to provide secure publishing across unprotected networks. Your personalized portal can benefit from the full range of authentication repositories supported by WebSphere Application Server security.

About this task

In some environments even SSL publishing may not be secure enough. The pznload command-line program lets you fully control the transportation of the rules and campaigns during publish. You can encrypt the exported .nodes file and send it using email, or you can use another secure channel such as physical media transported between the staging and production servers.

Procedure

1. Enable SSL between the personalization servers. To enable Personalization publishing over SSL, see the Personalization Navigator's inline help: click the question mark, and scroll down to the end of the page to locate the link to the help topic on publishing.
2. Alter the publish servlet URL for secure publishing. If the remote server is not using the default HTTPS port of 443, modify the URL by adding a colon and the port number immediately after the host name.

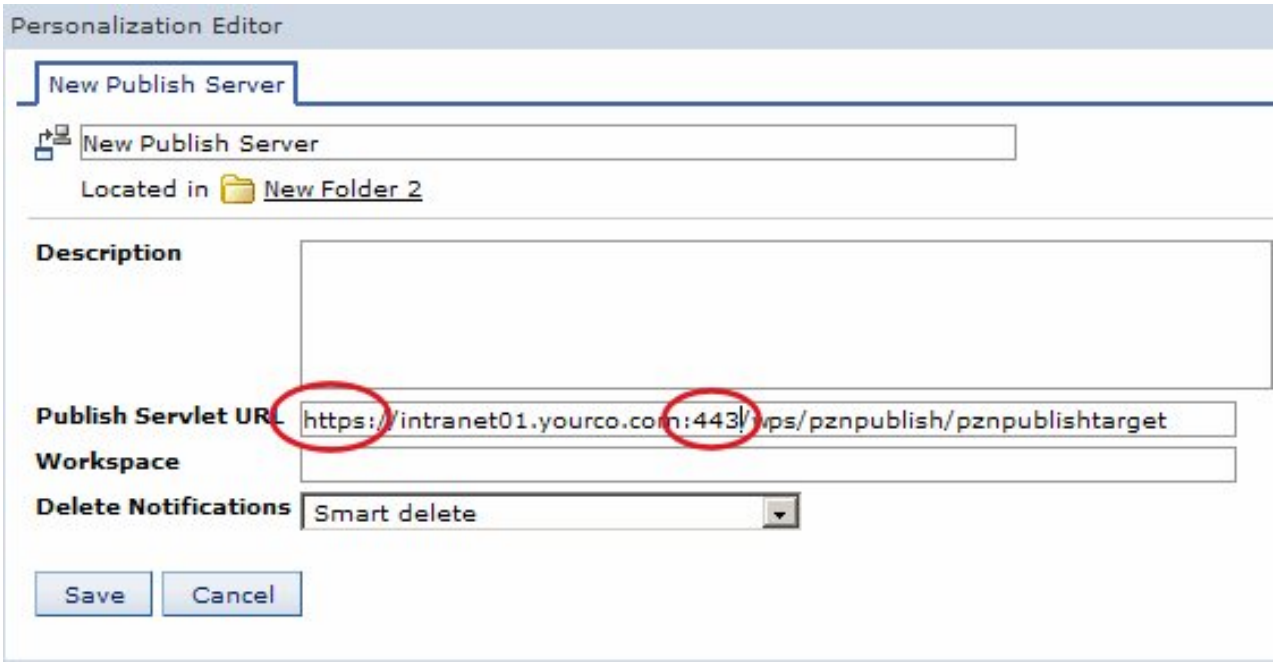


Figure 28. Altering the publish servlet URL for secure publishing by adding a colon and the port number after the host name for remote servers not using the default HTTPs port of 443.

3. Configure the personalization server from which you will be publishing to use the HTTPS protocol. To determine whether a particular URL is valid, point your browser to that location and enter your username and password for the system. If you see the message Publish servlet available and all SSL certificates have been properly imported, you should be able to publish. You can change this URL to redirect all publish jobs through a specific cluster member. If you encounter an error message that indicates the publish service was not available, the local publish servlet may not be configured correctly. To configure the local publish servlet URL:
   a. Click the **Administration menu** icon. Then, click **Portlet Management** > **Portlets**.
   b. Locate the Personalization Navigator portlet in the list.
   c. Click **Configure portlet** to configure the portlet.
   d. Add a new portlet parameter whose name is **pzn.publishServlet.url** and specify the appropriate value.

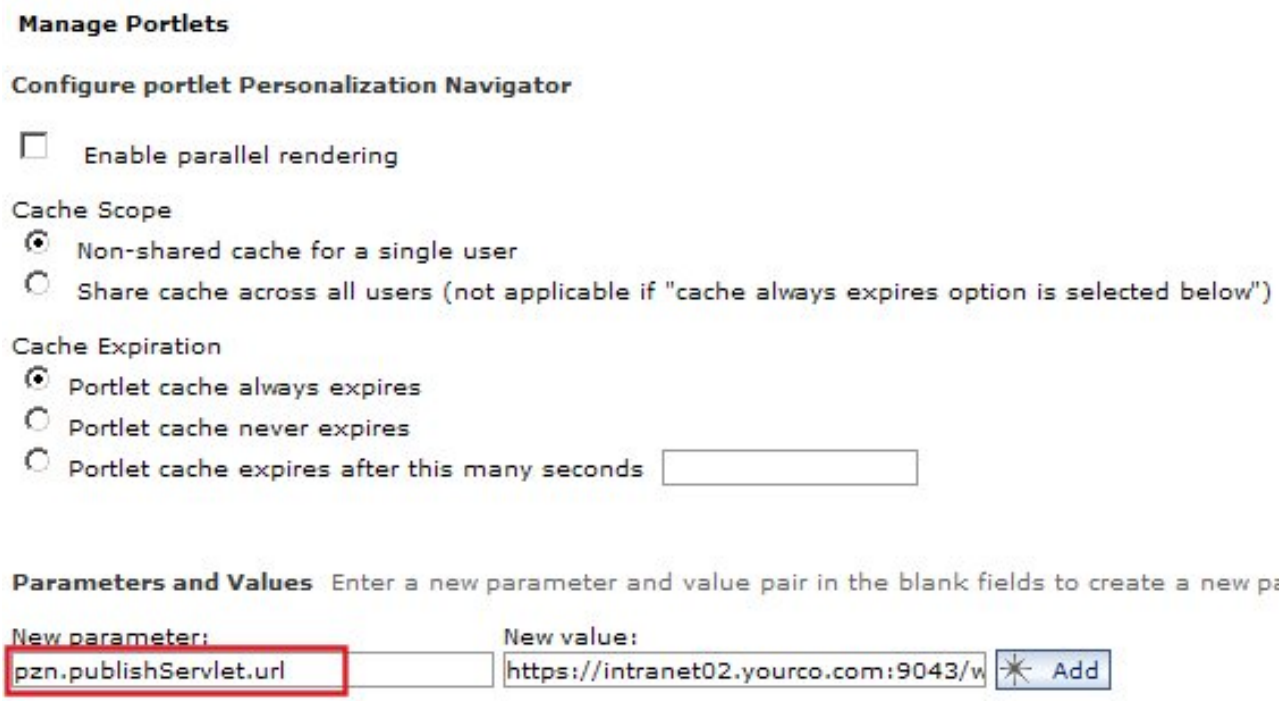


Figure 29. Configuring the local publish service

Results

If a Personalization server is configured to use a nonstandard HTTPS port or context root, or if you see messages such as `EJPVP20002E: The local publish service was not available` when publishing from the authoring environment, the local publish servlet URL might be incorrect.

Monitoring the status of publishing:

After you start a job to publish a personalization rule, you can monitor the status to make sure the publish completes successfully.

About this task

All publish jobs that are currently running or have been completed are displayed in the status view. If an error occurs, to get more information, turn on the Java Run time Environment tracing for WebSphere on the client system or examine the error and trace logs on the server system.

Procedure

1. To see the status of all current publish jobs, select **More Actions** > **Publish** > **View Status**.

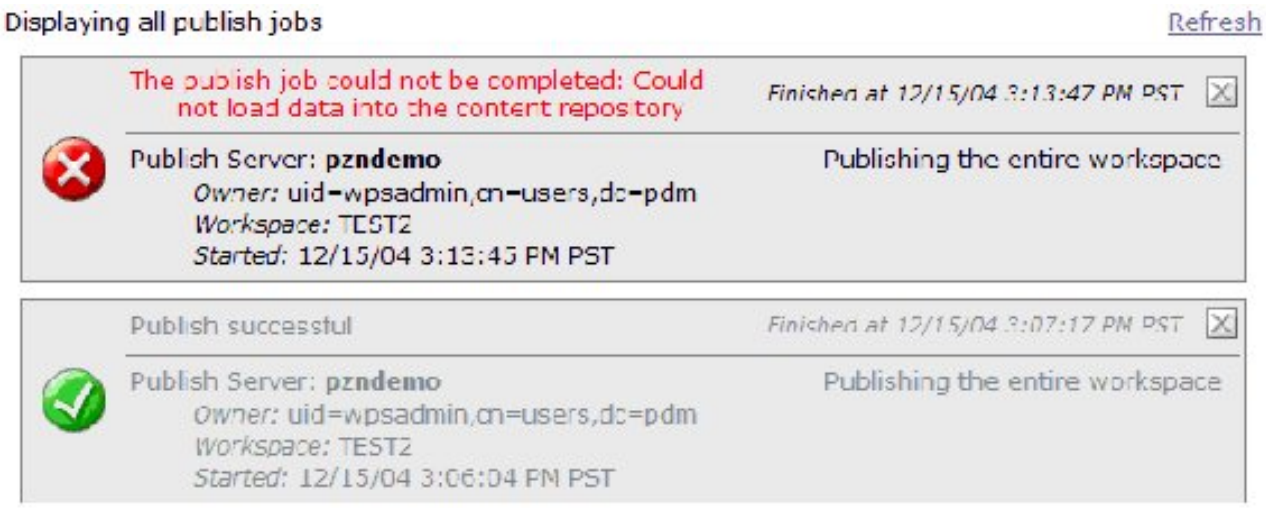


Figure 30. Publish status window

2. After a job has completed (successfully or otherwise) a close icon displays that you can click to remove the job from the list of monitored jobs. If you click this icon, you can no longer view the status of that job.

The Web Content resource collection

The Web Content resource collection is installed and configured out of the box. This predefined collection allows you to write rules that select lists of content from IBM Web Content Manager. Rules specify which Web content to show in a Portal Personalization component in Web Content Manager.

The Personalization component is similar to a menu component in that its content is decided by a rule. The Personalization component specifies how the content returned from the rule is presented and points to a rule to decide what content to display. That rule may make use of the Web Content resource collection to select a list of Web content.

Personalization rules that you create using the Personalization editor are not managed in Web Content Manager and so are not available for versioning or included in syndication. These rules must be published using pznload or by publishing with Personalization.

The Web Content resource collection allows rules based on the following attributes:

Table 354. Attributes used in Personalization rules for Web Content resource collections. The names and descriptions of Web Content attributes that you can use in Personalization rules

Attribute	Description
Author	The author of the content as set in Web Content Manager. This attribute value is stored as a Distinguished name; for example, `uid=wpsadmin,o=defaultWimFileBasedRealm`.
Authoring template	The authoring template used to create the Web content. The value may be selected using an authoring template picker.
Authoring Template Properties	All authoring template properties that are available to be used within rules.

Table 354. Attributes used in Personalization rules for Web Content resource collections (continued). The names and descriptions of Web Content attributes that you can use in Personalization rules

Attribute	Description
Category	The categories to which the piece of Web content belongs. If you are matching this attribute to a value from another resource collection or application object, the format for the value should be `/parentcategory/childcategory`.
Creation date	The date the piece of Web content was created.
Creator	The person who first created this piece of Web content. This attribute value is stored as a Distinguished name; for example, `uid=wpsadmin,o=defaultWimFileBasedRealm`.
Description	The description provided for the piece of Web content.
Expiration date	The date the content is set to expire in Web Content Manager.
Full text	Use this attribute to search the full text of a piece of Web content. It should be used sparingly. This attribute may not perform as quickly as other attributes when used in rules, especially when used in conjunction with other attributes.
Keywords	The keywords stored on the piece of Web content.
Last modified date	The date a modification last occurred on the piece of content.
Last modifier	The last person to modify the piece of content. This attribute value is stored as a Distinguished name; for example, `uid=wpsadmin,o=defaultWimFileBasedRealm`.
Name	The name of the piece of Web content as specified in Web Content Manager. This property uses case-insensitive matching. For example, a piece of content with a name of "SampleContent" matches "SampleContent," "samplecontent," "SAMPLECONTENT," and other variations.

Table 354. Attributes used in Personalization rules for Web Content resource collections (continued). The names and descriptions of Web Content attributes that you can use in Personalization rules

Attribute	Description
Position	The zero-based numeric position of the piece of content among its siblings in the nodes of the content hierarchy. Web Content resource collections use absolute positioning and relative positioning to support the ordering of content items. The algorithm used to generate position numbers results in fractional and negative values that control the display order of a set of child content items under a given area of a site. For more information, refer to the description of the position attribute in the *Command reference for the Portal Scripting Interface*.
Publish date	The publish date as specified in Web Content Manager.
Location	The document library or site area from which to select content. If specified as a string, it should be in the format `/Library/Site/SiteArea`. You can also specify `*/Site/SiteArea` to search by site area in all document libraries. Rules from previous versions with the Site Area attribute will automatically reference Location. If the site area was specified as a string, ensure the string starts with `/Library` or an asterisk (*).
Title	The display title as specified in Web Content Manager.
Unique Identifier	Use this attribute to select pieces of Web Content.

When a text, short text, numeric, or date component is added to an authoring template in Web Content Manager, that component will appear as an attribute on the resource collection under the Authoring Template Properties menu item. For example, if an authoring template is created for "Benefit Announcement" which includes an "enrollment begin" date component, you will have an "Enrollment begin" attribute on the Web Content resource collection. This new attribute would appear under a submenu for "Benefit Announcement". This will allow you to write rules based on custom metadata you attach to Web content. The performance of the standard metadata will do better compared to the performance of rules using attributes added to authoring templates. The use of keywords and categories should be considered since these are part of the standard metadata of Web content. If you are running a rule that uses the **`Name`** property as a sort by attribute, you might experience processing delays of up to 10 minutes. For best processing performance, sort by the **`Title`** property instead.

If too many authoring template properties have been designed, the Authoring Template Properties menu may become to large to be easily used. Once there are more than 15 authoring template properties, they are replaced with a chooser to

select properties. This threshold can be adjusted by changing the value of wcm.authoringTemplate.menu.threshold in *wp_profile_root*/PortalServer/config/config/services/PersonalizationService.properties.

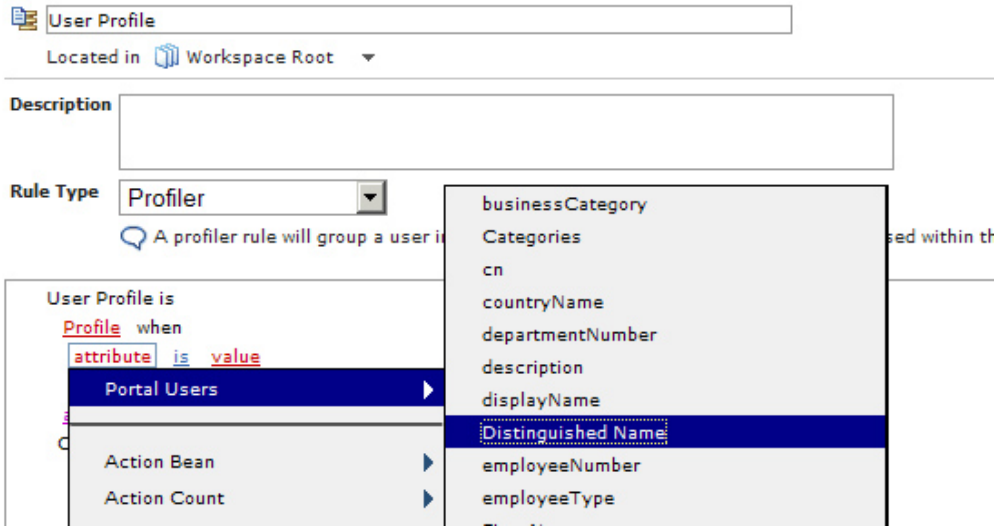
Personalization rules querying on short text run faster than those querying on text. Short text components can only store 254 characters. Text components can store an unlimited number of characters, but personalization rules will only see the first 254 characters. Anything after that is ignored when the rule runs.

Note: All attributes which store a reference to a person are stored as distinguished names. This format will match against the value of the Distinguish Name attribute on the Portal User resource collection. For instance, to select all documents authored by the user viewing the portlet, you would write, "Author is current Portal Users.Distinguished Name."

The Portal User resource collection

Portal Personalization comes with a Portal User resource collection. This collection uses public APIs provided by IBM WebSphere Portal Express to access user information.

The collection allows rules to be written based on all properties of the WebSphere Portal Express user.



You can also use the Portal User resource collection to profile based on the property extension database. Use the **Manage Properties** menu option in the rule editor to add or remove dynamic or Lookaside properties from the collection. You may add any properties to this collection, but for them to function at runtime some values must be stored in Virtual Member Manager for the properties being used and the user executing the rule. The Manage Properties screen can also be used if the server on which you author rules has a different Virtual Member Manager configuration than the one on which the rules are deployed.

The resource collection is not new to WebSphere Portal Express Version 8.5, but it now shows the list of Portal User attributes automatically. You no longer have to add each WebSphere Portal Express property you want to use as a dynamic property in the Personalization rule editor. You do not need to generate Java code to use this collection or configure a security ID translator as is often required for

Personalization User collections. You cannot currently write rules to select or recommend a list of users from this collection. This collection works with update rules as long as your repository allows writes. The collection works as if it is an LDAP collection that is automatically configured for the LDAP server.

You may continue to use your existing LDAP or custom user resource collections, and even use them in the same rules as the Portal User Collection. This is useful if you have multiple user repositories, you a have repository that is generally not used for Portal and only used for Personalization rules, or you requirements for attribute value translation ('CHI' should be interpreted as 'Chicago' for instance).

Using the Groups Attribute

The Groups attribute on the Portal User object exposes the distinguished names (dn) of the groups. An example of a distinguished name of a group is `cn=wpsadmins,o=defaultWimFileBasedRealm`, though the exact form will vary by your installation. Using distinguished name allows for more exact matching of groups, since it is possible for two groups to share a common name, such as `wpsadmins`. The `includes` operator may be used for inexact string matching, but will perform slightly slower. When possible, use the `is` operator and match to the distinguished name of the group.

Adding and extending user attributes

You can make attributes from your user registry available to the personalization portlet as required. However, in most cases the schema for your user registry does not match the default schema for Virtual Member Manager (VMM). For this reason, you must first extend the default VMM schema by adding attributes that you must then map from the VMM schema to your user registry.

Do the following to make user attributes available:

1. Extend the default VMM schema by adding attributes that you can map to your user registry. For instructions, see the following topic in the Installation section of this Information Center: *Adding attributes*. Refer to the topic that corresponds to your operating system and environment configuration.
2. Map the attributes you added to the VMM schema to the attributes in your user registry. For instructions, see the following topic in the Installation section of this Information Center: *Mapping attributes*. Refer to the topic that corresponds to your operating system and environment configuration.

Configuring which properties show up for the Portal User Collection

The Personalization rule editor discovers the list of properties to show in rules through a public API (`com.ibm.portal.um.PumaProfile`). The list exposed by this API is configurable in your Member Manager configuration.

You may have a long list of properties that are available on the portal user, but do not want all of them to show up in the Personalization rule editor. For this purpose, set the property **`wmm.property.hide`** in the file *wp_profile_root*`/PortalServer/config/config/services/PersonalizationService.properties` as illustrated in this example:

```
# Use this configuration property to control which WMM properties show
# in the Personalization rule editor. wmm.property.hide will only
# hide those properties which are introspected from the WMM configuration.
wmm.property.hide=mobile,pager,roomNumber,secretary,carLicense,telephoneNumber,
facsimileTelephoneNumber,seeAlso,userPassword,ibm-firstWorkDayOfWeek,
```

```
ibm-alternativeCalendar,ibm-preferredCalendar,ibm-firstDayOfWeek,ibm-primaryEmail,
ibm-otherEmail,ibm-generationQualifier,labeledURI,createTimestamp,
modifyTimestamp,ibm-middleName,ibm-timeZone,initials,jpegPhoto,WCM\:USERDATA,groups
```

The Portal User collection and the Personalization Server installed without Portal

The Portal User collection can only be used in rules running within a WebSphere Portal Express installation. The system will not prevent you from publishing rules using this collection to a Personalization Server installed outside of WebSphere Portal Express, but the rules will not function in this environment since Virtual Member Manager is not available. When WebSphere Portal Express is installed, the security context of the logged in user must have access (through Portal resource permissions) to the user information being accessed through the collection.

LikeMinds Recommendations

Personalization contains a dynamic recommendation system based on LikeMinds. LikeMinds is software that is used with your e-commerce applications. LikeMinds analyzes user interactions that occur on your Web site and generates real time predictions and recommendations to your Web site users.

Real time predictions are generated by three LikeMinds engines using recommendation rules within Personalization. These rules, called recommend content, base their predictions on transactions logged through Personalization's rating and action beans.

When a user visits your Web site, rating and action beans log captured transactional data. If your e-commerce Web site is set up so that users can rate content (or products), you use Rating beans to capture rating data. Similarly, if you use shopping cart technology, you use action logging beans to capture content affinity behavior to capture shopping activity. Both rating and action data is stored in your database. For example, the following types of transactions may be recorded:

- Products a user has purchased
- Items added or removed from a shopping basket
- A history of the user's navigation throughout the application
- Products that go best with a product that the user has already selected
- Any action or series of actions that are meaningful for a site

Using recommend content rules, LikeMinds surfaces results through a set of recommendation engines. These engines predict relevant content for users based on their past Web browsing habits.

Typically, after a user has rated a minimum number of items or completed a minimum number of transaction activities, that user is assigned a set of mentors. A mentor is a specially designated user who has visited the e-commerce application a number of times, and whose profile is similar to the user's. LikeMinds uses a technique called collaborative filtering to build a mentor's profile for each user to predict how much a user will like particular items and which items that user will enjoy, buy, or add to their shopping cart.

Predicting a matching product to go with a user's selected product, independent of actual user preferences, is accomplished by the discovery of probable pairs of product matches to be recommended. This concept is called *item affinity* and uses a family of algorithms different from collaborative filtering. While collaborative

filtering uses its algorithms to discern the highly variable affinities between individual Web-surfers, the item affinity approach looks at relationships that can exist between items.

You can use LikeMinds in a variety of situations, including:

- eRetailer promotion and personalization Web sites
- Financial portal content recommendation and personalization Web sites
- Help desk and/or on-line technical support content recommendation Web sites
- Gift recommendations for eRetailer
- Music, movie, book, or other product rating and recommendations
- Travel bureau trip planners

"LikeMinds Recommendation Engine architecture"
Get an overview of the functioning and architecture of the LikeMinds Recommendation Engine. In order to build your own recommendation application you need to customize the LikeMinds Recommendation Engine settings to work with your database and Web applications.

"How LikeMinds generates recommendations" on page 2316
Learn how LikeMinds generates recommendations when a user logs on and navigates through your web site.

"The LikeMinds Recommendation Engines" on page 2318
LikeMinds Recommendation Engines communicate with a relational database and generate recommendations. Learn about the three types of recommendation engines, Preference engine, Clickstream engine, and Item Affinity engine.

"The LikeMinds utilities" on page 2319
Get an overview of `sifter`, `buildstats`, `buildvisit`, and `accumulator`, the utilities that support running of background processes along with the LikeMinds server.

"Configuring LikeMinds" on page 2320
Use a suitable database modification tool or edit the `likemindsdb.properties` file to configure your LikeMinds server installation.

"MovieSite Sample" on page 2338
The MovieSite documentation focuses on six aspects of LikeMinds capabilities:

"Using the LikeMinds utilities" on page 2342
Learn about the four utilities LikeMinds uses to update the database, `buildvisit` (for the Preference Engine), `sifter`, `buildstats`, and `lpsIAA` (for the Item Affinity Engine's accumulator utility).

"Filtering LikeMinds recommendations" on page 2343
When LikeMinds makes recommendations, it can make the recommendations based on all items in your resource collection, or it can limit the predictions to only items that have certain characteristics.

Related information:

"LikeMinds Recommendation Engine architecture"
Get an overview of the functioning and architecture of the LikeMinds Recommendation Engine. In order to build your own recommendation application you need to customize the LikeMinds Recommendation Engine settings to work with your database and Web applications.

LikeMinds Recommendation Engine architecture:

Get an overview of the functioning and architecture of the LikeMinds Recommendation Engine. In order to build your own recommendation application

you need to customize the LikeMinds Recommendation Engine settings to work with your database and Web applications.

The LikeMinds Recommendation Engine captures data based on user actions. From this set of actions, LikeMinds bases its construction of mentor sets and subsequent predictions. User actions can include:

- A history of the user's navigation throughout the application
- Products that go best with a product that the user has already selected
- Products purchased
- Items added to or removed from a shopping basket

The current user preferences are collected by the engine in the so-called rating vector and used to identify those people most like the current user.

People with a behavior similar to the current user become mentors for that user. The LikeMinds Recommendation Engine assigns a numeric weight to each mentor based on the level of similarity of the rating vector to yours. The more the mentor's rating values resemble a user and the more products the mentor has rated, the greater the weight.

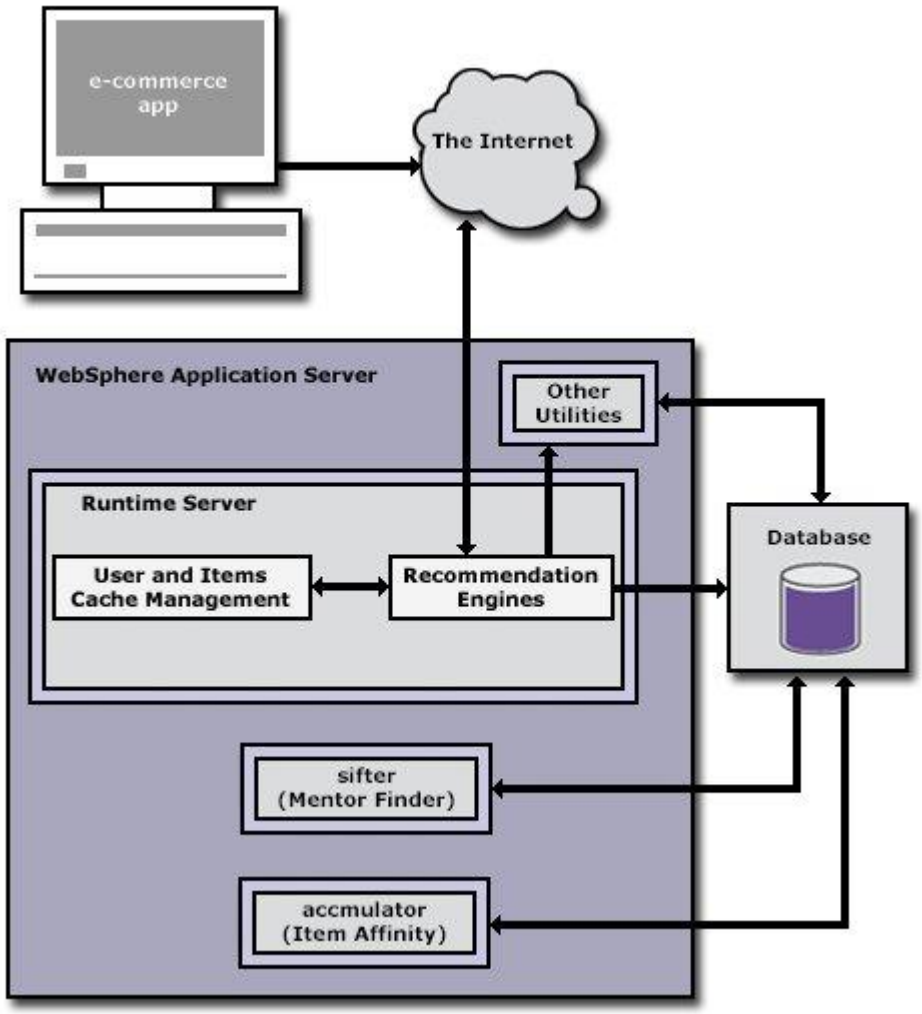
The LikeMinds Recommendation Engine assembles a set of recommendations by finding the products each mentor recommends and creating a prediction vector containing the predicted rating of each product. With each predicted rating, it also stores a numeric value representing the confidence for the rating.

The confidence values determine the quality of predictions. The LikeMinds Recommendation Engine assigns a confidence level to each recommendation based on how many users have rated the recommended item and how similar the ratings are to each other.

A user is assigned a set of mentors only after he has rated a minimum number of items or completed a minimum number of transaction activities.

If you want your application to predict a matching product to go with a user's selected product, you can configure probable pairs of product matches to be recommended. This concept is called item affinity.

Depending on the type of recommendation you want to extract, the specific engines must be configured, such as the Preference Engine or Item Affinity Engine. The following image shows the details of the overall LikeMinds Recommendation Engine architecture.



How LikeMinds generates recommendations:

Learn how LikeMinds generates recommendations when a user logs on and navigates through your web site.

When a user logs on and navigates through your Web site, LikeMinds follows these steps to generate recommendations for that user:

1. Personalization Rating beans and action logging beans create a record for new users in the Lps_User_Data table. The Lps_User_Data table stores the following types of information about the user: the user's resource ID, a user ID, the number of items the user has rated or selected, and so on.
2. The Personalization Rating beans and action logging beans log data for that user as that user navigates your Web site.

   The profile data is first stored in the server's cache, then the server writes all the new data to the database. The Lps_User_Rating table stores the user's explicit preferences; the Lps_User_Trx table stores the user's clickstream and purchase behavior. The Lps_User_Trx table also stores item affinity input data.
3. The application can then query LikeMinds for recommendations. Recommendation queries are transaction data-specific.

- Preference recommendations are surfaced by the recommend content rule. To receive Preference recommendations, your application must record users' explicit preferences (ratings) using the Rating bean.
- Clickstream recommendations are surfaced by the recommend content rule. To receive Clickstream recommendations, your application must record users' clickstream behaviors (that is, product detail views, shopping basket inserts, and so on) using the Action bean.
- Item Affinity recommendations are in the form: "`this mouth guard is a likely product to go along with the hockey puck the user has just added to his or her shopping cart`". To receive Item Affinity recommendations, your application must record users' likely product pair matches. In other words, your application must capture the current "content/product" context using the Action bean, in order to return those items most associated with that "content/product".

4. Depending on the engine you are using, the following step occurs next:
   - **Preference and Clickstream engines**: For a new user, if your application queries LikeMinds for recommendations before mentors have been assigned for that type of data (that is, Preference, Clickstream, or Item Affinity), the server will assign mentors from a cached pool of mentors.

     If the server is unable to, for lack of profile data, match cached mentors to this user, the server will provide an empty set of recommendations. An important distinction, profile data means the transaction data for the current user and not the attributes of that user.
   - **Item Affinity engine**: If your application is predicting item affinity product pair matches, it will collect data based on a set of definitions that you create, called an item affinity set. For input data, the item affinity set uses transactions from a specified input table.
5. Depending on the engine, the following step occurs next:
   - **Preference and Clickstream, engines**: Once a user's profile is stored in the database, the `sifter` utility can calculate mentors for that user. The `sifter` is a background utility which assigns a set of mentors to each user.
     - Mentor assignments are specific to each type of data.
     - Mentor assignments are stored in the mentor table associated with this type of data.
   - **Item Affinity engine**: The `accumulator` generates item affinity product pair matches by analyzing data in the item affinity set (same as transaction table) and recording its findings to an output table.
6. Depending on the engine, the next step is as follows:
   - **Preference and Clickstream engines**: As new transaction data is recorded for a user, the user is prioritized for reprocessing by the `sifter` to calculate new mentor assignments. Users are prioritized by a calculated 'sift priority', reflecting the percentage of new or changed profile data for that visitor.
   - **Item Affinity engine**: As new product selection behaviors are recorded in the transaction input table specified in the item affinity set definition, the `accumulator` uses this data to calculate new item affinity recommendations.
7. When your application runs LikeMinds rules, the following occurs, depending on the engine:
   - **Preference and Clickstream engines**: LikeMinds looks up that user's mentors, and calculates recommendations.
   - **Item Affinity engine**: LikeMinds calculates the most likely item-content pairs based on accumulated item transaction history.

The LikeMinds Recommendation Engines:

LikeMinds Recommendation Engines communicate with a relational database and generate recommendations. Learn about the three types of recommendation engines, Preference engine, Clickstream engine, and Item Affinity engine.

Following is a description of the three types of recommendation engines:

- **Preference engine**: This engine generates recommendations using patented collaborative filtering algorithms based on users' item ratings.
- **Clickstream engine**: This engine, which also accesses transaction information, generates recommendations based on users' actions as they navigate a Web site; that is, the history of user "clicks" during website visits, and the items that users view, click, and add to their shopping carts.
- **Item Affinity engine**: This engine generates recommendations based on the history of the user's site browsing activity. It matches a currently selected product with a second product that the user would most likely want to purchase along with the first product. For example, if a user is purchasing groceries and adds French onion soup to the shopping cart, the Item Affinity engine could recommend Gruyere cheese to go with it.

  "Preference Engine"
  The Preference Engine uses explicitly stated user preferences to make highly accurate recommendations for products and content that your website visitors like.

  "Clickstream Engine" on page 2319
  Based on navigational data gathered as customers browse your Web site, the Clickstream Engine tracks clickstream (or rating) behavior and generates recommendations based on mentors who exhibit similar content/product affinities.

  "Item Affinity Engine" on page 2319
  The Item Affinity Engine generates recommendations based on any transactional history available, such as shopping cart activity, external legacy transactions, and web transaction completely unrelated to shopping cart activity (page views, product inquiries, searches, and so on).

Preference Engine:

The Preference Engine uses explicitly stated user preferences to make highly accurate recommendations for products and content that your website visitors like.

The Preference Engine enables customers get exposure to items they might otherwise miss. For example, users shopping for gifts can get recommendations based on the gift recipient's shopping preferences.

The shipped with the LikeMinds Recommendation Engine presents a typical implementation of the Preference Engine:

- A user can rate a movie and LikeMinds returns a predicted rating for that movie.

  For example, a user can assign the movie *Fantasia* with a rating such as "I loved it". Internally the rating corresponds to a numerical value.
- A user can ask for a recommendation about the best bet, which is provided by a LikeMinds rule. This corresponds to asking which movie a user would like the most among all available movies.

Clickstream Engine:

Based on navigational data gathered as customers browse your Web site, the Clickstream Engine tracks clickstream (or rating) behavior and generates recommendations based on mentors who exhibit similar content/product affinities.

The Clickstream Engine tracks the pages that users have looked at. After analyzing all users' traffic patterns, it then makes content recommendations for each specific user, using data from relevant subsets of the user base.

Item Affinity Engine:

The Item Affinity Engine generates recommendations based on any transactional history available, such as shopping cart activity, external legacy transactions, and web transaction completely unrelated to shopping cart activity (page views, product inquiries, searches, and so on).

An Item Affinity Engine can, for example, predict that a user who purchases a digital camera would likely want to purchase compact flash cards or a USB card reader. The Item Affinity Engine also spots the less obvious connections—that users who purchase beer are likely to purchase diapers at the same time, for example.

The Item Affinity engine lets you track more than mere purchases measured at check-out time. It can identify items the user only considered for purchase. For example, it can know when a user only considered purchasing rye bread rather than actually purchasing the rye bread. (This is measured by a shopping cart add, followed by a shopping cart drop, of the rye bread.) In this case, the grocer could not possibly know that the user considered purchasing rye bread during the shopping session, since the rye bread was not in the shopping cart at check-out time. Even though a considered purchase does not necessarily imply the same level of item affinity as a completed purchase, it does convey item affinity information.

Unlike the other engines, Item Affinity Engine recommendations are based on Market Basket Analysis statistics, not collaborative filtering. Market Basket Analysis enables content affinity predictions even when cold-start situations obscure the relevance of collaborative filtering. The Item Affinity Engine can be used to provide improved automated recommendations, such as cross-sells, even for first time visitors to the Web site.

The LikeMinds utilities:

Get an overview of `sifter`, `buildstats`, `buildvisit`, and `accumulator`, the utilities that support running of background processes along with the LikeMinds server.

The following utilities support the background processes operating on the database when the LikeMinds server is running:

- `sifter`: This utility runs continuously to identify mentors for new users and recomputes the best set of mentors for existing users. The mentor set for a user may change as the sifter gathers more information about the user or the mentors. The `sifter` identifies mentors for the Preference, and Clickstream engines.
- `buildstats`: This utility runs once a day to update statistics for each item, such as the number of ratings or transactions, the average rating, the standard

deviation in the average rating, and the default recommendation information. The Clickstream and Preference engines use `buildstat`. The Item Affinity engine does not.

- `buildvisit`: This utility, which the Preference engine uses, runs daily to construct lists of items to be presented to users for rating. If your applications do not use the Preference engine, `buildvisit` is not necessary.
- `accumulator`: For the Item Affinity engine, the `accumulator` (listed as `lpsIAA` in the `util` directory) accumulates the number of times every possible item-to-item combination occurs and writes its findings to an output table specified by the item affinity set. An item affinity set defines the type of data required to build an item-to-item combination.

Configuring LikeMinds:

Use a suitable database modification tool or edit the `likemindsdb.properties` file to configure your LikeMinds server installation.

Set the following general **`lps_cfg`** parameter information for the LikeMinds server installation:

- Basic server information
- Scheduling LikeMinds events
- Server load management
- Cache behavior
- Recommendation behavior

All these parameters are set in the `likemindsdb.properties` file. This file is in the directory *`wp_profile_root`*`/pzn/config/runtime/likemindsdb.properties`. The file is in ASCII. To edit it, use an ASCII editor.

LikeMinds stores its configuration information in the lps_cfg table of its database, which is initialized with the data in the `likemindsdb.properties` file. To update this configuration, you can update this file and reload the configuration data, or you can use any database modification tool, to modify the LikeMinds configuration parameters. To update configuration values, complete these steps:

1. Stop the WebSphere_Portal server.
2. Edit the `likemindsdb.properties` file.
3. Choose the appropriate task to update the configuration:
   - Linux : `./ConfigEngine.sh likeminds-load-config -DWasPassword=`*`password`*
   - IBM i: `ConfigEngine.sh likeminds-load-config -DWasPassword=`*`password`*
   - Windows: `ConfigEngine.bat likeminds-load-config -DWasPassword=`*`password`*
   - Use the DB modification tool to update the configuration directly.
4. Start the WebSphere_Portal server.

"Estimating database size" on page 2321
The size of your database depends on your application, as well as the number of users and items. View some general guidelines for estimating the size of your database, but your results may vary.

"Database performance" on page 2322
View some guidelines for performance optimization in your LikeMinds database.

"Scheduling LikeMinds Events" on page 2322
Use the `lps.schedule` setting to schedule events to be fired at specific dates and times.

"Configuring the LikeMinds engines" on page 2324

"Specifying recommendation behavior" on page 2336
You can configure several parameters that affect the way the LikeMinds server generates recommendations.

Estimating database size:

The size of your database depends on your application, as well as the number of users and items. View some general guidelines for estimating the size of your database, but your results may vary.

A database containing the seed data supplied with the Movie Site application may use just 250 MB, while the LikeMinds tables for a large site with millions of users may take up 10 GB. Following are some general guidelines:

The tables that contribute the most to the size of the database are as follows:

- **Lps_User_Rating**: This table normally dominates your space considerations. Users typically average 50 to 100 ratings. The seed users supplied with Movie Site average about 500 ratings.
- **Lps_User_Trx**: This table can grow very large, depending on the number of item affinity, clickstream, or purchase activities recorded from your applications.
- **Lps_MBA_Scored**: This table can grow quite large, depending on the number of products your site sells and the number of relationships you want to configure for each product. For example, if you have 1000 products listed in your Lps_Item_Data table, and you want to store 10 relationships for each product, an Lps_MBA_Scored table can grow to 10,000 rows.
- **Lps_User_Mentor**: The size of this table depends on the number of users and the number of mentors associated with each user (50 by default).
- **Lps_User_Data**: This table may contribute a large portion of the database size if you have large numbers of users who each have few ratings. This table is heavily indexed, which can affect performance.
- **Lps_Item_Data**: This table is normally fairly small, but may be significant if you store large amounts of data about each item.

The remaining tables are typically less than 100 KB each.

The following table gives typical numbers of rows, row sizes, and index sizes for a "typical" Microsoft SQL Server database with 5000 items and 100,000 users. The row sizes include only the fields required by LikeMinds, and they account for typical null fields and clustered index overhead. Sizes will vary for other database systems, especially for indexes.

Table 355. Example Table Sizes for a Typical Database

Table	Rows in Typical Site	Row Size (Bytes)	Total Size	Index Size (Bytes Per Row)	Total Size
Lps_User_Rating	8,000,000	25	200 MB	about 20	160 MB
Lps_User_Trx	8,000,000	32	256 MB	about 20	160 MB
Lps_User_Mentor	5,000,000	25	125 MB	about 20	100 MB

Table 355. Example Table Sizes for a Typical Database (continued)

Table	Rows in Typical Site	Row Size (Bytes)	Total Size	Index Size (Bytes Per Row)	Total Size
Lps_User_Data	100,000	typical: 100 maximum: 400	100-400MB	about 100	10 MB
Lps_Item_Data	5000	136 (for required fields only)	68 MB	4	2 MB
Lps_MBA_Scored	10,000	32	32 MB	about 20	1MB
Lps_Genre_Data	10-1000	116	1160 KB-116 MB		N/A
Lps_Item_Genre	5000-20,000	12	60-240MB		N/A
Lps_User_Selector	25,000-100,000	12	30-120MB	4	100-400 MB

In your size estimate, remember to include space for transaction logging and rollback areas. Because the LikeMinds server commits frequently, the rollback area need not be especially large relative to the database. Allow space equal to the size of the database for transaction logs, since the LikeMinds server performs frequent updates.

Database performance:

View some guidelines for performance optimization in your LikeMinds database.

How you configure your database during installation has a significant impact on performance. Use the following guidelines to achieve the maximum performance in your LikeMinds database:

- **Hardware setup**: Ideally, you should have one machine (with two CPUs), preferably the fastest, dedicated to the database, and the WebSphere Portal software on a separate machine. See the system requirements in the WebSphere Portal Information Center for more information.
- **load distribution**: You can install the LikeMinds utilities onto separate machines to distribute load. Refer to the documentation for information about installing utilities to multiple machines.
- **sifter configuration**: You can configure the `sifter` to accommodate heavy loads or busy sites. Be aware that the `sifter` has heavy memory usage.
- **accumulator configuration:** As with the `sifter` , the `accumulator` has heavy memory usage, so you should schedule it to run during off-peak hours.

Scheduling LikeMinds Events:

Use the `lps.schedule` setting to schedule events to be fired at specific dates and times.

The syntax for using the `lps.schedule` setting is as follows:

```
lps.schedule.<event name> = <schedule><event type><event args>
```

where:

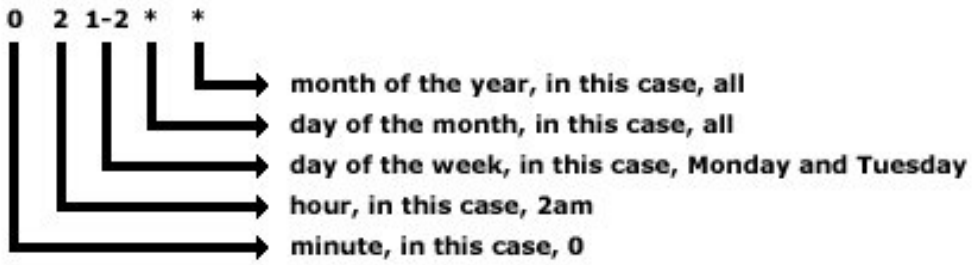
`event name`: Refers to a unique name for the event.

`schedule`: Refers to the scheduling time, in five fields separated by spaces or tabs, and are integer patterns. These fields are:

Table 356. Schedule Time Values

Field	Values
minute	0–59
hour	0–23
day of the week	0-6, with 0 being Sunday
day of the month	1–31
month of the year	1–12

To specify all values of a particular field (for example, to schedule events every day of the week), use an asterisk (*), in the order listed in this table. For example:



When you do specify actual values, you can enter either the value itself (such as 1 to schedule an event on Monday) or a range of values (for example, 1-2 to schedule events for Monday and Tuesday). When you want to specify days, you need to include both the day or the week field and the day of the month field. For examples of using these values, see

`event type` and `event args`: Refer to any of the following events and their arguments:

Table 357. Event Types and Arguments

Event type	Argument	Description
`lazyDBWrite`	N/A	Specifies when all the changed information since the last read is written to the database. The default setting is: `12,24,36,48,00 **** lazyDBWrite`
`purgeUserCache`	`timeval`	Specifies the time in seconds after permanent users have not been used before purging them from cache. If timeval is not specified, the setting from `db.tune.user_cache_age_time` is used.
`purgeItemCache`	`timeval`	Specifies the time in seconds after items have not been used before purging them from cache. If `timeval` is not specified, the setting from `db.tune.item_cache_age_time` is used.

Table 357. Event Types and Arguments (continued)

Event type	Argument	Description
syncCache	user	Refreshes all permanent user objects in cache.
	item	Refreshes all item objects in cache.
	engine	Refreshes all engine cache.
runBuildstats	verbose	Runs the buildstats utility. If verbose is specified, additional information is printed to the trace log.
runBuildvisit	verbose	Runs the buildvisit utility. If verbose is specified, additional information is printed to the trace log.
runAccumulator	verbose	Runs the accumulator utility. If verbose is specified, additional information is printed to the trace log.

Configuring the LikeMinds engines:
You can configure your LikeMinds recommendation engines to control aspects such as predictability, number of mentors used, which rating, transaction, or mentor set to use, and so on, for the recommendations returned.

The Preference and Clickstream engines rely on the `sifter` utility to assign mentors to users. The Item Affinity Engine uses the `accumulator` utility to collect item affinity data.

"Configuring the Preference Engine"
"Configuring LikeMinds utilities" on page 2327
"Configuring the Clickstream Engine" on page 2333
"Configuring the Item Affinity Engine" on page 2335

Configuring the Preference Engine:
The Preference Engine generates recommendations based on users' ratings of items. You can configure the following settings for the Preference Engine.

"Number of mentors to use"
"Mentors to look for in cache"
"Use of "average user" to improve recommendation confidence" on page 2325
"Archetypes" on page 2325
"Guidelines for configurable recommendation dynamics" on page 2325
"Setting the number of archetypes in cache" on page 2325
"Enabling or disabling the use of archetypes" on page 2325
"User predictability" on page 2325

Number of mentors to use:
Set `<eng_instance_name>.db.engine.tune.num_mentors` to the maximum number of mentors to assign to a given user. The default is 50. For example:

```
movie_pref.db.engine.tune.num_mentors = 60
```

If you do not set this parameter, then the Preference Engine uses the value set for `<mentor_set>.max_mentors = <number>`.

Mentors to look for in cache:

If a user requests recommendations before the sifter has found mentors for that user, the Preference Engine checks the cache for mentors for the user. Set `<eng_instance_name>.db.engine.tune.max_cached_mentors` to the number of potential mentors which it should consider for a given user. The default is 500. For example:

```
movie_pref.db.engine.tune.max_cached_mentors = 600
```

Use of "average user" to improve recommendation confidence:
The Preference Engine can use an "average user," whose ratings are the average of all users' ratings. The `buildstats` utility computes the average ratings. Configuring an average user improves the confidence level of recommendations for that user. To do so, set *`eng_instance_name`*`.db.engine.tune.disable.avg_user` to `false`.

However, remember that this feature can be costly in system resources. To disable it, set it to `true`. (The default setting is `true`.)

Archetypes:
To generate recommendations for items that no mentors have rated, the Preference Engine uses synthetic users called "archetypes." These users possess a particular characteristic very strongly. For example, an archetype who only likes action movies might rate all action movies very highly. Recommendations from archetypes are added to recommendations produced by mentors.

Guidelines for configurable recommendation dynamics:
You can use the following configuration parameters to control, or at least balance, effects of external factors.

- The number of mentors to use in making recommendations, using the following configuration parameters:
  - `<mentor_set_name>.min_mentor_ratings`
  - `<mentor_set_name>.max_mentor_ratings`
  - `<mentor_set_name>.min_mentor_transactions`
  - `<mentor_set_name>.max_mentor_transactions`
- Balancing the mentor selection process with the following configuration parameters:
  - `<mentor_set_name>.mentor.pool.size`
  - `<mentor_set_name>.max_mentors`

Setting the number of archetypes in cache:
The list of archetypes is kept in the cache for a specified number of uses before it is reloaded. To change the number of users, set `<eng_instance_name>.db.engine.tune.max_archetype_list_use` . The default is 100000.

For example:

```
movie_pref.db.engine.tune.max_archetype_list_use = 150000
```

Enabling or disabling the use of archetypes:
To turn on or off the use of archetypes, set `<eng_instance_name>.db.engine.tune.consult.archetype_for_list` to `true` or `false` . The default is `false` . For example:

```
movie_pref.db.engine.tune.consult.archetype_for_list = true
```

User predictability:
This section contains the following information regarding user predictability:

Maximum number of mentors assigned to each user:
Use the following setting to specify the number of mentors to be assigned to each user. The number of actual mentors can be less than the maximum setting but never greater than the value specified. For example:

```
<mentor_set_name>.max_mentors = 50
```

Note: Use a value between 50-100. See "Maximum ratings a user needs before becoming a mentor" for guidelines on setting `<mentor_set_name>.max_mentors`

Maximum ratings a user needs before becoming a mentor:
The following setting specifies the maximum number of ratings a user can have to become a mentor. The `sifter` uses this setting if the Preference Engine is using that mentor set. You can specify the value as one of the following:

- A percentage of the total number of items in the database. For example:

  ```
  <mentor_set_name>.max_mentor_ratings = 75%
  ```

- The maximum number of ratings (that is, without a percentage). For example:

  ```
  <mentor_set_name>.max_mentor_ratings = 18
  ```

Use the following guidelines for setting `<mentor_set_name>.max_mentor_ratings:`

Table 358. Guidelines for Setting the Maximum Ratings

Total Number of Items in Database	Suggested Setting
< 1000	<= 100
1000–5000	<= 500
5000–10,000	<= 1500
10,000+	<= 1500

Note: For more information on how the `sifter` makes use of `<mentor_set_name>.max_mentor_ratings`, see "Maximum number of mentors assigned to each user."

Maximum transactions a user needs before becoming a mentor:
The following setting specifies the maximum number of transactions a user can have before becoming a mentor. The `sifter` uses this setting if the Purchase or Clickstream Engine is using that mentor set. You can specify the value as one of the following:

- A percentage of the total number of items in the database. For example:

  ```
  <mentor_set_name>.max_mentor_xactions = 75%
  ```

- The maximum number of transactions (that is, without a percentage). For example:

  ```
  <mentor_set_name>.max_mentor_xactions = 18
  ```

Use the following guidelines for setting
<mentor_set_name>.max_mentor_transactions

Table 359. Guidelines for Setting the Maximum Transactions

Total Number of Items in Database	Suggested Setting
< 1000	<= 100
1000–5000	<=500
5000–10,000	<= 1500
10,000+	<= 1500

Minimum number of ratings for user recommendations:
Set <eng_instance_name>.engine.titan.predictable.min_ratings_cutoff to the minimum number of ratings a user must make before the Preference Engine will make recommendations for that user. The default is 2. For example:

```
movie_pref.engine.titan.predictable.min_ratings_cutoff = 3
```

Recomputing Preference Engine predictions:
The <eng_instance_name>.engine.titan.recomputation_bound configuration parameter specifies the percentage change allowed in a user's ratings before the LikeMinds server recomputes the user's predictions. For example:

```
music_pref.engine.titan.recomputation_bound = 10.0
```

Ordinarily, the LikeMinds server generates predictions based on a user's mentors, which the sifter computes and makes available to the database. When a user has no mentors, perhaps because he or she has just arrived at the site, or when the user's ratings or transactions have changed beyond the percentage specified here, the LikeMinds server selects mentors from a reduced set of candidates and recomputes the user's predictions.

Use this setting with caution, as selecting mentors is a relatively expensive operation: a low percentage setting can lead to excessive CPU load with little or no gain in prediction quality. A higher setting will improve performance, but predictions may be less accurate.

Default Preference Engine recommendations:
The Preference Engine reads the value from the score field in the Lps_Item_Data table to determine which items are popular. Hence, a higher score means that the item is more popular. When buildstats runs, it updates the score field to the average rating for that item; it generates this value based on the rating set defined in the db.applic.rating.source parameter. To get reliable default recommendations, run the buildstats utility on a regular basis. By default, the installer sets buildstats to run once a day. Some applications use their own business logic to assign the score to items. If you want to override the score field value to use your application's scoring instead, run buildstats using the -noscore argument.

Configuring LikeMinds utilities: You can configure the following settings for the buildstats and buildvisit background utilities:

- Ratability parameters
- Repeated items in the visit list

You can use the `buildstats` utility for all of the Recommendation engines except for the Item Affinity Engine. The `buildvisit` utility is only used for the Preference Engine.

"Configuring the sifter for mentor selection"
"Ratability parameters" on page 2332
"Repeated items in visit list" on page 2333

Configuring the sifter for mentor selection:
The sifter finds mentors for users, using the information from the rating or transaction data for the LikeMinds server engines. The sifter is used by all of the LikeMinds server engines except for the Item Affinity Engine, which uses the `accumulator` utility Figure 31 illustrates how the sifter works:

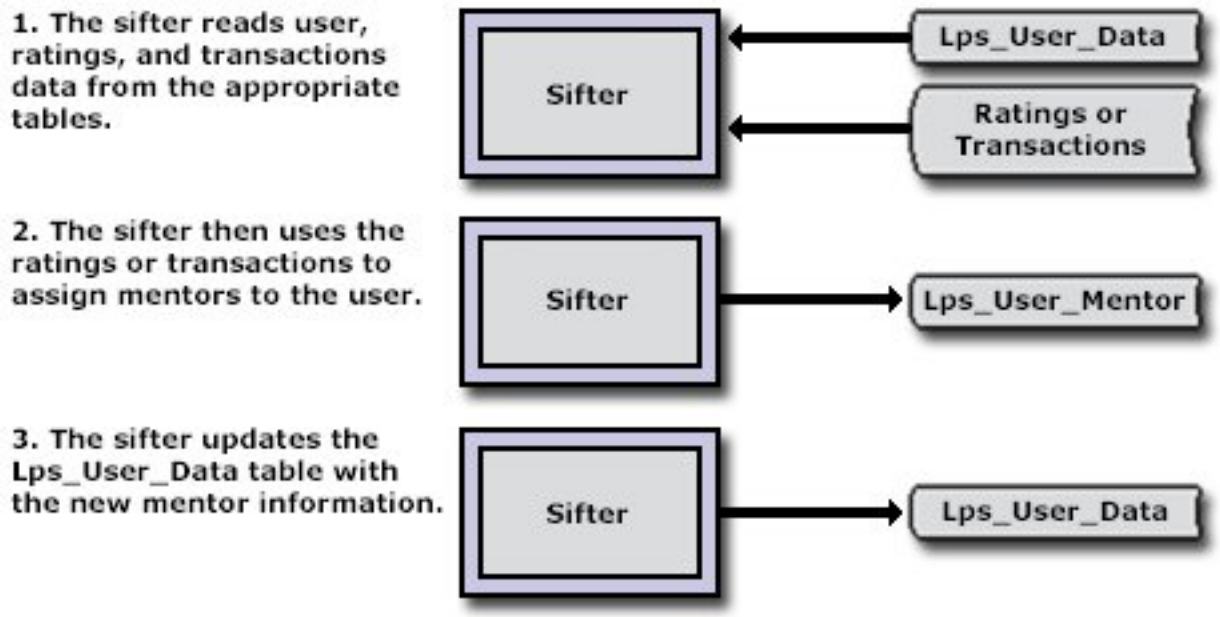


Figure 31. How the Sifter Works

"Sifter-specific mentor set configuration"
"How the mentor selection process works" on page 2329
"Sifter performance considerations" on page 2330
"Recomputing (rebuilding) the mentor pool" on page 2331
"Running multiple sifters" on page 2331
"Preventing multiple sifters from sifting the same user" on page 2332
"Number of threads to sift users" on page 2332
"Sifter sleep time when the Lps_User_Data sift_pri field Is 0" on page 2332
"Time interval for checking sift priority" on page 2332

Sifter-specific mentor set configuration:
You can specify the following categories of `sifter`-specific configuration parameters for mentor sets:

- The engine to be used for mentor tables to be rebuilt
- Time interval for checking the sift priority
- Number of sift priority users per batch

- Maximum number of mentors assigned to each user
- Minimum ratings a user must have before becoming a mentor
- Maximum ratings a user must have before becoming a mentor
- Minimum transactions a user must have before becoming a mentor
- Maximum transactions a user must have before becoming a mentor
- How to set the mentor pool size
- `sifter` sleep time for when the Lps_User_Data table sift_pri field is 0
- Running multiple `sifters`
- Number of threads to sift users
- Recomputing (rebuilding) the mentor pool
- Pausing the `sifter` during heavy database access

How the mentor selection process works:
In order to fully understand the `sifter`, it is important to have a clear idea of how the mentor selection process works, and how to set configuration parameters that increase the accuracy of your recommendations. You should understand the following terms before proceeding:

- **Collaborative Filtering** -- Collaborative filtering (CF) is a technology that calculates the similarity between users. It uses the behaviors of those who most closely resemble any given user as a functional basis for making predictions and recommendations for that user. Given that definition, the process by which "...the entire population of users is analyzed, their fitness as mentors is calculated, and they are assigned as mentors to individual users..." is critically important to the ultimate recommendations that come from the collaborative filtering approach. All of the LikeMinds server engines, except the Item Affinity Engine, use collaborative filtering.

  The data used to decide on the levels of similarity can come in a variety of forms. From a stream of self-supplied ratings made explicitly to get recommendations back, to clickstream events that comprise the sequence, duration, and outcome of a Web-surfer's session, to data from a company's legacy databases (such as transactions, demographics, or credit events)--all can form the necessary basis for making similarity calculations. From those similarity measures, this data can result in the measured recommendations from those users deemed most similar to any given user.
- **Mentor** -- A like-minded user that is used as the basis for recommendations for new users. Every user is assigned mentors by the `sifter` program, whose stored preferences are judged to be like-minded to the new user.
- **Mentor Pool** -- While the purpose of mentors is to form the basis for recommendations for those users deemed most similar to them, in its most basic form the mentor pool should reflect a representative sample of users in the transaction set for which the recommendations are required. And despite the clearly required emphasis on similarity, no recommendation process can make lucid suggestions without a concomitant space of dissimilarity. We might base our final recommendations on the similar, shared tastes discovered in our analysis of the users being considered for entry to the mentor pool, but it is truly the confluence of similarity in purchases with some difference in the items purchased that make the LikeMinds server collaborative filtering-based recommendations possible.
- **Sifter** -- Creates mentors by analyzing stored user transactional data. The `sifter` runs in the background when you run the LikeMinds server.
- **Coverage** -- The volume or number of items rated or transactions performed.

"Mentor selection and assignment" on page 2330

"Number of sift priority users per batch"

"Pausing the sifter during heavy database"

Mentor selection and assignment:
Several factors determine the fitness of any user as a mentor. Similarity to any other user is the final arbiter of any mentor's fitness to make recommendations for a specific user. Yet it is **coverage** (the volume of number of items rated or transactions performed) that is most important when forming the mentor pool. (The mentor pool is a superset of the final mentors chosen to make the recommendations.) Although a case could be made for using the extent of the unshared purchase space as another dimension of dissimilarity, the shared space is where we find the most data (and the most predictive data) for making the required similarity distinctions within the user population.

In a nutshell, while similarity to the user is important, it is the ability of a mentor to contribute items outside of any user's typical purchasing space into the final pool of possible recommendations that qualifies the user as a possible mentor.

For this purpose, the first step the `sifter` uses in mentor assignment is to periodically create a new mentor pool (see `lps_rtg_pool` and `lps_trx_pool`) in an effort to collect a representative sample of experienced users who will then be considered as potential mentors for any user who requires recommendations.

The second step in the mentor selection process is to assign mentors from the mentor pool to be mentors for specific users. The challenge is to create a fair balance between the similarity and the coverage of users being considered as mentors. You can configure the `sifter` to emphasize similarity, coverage, or to automatically determine which dynamic to emphasize for each user in the final assignment of mentors to users.

Number of sift priority users per batch:
To specify how many users to sift during a single batch; for example:

```
<mentor_set_name>.pri_list_size = 10
```

Pausing the sifter during heavy database:
You can pause the `sifter` during times of heavy database access. This frees up database resources for other activities during heavy database activity. For example, to put the `sifter` asleep at 11 a.m.:

```
<mentor_set_name>.pause_sifting_at= 0 11 * * *
```

If you set `<mentor_set_name>.pause_sifting_at`, use the following setting to wake up the `sifter` afterwards. For example, to wake the `sifter` up at 4 p.m.:

```
<mentor_set_name>.resume_sifting_at= 0 16 * * *
```

Sifter performance considerations:
The *sifter* places high processing demands on the system running it. For this reason it is important to tune it with consideration of its usage and environment. Use the following simple guidelines when tuning the sifter:

- Schedule the sifter to build the mentor pools during hours of the day when there is low traffic volume, for example, at 3 a.m.
- Tune the number of processing threads used by the process. You can configure this setting for each mentor set.

  To optimize the resource utilization of the sifter, set the number of threads to spawn to twice the number of CPUs in the machine that the accumulator is

running on. The LikeMinds Recommendation Engine configuration value that controls this is *item_affinity_set*`.item_affinity.num_threads`.

- Distribute the sifter onto a separate server from the LikeMinds Recommendation Engine.
- Run multiple sifter processes distributed across multiple machines for greater scalability.

Recomputing (rebuilding) the mentor pool:
For better recommendations, you must recompute, or rebuild, the mentor pool at least once a day, ideally at a time when the database has little usage. This allows new users to become mentors and removes current mentors who are no longer good mentors. If you are running multiple instances of the `sifter`, use the following procedure to recompute the mentor pool:

1. Disable all instances of the `sifter` except for one, which you will use as the master `sifter` instance. Rebuilding the mentor pool uses a large amount of database resources, so it is better to use only one `sifter` to rebuild the mentor pool.

   For the `sifter` instance to be used as the master, type the following:

   ```
   <mentor_set_name>.disable_build_mentor_pool = false
   ```

   For each `sifter` instance to be disabled, type the following:

   ```
   <mentor_set_name>.disable_build_mentor_pool = true
   ```

2. Rebuild the mentor pool on the master `sifter` instance by specifying a time for the recomputation to take place.

   Because rebuilding the mentor pool requires heavy database access, schedule it for a time when the database has little usage. For example, the following setting is for 2 a.m.:

   ```
   <mentor_set_name>.build_mentor_pool_at = 0 2 * * *
   ```

   The times are interpreted as follows:

3. For each `sifter` instance you disabled, set a time to reload the mentor pool after it has been rebuilt.

   The `sifter` will destroy the current mentor pool and reload the pool from the mentor pool table specified in the `<mentor_set_name>.mentor_pool.table` setting. Usually, you should set it for about an hour after the `sifter` instance has been reloaded.

   For example, to reload the mentor pool at 3 a.m.:

   ```
   <mentor_set_name>.reload_mentor_pool_at= 0 3 * * *
   ```

Running multiple sifters:
Use the following settings to run multiple instances of the `sifter` and have all the `sifters` share some configuration parameters and override others. Running multiple `sifters` simultaneously allows you to distribute load on very large systems.

To override a particular default parameter, add a prefix to the parameter. For example, to reload the mentor pool for the MovieSift `sifter` instance at 4 a.m.:

```
MovieSift.<mentor_set_name>.reload_mentor_pool_at = 0 4 * * *
```

If you run a `sifter` with the following setting,

```
sifter -config MovieSift -conf <lps config file>
```

that `sifter` will use the`MovieSift.<mentor_set_name>. reload_mentor_pool_at = 0 3 * * *` setting.

Preventing multiple sifters from sifting the same user:
When you run multiple `sifters` simultaneously to distribute load, use the following setting to prevent multiple `sifters` from sifting the same user in the Lps_User_Data table. This setting limits the `sifter` to look at only a certain group of users, or eliminates a group of users from being sifted. For example, to allow the `sifter` to sift only users whose IDs are greater than 100000, and to ignore users whose IDs are less than 100000:

```
<mentor_set_name>.constraint = user_id > 100000
```

You can specify any field in the Lps_User_Data table for this parameter. For example, assuming there is an age column in the Lps_User_Data table, you could use the following setting to constraint the `sifter` to look at all users whose IDs are greater than 10000 and who are over 25 years of age:

```
<mentor_set_name>.constraint = user_id > 100000 and age > 25
```

Number of threads to sift users:
To set the number of threads to use for sifting users in `lpsconfig`:

```
<mentor_set_name>.num_sift_threads = 3
```

Sifter sleep time when the Lps_User_Data sift_pri field Is 0:
The Lps_User_Data sift_pri field values must be greater than 0 in order for the `sifter` to get useful data (that is, updated mentors) for users. If set to `true`, the following setting causes the `sifter` to sleep for the specified number of seconds by `<mentor_set_name>.pri_check_interval` if the sift_pri field values in the Lps_User_Data table are 0. If you set the parameter to `false`, the `sifter` sifts random users when there are no users with sift_pri greater than 0.

```
<mentor_set_name>.sleep_on_no_sift_pri = true
```

Time interval for checking sift priority:
To specify the interval in seconds for the `sifter` to check the sift priority value for a user, use the following setting. The default is 20 seconds.

```
<mentor_set_name>.pri_check_interval = 40
```

Note: You can also use this setting when you are setting the time for the `sifter` to sleep during busy periods of database access. See "Sifter performance considerations" on page 2330.

Ratability parameters: The `buildvisit` background utility computes a ratability value for each item in the database. It is only used for the Preference Engine. The LikeMinds server presents items for rating in decreasing order of ratability; items with the same ratability are presented in random order.

The ratability value is computed as

ratability = $A_i+B_i+C_i$

where A_i is the popularity of item $_i$:

$A_i=Kx(\log_{10}(num\_rating))^{rpow}$

num_rating is the number of ratings for the item.

To specify K, set `db.ratability.num_rating.coefficient` (default is 1.0):

```
db.ratability.num_rating.coefficient = 1.0
```

To specify rpow, the rating power, set `db.ratability.num_rating.power` (default is 1.0):

```
db.ratability.num_rating.power= 1.0
```

B_i is a measure of how controversial the item is:

$B_i=Kx(rating\_std\_dev)^{sdpow}$

rating_std_dev is the standard deviation of the ratings for the item.

To specify K, set `db.ratability.stddev.coefficient` (default is 1.0):

```
db.ratability.stddev.coefficient = 1.0
```

To specify stdpow, the standard deviation power, set `db.ratability.stddev.power` (default is 1.0):

```
db.ratability.stddev.power= 1.0
```

$C_i$is a weight based on the age of the item:

$C_i=Kxexp(-(current\_year - release\_year))\ exp\ \hat{}\ apow$

release_year is the value of the year field in the `Lps_Item_Data` table for the item.

To specify K, set `db.ratability.age.coefficient`. You can set this to zero if you do not want to consider age in determining ratability (default is 2.7):

```
db.ratability.age.coefficient = 2.7
```

To specify ^{apow}, set `db.ratability.age.power` (default is 0.5):

```
db.ratability.age.power= 0.5
```

Repeated items in visit list: To specify whether the `buildvisit` utility should repeat an item in the visit list to be shown to a user, set `db.visitlist.ratability.duplication_threshold.` Items with ratability values greater than this threshold may be repeated.

For example:

```
db.visitlist.ratability.duplication_threshold = 6
```

Configuring the Clickstream Engine:
The Clickstream Engine generates recommendations based on a record of user shopping behavior as the user navigates through a Web site; that is, a history of user "clicks" during the user's website visit. It uses data such as the items the users view, click, and add to their shopping carts.

You can use the “User predictability for the ClickStream Engine” on page 2334 setting to specify recommendation behavior for the Clickstream Engine.

“User predictability for the ClickStream Engine” on page 2334

User predictability for the ClickStream Engine:
This section contains the following information regarding user predictability:

- The minimum number of clickstream activities required for a user before he or she can receive recommendations .
- The number of mentors the Clickstream Engine will examine before it checks to see whether the user is predictable.
- The percentage change allowed in a user's transactions before the LikeMinds server recomputes the user's predictions.
- Default recommendations.

"Minimum number of Clickstream activities for a user"

"Minimum mentors the engine examines for predictability"

"Recomputing Clickstream Engine predictions"

"Default Clickstream Engine recommendations" on page 2335

Minimum number of Clickstream activities for a user:
Set `<eng_instance_name>.engine.saturn.tune.predictable.min_xactions_cutoff` to the minimum number of transactions a user must make before the Clickstream Engine will make recommendations for that user. The default is 10.

For example:

```
movie_click.engine.saturn.tune.predictable.min_xactions_cutoff = 2
```

Minimum mentors the engine examines for predictability: When it determines whether it can make recommendations for a given user, the Clickstream Engine must examine the user's mentors.
`<eng_instance_name>.engine.titan.predictable.loop_check_cutoff` specifies the number of mentors the Clickstream Engine will examine before it checks to see whether the user is predictable. The default is 10.

For example:

```
movie_pref.engine.titan.predictable.loop_check_cutoff = 12
```

If the user is not predictable, the engine will examine another group of `<eng_instance_name>.engine.titan.predictable.loop_check_cutoff` mentors, and then check to see whether the user is predictable. It will continue until it finds that the user is predictable or it runs out of mentors.

Recomputing Clickstream Engine predictions:
The `<eng_instance_name>.engine.saturn.recomputation_bound` configuration parameter specifies the percentage change allowed in a user's transactions before the LikeMinds server recomputes the user's predictions. For example:

```
music_click.engine.saturn.recomputation_bound = 10.0
```

Ordinarily, the LikeMinds server generates predictions based on a user's mentors, which the `sifter` computes and makes available to the database. When a user has no mentors, perhaps because they have just arrived at the site, or when the user's ratings or transactions have changed beyond the percentage specified here, the LikeMinds server selects mentors from a reduced set of candidates and recomputes the user's predictions.

Use this setting with caution, as selecting mentors is a relatively expensive operation: a low percentage setting can lead to excessive CPU load with little or no gain in prediction quality. A higher setting will improve performance, but predictions may be less accurate.

Default Clickstream Engine recommendations:
The Clickstream Engine uses transaction data stored with a special user to generate default predictions. The `buildstats` utility generates clickstream transaction data, based on the transaction set named in the `db.applic.transaction.source` parameter. This transaction data is a summary of all clickstream transaction data for users, based on transaction data from all configured transaction sets. `buildstats`, the LikeMinds server, and other utilities identify the special user as one whose Lps_User_Data `mentor_type` field is set to "z". There should be exactly one user with this setting, not more. `buildstats` will automatically generate this user if custom IDs are not configured; otherwise, you need to create the user by hand, since `buildstats` cannot know what restrictions apply to the custom IDs.

Run the `buildstats` utility on a regular basis. By default, the installer sets `buildstats` to run once a day. Some applications use their own business logic to assign the score to items. If you want to override the score field value to use your application's scoring instead, run `buildstats` using the `-noscore` argument.

Configuring the Item Affinity Engine:
For every cross-selling transaction in the user's shopping history, the Item Affinity Engine derives its calculations from the following statistics:

- **Support**: This is the number of times an item pair is involved in the item affinity set you defined, divided by the total number of paired item affinity events. Item affinity events can be critical page views, purchases, shopping cart adds or drops, or combinations of these events. In other words, the support is the percentage of the time that the item pair occurred together, relative to all transactions.
- **Prediction**: Using the support statistic as a basis, the prediction is the number of times an item-to-item pair occurs together, divided by the singular support of the first item of that item pair. In other words, the prediction is the probability of the item pair, given the first item in the pair.
- **Confidence**: Using the prediction statistic as a basis, confidence is the prediction divided by the singular support of the second item of the pair for which the prediction is computed. That is, the confidence is the probability of the item pair given the first and second item in the item pair.

In general terms, you use the Item Affinity Engine as follows:

1. Define the shopping cart analysis requirements.

   This refers to the scope and filters for predictions you plan to use for the shopping cart analysis. This definition is very flexible. For example, the shopping cart can be based on the user's entire shopping history, on seasonal product sales, on the user's current session, user page views critical to an item's purchase, and so on.

   The type of purchases you expect from your users will determine the type of shopping basket you define. For example, a session-based shopping cart would be suitable for a grocery store site, where users tend to purchase a variety of items at once. A lifetime shopping cart is suitable for a site where users tend to purchase one large item occasionally, such as expensive electronic equipment.
2. Configure an item affinity set for the shopping cart analysis.

   The item affinity set defines your shopping cart analysis requirements. Similar to mentor sets or transaction sets, the item affinity set defines the type of data you want to collect. The item affinity set starts by collecting transactions from a transactions input table similar to Lps_User_Trx. You can include filters to query fields in the input table, specify the types of transactions to query, the number of highly associated items to consider, and so on, for the shopping cart

definition. Finally, you specify an output table to which the results of the analysis are written. The output table must have the same fields and datatypes as the Lps_MBA_Scored table.

3. Run the `accumulator` utility to collect the item affinity set data and populate it into the output table specified in the item affinity set.

   Similar to the `sifter`, which is used for the other LikeMinds server engines, the `accumulator` uses the item affinity set data to make the support, prediction, and confidence calculations (described earlier). It writes its findings to the output table specified in the item affinity set.

   You should configure the `accumulator` to run at times of low database usage, since it can make heavy use of system resources.

Specifying recommendation behavior:

You can configure several parameters that affect the way the LikeMinds server generates recommendations.

You can set the following recommendation behavior:

- Allowable rating values
- Allowable confidence levels
- Prediction quality
- Best bets

The parameters in this section affect the way the LikeMinds server generates recommendations. If you change them, be sure that the results will still be appropriate for your application. Because the Movie Site application expects these parameters to be set to their default values, Movie Site application developers should use extreme caution when modifying them. Parameters that apply to the background applications can also affect the way recommendations are generated.

"Allowable rating values"
Learn about the parameters that govern the allowable range of ratings.

"Allowable confidence levels" on page 2337
The LikeMinds server assigns a confidence level to each recommendation based on how many users have rated the recommended item and how similar the ratings are to each other. "Confidence" refers to the accuracy of the prediction. Learn how to set allowable confidence levels.

"Prediction quality values" on page 2337
The "quality" value refers to the degree a user will like an item. The LikeMinds server presents predictions in decreasing order of quality.

"Best Bets values" on page 2338
The LikeMinds server generates recommendation vectors that include all recommendations for a given user in order from best quality to the least. You can configure the number of items to be returned as Best Bets and the maximum percentage of the total recommendation vector to include in the Best Bets list.

Allowable rating values:

Learn about the parameters that govern the allowable range of ratings.

Specify the allowable range of ratings by setting `db.applic.param.lowestrating` (default is 0) to the lowest allowable rating and

db.applic.param.numberratingvalues(default is 13) to the number of possible ratings. This setting also determines the recommendation value. This example specifies a rating scale of zero to twelve:

```
db.applic.param.lowestrating = 0
db.applic.param.numberratingvalues = 13
```

The parameters db.applic.param.wontrate (default is -1) and db.applic.param.unrated (default is -2) set special allowable rating values that are not part of the normal rating scale. Setting an item to unrated marks the rating for deletion. wontrate means that a user has specifically indicated that he or she does not plan to rate the item.

```
db.applic.param.wontrate = -3
db.applic.param.unrated = -2
```

The LikeMinds server ignores ratings which are neither in the specified range of rating values nor equal to db.applic.param.wontrate or db.applic.param.unrated.

Note: You can use these parameters for all of the Recommendation engines except for the Item Affinity Engine.

Allowable confidence levels:

The LikeMinds server assigns a confidence level to each recommendation based on how many users have rated the recommended item and how similar the ratings are to each other. "Confidence" refers to the accuracy of the prediction. Learn how to set allowable confidence levels.

You can specify the range of confidence values by setting db.applic.param.lowestconfidence (default is 1) to the lowest confidence level and db.applic.param.numberconfidenceLevels (default is 4) to the number of confidence levels. You can set confidence levels for all of the Recommendation engines.

```
db.applic.param.lowestconfidence = 1
db.applic.param.numberconfidencelevels = 4
```

Prediction quality values:

The "quality" value refers to the degree a user will like an item. The LikeMinds server presents predictions in decreasing order of quality.

Recommendation quality is calculated using this equation:

quality = $\text{confidence}^{\text{mpow}}$x (pred rating —mean pred rating) + K x $\text{confidence}^{\text{pow}}$

where:

- **confidence** is the confidence value for the recommended item.
- **pred rating** is the predicted rating for the item.
- **mean pred rating** is the mean recommended rating for all items for the user.

To specify $^{\text{mpow}}$, set db.applic.param.BestBets.MultPower:

```
db.applic.param.BestBets.MultPower = 0.5
```

To specify K, set db.applic.param.BestBests.Coefficient:

```
db.applic.param.BestBets.Coefficient = 0
```

To specify pow, set db.applic.param.BestBets.Power:

```
db.applic.param.BestBets.Power = 1.0
```

Note: You can use these parameters for all of the Recommendation engines except for the Item Affinity Engine.

Best Bets values:

The LikeMinds server generates recommendation vectors that include all recommendations for a given user in order from best quality to the least. You can configure the number of items to be returned as Best Bets and the maximum percentage of the total recommendation vector to include in the Best Bets list.

The LikeMinds server produces Best Bets by beginning with the highest-quality item in the recommendation vector and listing items in decreasing order of rating quality.

To specify how many items the LikeMinds server should return as Best Bets, set db.applic.param.BestBets.Threshold. The default value of this parameter is the number of rating levels (specified by the db.applic.param.NumberRatingValues parameter) divided by 2. For example, if db.applic.param.NumberRatingValues were set to 12, the default would be 6. You can specify a different value, for example:

```
db.applic.param.BestBets.Threshold = 7
```

Set db.applic.param.BestBets.list.cutoff to the maximum percentage of the total recommendation vector to include in the Best Bets list. For example:

```
db.applic.param.BestBets.list.cutoff = 50
```

MovieSite Sample:

The MovieSite documentation focuses on six aspects of LikeMinds capabilities:

Best Bets
: The Best Bets task provides a list of recommended movies for each user. These recommendations are calculated based on a user's previous ratings, as well as the ratings of other similar users, using collaborative filtering. The system determines users' similarities by comparing their ratings history among all users. For example, if both user A and user B have many similar ratings, but user B has rated many more different movies; user B might make a great mentor for user A. It is from the set of mentors (user B and other similar users) that LikeMinds makes recommendations, in this case a movie recommendation for user A.

Get Items to Rate
: The Get-Movies-to-Rate task retrieves movies, which the user has not yet rated, from the database. These movies are selected based on the impact that a rating from the user would have on LikeMinds' ability to recommend movies (Best Bets). (See the task example.)

View the User's Previous Ratings
: This task retrieves ratings that a user previously entered from the LikeMinds database. An item that has previously been rated can always be given a different rating. (See the task example.)

Log Item Ratings

The Personalization rating bean logs each user rating to the LikeMinds database. There are a few requirements to remember when using the rating bean:

- The session attribute "pzn.userName" must be set with the user resource ID. This is how LikeMinds pairs the user with an item rating.
- If you log an item that previously did not exist in the LikeMinds database table, an entry will automatically be created for that item.
- In the same way, if an item is logged for a user who does not exist in the LikeMinds database table, LikeMinds creates an entry for that user. This is how a "new user" is to be entered into the LikeMinds system.
- The resource collection that is sent as a parameter to the logging bean is case sensitive.

Database Access

Developers are responsible for maintaining their own database tables. In MovieSite's case, this includes MovieSite's user table, item table, and also genre tables. All MovieSite tables begin with **MS*** and LikeMinds tables begin with **Lps***. To implement filtering, the filter ID columns must be added to the LPS_ITEM_DATA table. These filter IDs are populated when a movie is logged for the first time, or the IDs are set directly. For more information on filtering, see the section about filtering recommendations that follows.

Filtering Recommendations based on filter IDs

LikeMinds implements a filtering system for each of the Personalization rules that query LikeMinds. This system filters items at the LikeMinds level according to any number of filter ID's associated with each item. Only the items which satisfy the filter are surfaced to the content spot in the JSP.

"Exploring Movie Site"
Movie Site demonstrates the LikeMinds Recommendation Engine, guiding you through a Personalization scenario that is based on factual data from a site on the internet. The website uses a personalization solution to analyze visitor behavior and to recommend individualized content information and services while the visitor is actively engaged on the site.

Exploring Movie Site:

Movie Site demonstrates the LikeMinds Recommendation Engine, guiding you through a Personalization scenario that is based on factual data from a site on the internet. The website uses a personalization solution to analyze visitor behavior and to recommend individualized content information and services while the visitor is actively engaged on the site.

Notes:

- The provided Resource Collections are for IBM DB2 Universal Database Enterprise Server Edition and DB2 for z/OS only. If you want to use the sample on other database types, you can regenerate them using Rational Application Developer. For information about Resource Collections, see the topic *Resources, resource instances, and resource collections*.
- It is assumed that you are the Movie Site user, `likeminds`. This user previously rated some movies and registered preferences as configured in the installation.

To set up the Movie Site sample, choose the appropriate option:

- Linux : `./ConfigEngine.sh cfg-likeminds-samples -DWasPassword=password -DLikemindsDbPassword=password`
- IBM i: `ConfigEngine.sh cfg-likeminds-samples -DWasPassword=password -DLikemindsDbPassword=password`
- Windows: `ConfigEngine.bat cfg-likeminds-samples -DWasPassword=password -DLikemindsDbPassword=password`

Note: Check the output for any error messages before you proceed with the next task. If any of the configuration tasks fail, verify the values in the `wkplc.properties` file.

Complete the following steps to start the application:

1. Open the WebSphere Integrated Solutions Console.
2. Go to `http://machine:port_number/ibm/console` where *port_number* is the port number for WebSphere Portal Express.
3. Click **Applications** > **Application Types** > **WebSphere enterprise applications**.
4. Select **Movie Site Application**, and click **Start**.
5. Rate movies by using the Movie Site home page.

Table 360. A quick demonstration of how to rate movies by using the Movie Site home page

Step number	Description	Action to be taken
1	Navigate to the Movie Site Home Page. For this demo, we concentrate on the recommendation engine, which uses mathematical algorithms to provide collaborative filtering.	Open a separate browser window to the Movie Site home page. The URL is `http://hostname:port/MovieSite` where *port* is the port number for WebSphere Portal
2	Log in as the LikeMinds user. Log on as the user, `likeminds`. This login name is the default user who registered to the site and rated some movies.	1. Type `likeminds` in the login field. 2. Type `likeminds` in the password field 3. Click **Enter**.

Table 360. A quick demonstration of how to rate movies by using the Movie Site home page (continued)

Step number	Description	Action to be taken
3	Sample previous ratings. Look at the movie ratings by selecting **Sample Previous Ratings**. This section of the site contains a few sample movie ratings the user `likeminds` previously placed.	1. Click **Sample Previous Ratings**. 2. Select **Get more movies** from the menu. 3. Click the **To Lobby** icon. You can show all of the movies or certain rated movies by selecting one of the ratings. You can look at the ratings of other movies by clicking **Get More Movies**. This demonstration allows you to rate movies. For more movies, type the name of a movie into the search panel to see whether specific movies are stored in the database.
4	Use the SuperRater to rate a movie. In Super-Rater mode, clicking the drop-down menu that is associated with each movie of the movie reveals a rating choice. You can rate any movies on the list and click **Submit** to log your rating choice. If you click the movie title, it links you to information about the movie by using the `imdb.com` site. Notice the prediction of the recommendation engine of how much you, as the LikeMinds login user, would like this movie.	1. Click **To Lobby**. 2. Click **Rate more movies**. 3. Click **Turn on Superrater**. 4. Click the dropdown menu that is associated with the rating to see the options. 5. Click **Movie title**. **Note:** This action opens a new window.
5	End demonstration.	Click **To Lobby**.

The LikeMinds collaborative filtering technology considers each user to be the center of their own preference cluster, instead of trying to find the best preordained preference category for every user, which is typical of profile-based approaches. LikeMinds recommendations can adapt to these changes much more quickly than aggregated, profile based techniques.

Choose the appropriate task to remove the application:

Linux : `./ConfigEngine.sh remove-likeminds-samples -DWasPassword=`*`password`* `-DLikemindsDbPassword=password`

IBM i: `ConfigEngine.sh remove-likeminds-samples -DWasPassword=password -DLikemindsDbPassword=password`

Windows: `ConfigEngine.bat remove-likeminds-samples -DWasPassword=password -DLikemindsDbPassword=password`

Note: Check the output for any error messages before you proceed with the next task. If any of the configuration tasks fail, verify the values in the `wkplc.properties`, `wkplc_dbdomain.properties`, and `wkplc_dbtype.properties` files.

Using the LikeMinds utilities:

Learn about the four utilities LikeMinds uses to update the database, `buildvisit` (for the Preference Engine), `sifter`, `buildstats`, and `lpsIAA` (for the Item Affinity Engine's accumulator utility).

The LikeMinds utilities are run from the LikeMinds utility servlet.

You must configure these utilities to run at appropriate times. See "Scheduling LikeMinds Events" on page 2322 for more information.

"Sifter"
The `sifter`, which is used for all the Recommendation engines except for Item Affinity, assigns mentors to users and updates its assignments as users rate items. When no users are rating items, the sifter updates the mentor assignments of all users based on changes to the pool of available mentors.

"`Buildstats`"
The `buildstats` utility updates database statistics. You can use `buildstats` for all of the Recommendation engines except for the Item Affinity Engine.

"Accumulator" on page 2343
Use the `accumulator` utility if you are using the Item Affinity Engine. This utility accumulates the number of times every possible item-to-item combination occurs.

Sifter:

The `sifter`, which is used for all the Recommendation engines except for Item Affinity, assigns mentors to users and updates its assignments as users rate items. When no users are rating items, the sifter updates the mentor assignments of all users based on changes to the pool of available mentors.

The `sifter` must be running whenever the LikeMinds server is running. However, you can install the `sifter` across multiple machines to distribute load.

`Buildstats:`

The `buildstats` utility updates database statistics. You can use `buildstats` for all of the Recommendation engines except for the Item Affinity Engine.

When you install the LikeMinds utilities, the installer sets `buildstats` to run daily at 3:30 a.m. This run time is because the best time to run `buildstats` is daily, at a time when the LikeMinds server receives relatively little use. The `buildstats` utility connects directly to the LikeMinds database and processes the transactional or ratings data that is constantly being written to the LikeMinds database. Because this utility does not use many resources, it is typically installed on the same server as the LikeMinds server. `buildstats` runs a number of data analysis functions on the information in the LikeMinds database and persistently writes out the results

into the LikeMinds schema. However, the function and the number of functions to test differ depending on the type of LikeMinds engines in use. If both a Preference and ClickStream are in use, `buildstats`, by default, analyze both the ratings and the transactional data.

For ratings data, `buildstats` writes to the following fields in the Lps_Item_Data table:

- **`num_rtg`**: Total number of ratings for the item.
- **`total_rtg`**: Sum of all ratings for the item.
- **`total_square_rtg`**: Sum of the squares of all ratings for the item.
- **`ratability`**: Priority for an item's presentation to users for rating. **`ratability`** is non-negative value. Higher numbers indicate a higher priority for rating.
- **`score`**: Popularity or unpopularity of an item. If you prefer to have your own applications update this field, you can disable `buildstats` from writing to score.

For transactional data, `buildstats` writes to these fields in Lps_User_Trx:

- **`value`**: Data value that is associated with the transaction.
- **`adj_count`**: Adjusted count of transactions. If a new activity is recorded in the transaction table, this value increases. It might diminish over time.

Accumulator:

Use the `accumulator` utility if you are using the Item Affinity Engine. This utility accumulates the number of times every possible item-to-item combination occurs.

Similar to the `sifter` for the other LikeMinds engines, the `accumulator` runs in the background processing item event data. It writes this data to the table specified by the `<item_affinity_set>.item_affinity.output.table` setting. The `accumulator` is installed as part of the PZN_Utilities.ear. It is run through the LikeMinds scheduling mechanism.

Filtering LikeMinds recommendations:

When LikeMinds makes recommendations, it can make the recommendations based on all items in your resource collection, or it can limit the predictions to only items that have certain characteristics.

Tell LikeMinds about an item by including key/value pairs describing its characteristics as you log actions and ratings that occur against it. The format for the key/value pair is

```
Key = LMFilter.<item_characteristic>
value = <value>
```

For example, to tell LikeMinds that an item's color is blue, and its category is sports, you would add the 2 key/value pairs:

```
LMFilter.color,blue
LMFilter.category,sports
```

Specify which characteristics that you want LikeMinds to use in making predictions by setting request attributes in the RequestContext object immediately before the content spot that contains the LikeMinds rule. To specify characteristics for filtering, do the following:

- Add a request attribute that tells LikeMinds which characteristic or characteristics you want to filter on. The name of that attribute should be

LMFilter and the value should be a string or an array of strings, where the string or each string in the array is a characteristic that you would like to filter on.

For example, to return predictions only from among items whose category is "clearance" and season is "spring" or "summer", you would add the following code before the content spot:

```
com.ibm.websphere.personalization.RequestContext.context =
com.ibm.servlet.personalization.context.PersonalizationContext.getRequestContext(httpRequest);

context.setRequestAttribute("LMFilter", new String[] { "LMFilter.category", "LMFilter.season" });

context.setRequestAttribute("LMFilter.category", "clearance");

context.setRequestAttribute("LMFilter.season", new String[] { "spring", "summer" });
```

To return predictions only from items whose color is blue, you would add the following code before the content spot:

```
com.ibm.websphere.personalization.RequestContext.context =
com.ibm.servlet.personalization.context.PersonalizationContext.getRequestContext(httpRequest);

context.setRequestAttribute("LMFilter", "LMFilter.color");

context.setRequestAttribute("LMFilter.color", "blue");
```

Feedback and analytics

Personalization provides a complete logging framework for collecting data on how visitors are using your Web site. If Feedback is enabled, data is automatically collected about each Personalization rule that is fired. In addition, development tools enable Web sites to collect a variety of data related to visitors' actions and behavior. By default this data is logged to a standard database schema for later analysis and reporting. The framework is also extensible, allowing Web sites to customize and supplement the way data is collected and stored to more fully meet their needs.

Personalization provides a complete logging framework for collecting data on how visitors are using your website. If Feedback is enabled, data is automatically collected about each Personalization rule that is fired. In addition, development tools enable Web sites to collect a variety of data related to visitors' actions and behavior. By default this data is logged to a standard database schema for later analysis and reporting. The framework is also extensible, allowing Web sites to customize and supplement the way data is collected and stored to more fully meet their needs. Using the Personalization Feedback framework you can collect data to help answer questions such as:

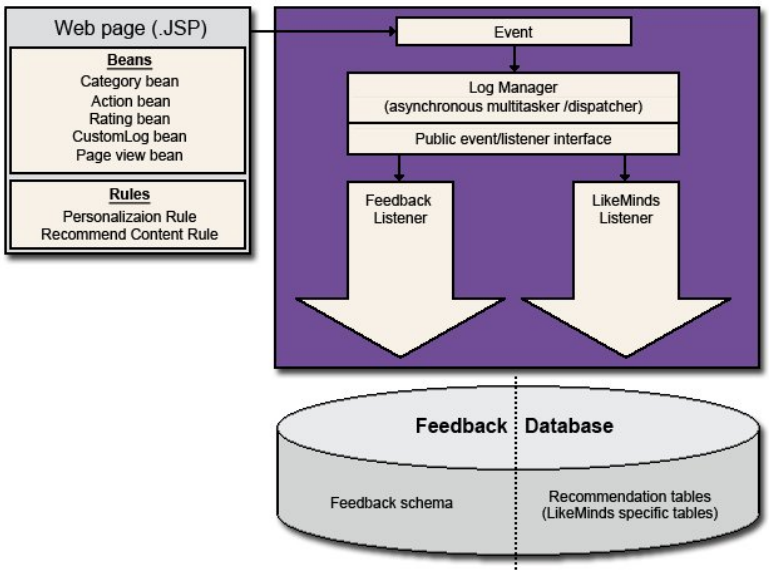
- How effective are the new campaign initiatives?
- Which items have the highest clickthrough ratio?
- What is the clickthrough ratio for health care products?
- Which items have the highest view to purchase ratio?
- What is the ratio for health care products?
- Which rules are resulting in the most number of downloads?
- During the month of December, which categories of pages were browsed most often (my special Christmas pages, or the regular part of my site)?

"Feedback subsystem overview" on page 2345
The Feedback listener captures data and writes it to the Feedback database schema for subsequent reporting, a capability which can be used to persist data in ways that specifically meet your requirements.

Feedback subsystem overview:

The Feedback listener captures data and writes it to the Feedback database schema for subsequent reporting, a capability which can be used to persist data in ways that specifically meet your requirements.

Logging occurs when a rule or logging bean is present in JSP or servlet a user visits. In either case, during execution an event containing data to be logged is generated and dispatched by the log manager to a set of listeners.Personalization supplies two such listeners. The first is the Feedback listener, which captures data and writes it to the Feedback database schema for subsequent reporting. The second is the LikeMinds listener (LMListener), which captures data for subsequent use by the LikeMinds recommendation engine. Web site developers can optionally extend the provided listeners or create additional custom listeners to persist data in ways that specifically meet their needs.



- Logging events are generated by Personalization rules or logging beans.
- The Feedback subsystem's multithreaded implementation prevents HTTP clients from waiting.
- The Log Manager maintains internal queue of events.
- Event is dropped on queue then control returned to web client.
- Events are dispatched from the queue to all registered listeners.
- Listeners in turn persist and process the data contained in the logging event.

Enable logging:

Create and set up the Feedback database. Then, enable logging on the runtime server that hosts Feedback data.

About this task

Choose the appropriate option to create, set up, and enable logging of the Feedback database:

Table 361. Operating system options to create, set up, and enable logging of the Feedback database

Operating system	Task
Linux	Complete the following steps: 1. Change to the *wp_profile_root*/ConfigEngine directory. 2. Create the Feedback database using the following command: `./ConfigEngine.sh feedback-database -DInitalizeFeedbackDB=true -DWasPassword=password` 3. Set up the Feedback database using the following command: `./ConfigEngine.sh setup-feedback -DWasPassword=password` 4. Open the FeedbackService.properties file from the *wp_profile_root*/PortalServer/config/config/services/ directory. 5. Set **loggingEnabled** to true. 6. Restart WebSphere Portal Express.
IBM i	Complete the following steps: 1. Change to the *wp_profile_root*/ConfigEngine directory. 2. Create the Feedback database using the following command: `ConfigEngine.sh -profileName profile_root feedback-database -DInitalizeFeedbackDB=true -DWasPassword=password` 3. Set up the Feedback database using the following command: `ConfigEngine.sh -profileName profile_root setup-feedback -DWasPassword=password` 4. Open the FeedbackService.properties file from the *wp_profile_root*/PortalServer/config/config/services/ directory. 5. Set **loggingEnabled** to true. 6. Restart WebSphere Portal Express.

Table 361. Operating system options to create, set up, and enable logging of the Feedback database (continued)

Operating system	Task
Windows	Complete the following steps: 1. Change to the *wp_profile_root*\ConfigEngine directory. 2. Create the Feedback database using the following command: `ConfigEngine.bat feedback-database -DInitalizeFeedbackDB=true -DWasPassword=password` 3. Set up the Feedback database using the following command: `ConfigEngine.bat setup-feedback -DWasPassword=password` 4. Open the FeedbackService.properties file from the *wp_profile_root*\PortalServer\config\config\services\ directory. 5. Set **loggingEnabled** to true. 6. Restart WebSphere Portal Express.

Feedback properties file:

The FeedbackService.properties file contains several customizable properties that are used by the Feedback component. These properties are used to enable custom log listeners, to tune the performance of the Feedback data collection system.

The FeedbackService.properties file is installed to the *wp_profile_root*/PortalServer/config/config/services directory.

To modify the properties:

- Edit the FeedbackService.properties file.
- Restart the IBM WebSphere Application Server where the Portal Personalization Runtime is installed.

loggingEnabled

Before information about the usage of your site can be recorded, logging must first be enabled. To enable logging, set the loggingEnabled property to true.

```
loggingEnabled = true
```

schemaName

It is possible to customize the name of the feedback schema used in the database. For most platforms the Personalization installer will create the feedback tables under a schema named FEEDBACK.

For IBM i installations, the schemaName property must be set to the name of the database library that contains the Feedback tables (for example, QWPS51).

```
schemaName = FEEDBACK
```

logListeners

Log events are generated whenever the log method of a bean is called or whenever a rule is fired. Log listeners process these log events and can store the event data or perform other actions as a result of these events. The Feedback component provides two default log listeners, the `LMListener` that collects data for use by the `LikeMinds` Recommendation Engine and the `FeedbackListener` that collects data for use in the Feedback reports. Custom log listeners can be used to modify the default behavior of the `FeedbackListener` or to provide a listener that processes the Feedback events in a user-specified manner.

After being developed, custom listener class files must be placed in the `classpath` of the server where Personalization run-time is installed. The class names of these listeners must be identified in the logListeners property. Multiple custom listener class names are separated by semicolons:

```
logListeners =
LogListenerClassName1;LogListenerClassName2
```

logBufferSize

When log events are produced by logging beans or rules, they are placed on a process queue. Each of the registered log listeners consumes events from this queue. This enables listeners to consume log events at independent rates, minimally impacting the performance of other listeners. It also enables the Feedback component to queue events originating from Web page requests with minimal impact on the Web site performance.

The log event queue (or log buffer) has a default maximum size of 10,000 events. The optimal size of this buffer is affected by Web site volume, Web site usage spikes, and many other Web application performance factors. You can change the maximum size of the log buffer by modifying the `logBufferSize` property in the `FeedbackService.properties` file:

```
logBufferSize = 10000
```

logWarningLevel

As the log buffer begins to fill due to heavy Web site loads or slow log listener performance, a warning message can be logged to the WebSphere Integrated Solutions Console. The `logWarningLevel` specifies the log buffer load that will trigger warning messages. The `logWarningLevel` is specified as a percentage value between 0 and 100. Specifying a value of 0 will repeatedly output the current log buffer load whereas a percentage value of 100 will result in no warning messages as the buffer fills. Once the log buffer is full, an error message is logged to the WebSphere Integrated Solutions Console and logging is discontinued until room becomes available in the log buffer. The default value for the log warning level is 75%. You can change this warning level by modifying the `logWarningLevel` property in the `FeedbackServices.properties` file:

```
logWarningLevel = 75
```

requestFlushInterval

Multiple rules or logging beans can be placed on a single Web page or JSP. The Feedback component collects all the data from the rules and log beans on a page and then updates the Feedback database with all of the information for the page

(HTTP request). Updating the Feedback database only once per page improves the performance of the Feedback component. It is necessary to determine when all of the data for a page request has been collected. Two mechanisms are used in the current release to determine when the log data for a page request should be written to the Feedback database. The first method is to flush all of the log data for a page request whenever a new page request is received from the same user session. The second method uses a time interval to determine when the response to a page request has been completed and, thus, all the logging data for the request has been generated.

The `requestFlushInterval` applies to this second method of flushing the logging data for a page request. It is the time interval during which no new logging data has been received during a user session after which the log data for the most recent request is assumed to be complete and is flushed to the database. The default value for the request flush interval is 10 seconds. You can modify this interval by changing the `requestFlushInterval` property (specified in milliseconds) in the `FeedbackService.properties` file:

```
requestFlushInterval = 10000
```

inactiveListenerInterval

It is possible for a log listener to become inactive due to a run-time error or resource deadlock. The result of such a deadlock would be an accumulation of data in the log buffer that will never be processed. Since the log buffer is finite in size, it will eventually reach capacity. Once it reaches capacity, no new data will be logged and all active listeners will be unable to collect and process additional log data. To protect against this scenario, the activity of all listeners is monitored. If log data is available for a listener to process and the listener remains inactive for a pre-determined, the log listener will be removed from the set of registered log listeners. The default period of inactivity is 2 minutes. You can modify this interval by changing the `inactiveListenerInterval` property (specified in milliseconds) in the `FeedbackService.properties` file:

```
inactiveListenerInterval = 120000
```

workManager

If running in an environment that supports it, the Feedback component uses the WebSphere Application Server `workmanager` system for asynchronous operations. Using the `workManager` property, you can specify the name of the `workmanager` that Feedback will use.

```
workManager = wm/default
```

Rule logging:

If logging is enabled, rule information is automatically logged whenever a rule is executed.

When a content spot's rule retrieves content, a RuleEvent object is constructed from the request, campaign name, rule name, resource collection name, and items. This event object is routed to all registered log listeners and the logged data are stored in the Feedback schema.

Logging beans:

Use logging beans to log data about a Web site visitor's actions, category interests, and ratings.

You can also use custom beans to log information specific to your Web application. You insert logging beans into your JSPs. The types of logging beans are:

- Action beans
- Rating beans
- Category beans
- CustomLog beans
- Page View beans

"Action beans"
You use Action beans to log specific actions of your Web site visitors.

"Category beans" on page 2354
Content categories enable you to classify Web user interests. You use Category beans to log content categories.

"Rating beans" on page 2356
The Rating bean is a specialized logging bean. You use Rating beans in applications that solicit user ratings for web content or other personalization resources.

"CustomLog beans" on page 2357
CustomLog beans give you the flexibility to log data that does not easily map to the other logging beans. You use CustomLog beans to log application specific data. You define the required key name(s) as well as the value(s) for that key.

"PageView beans" on page 2358
You use PageView beans to log HTTP request data for JSPs (JavaServer pages).

Action beans:

You use Action beans to log specific actions of your Web site visitors.

As a Web site visitor navigates your Web pages, an Action bean logs the visitor's events within the session. Action beans maintain action information for the current session including the actions logged and their corresponding log counts. This session data can be used within rules of from your JSPs. Action information is also routed to all registered log listeners.

All actions are logged to the Feedback schema and to the user session. The actions logged include the pre-defined actions listed in Table 362 on page 2352 and user defined actions. The actions that are logged to the Feedback schema and user session do not need to be pre-defined.

If you have LikeMinds installed, only the pre-defined actions listed in Table 362 on page 2352 are logged to the LikeMinds schema. These actions must be defined to LikeMinds in the lps_trx_type table. (You can also add to the list of predefined actions by adding items to the lps_trx_type table.) The actions logged through an Action bean provide the transaction data necessary for the LikeMinds Clickstream Engine. This set of actions may be accessed after deployment and configuration of a Web application.

Actions can be logged with or without respect to a specific resource. For example, the "OrderCancel" pre-defined action does not apply to a specific resource whereas the "BrowseContent" pre-defined action applies to a specific content resource.

You can create rules based on the activity of the current session, for example:

```
WebSite Visitor is
    Undecided when
       current Action Count.ShoppingCartDelete > 0
     Otherwise Decided
```

Table 362. Pre-defined Actions

Action ID	Action Name
1000	DetailedView
1001	Purchase
1002	Rated
1003	ShoppingCartInsert
1004	ShoppingCartDelete
1005	OrderCancel
1006	ItemView
1007	WishlistAdd
1008	WishlistDelete
1009	BrowseContent
1010	Search
1011	ShoppingCartUpdate

"Implementing action logging"
To implement action logging, insert an Action bean into your JSP. To log additional application data associated with the action, add key/value pair information to the log method call.

"Action beans reference" on page 2353
View some additional information related to Action beans and associated methods.

Implementing action logging:

To implement action logging, insert an Action bean into your JSP. To log additional application data associated with the action, add key/value pair information to the log method call.

About this task

To implement action logging, insert an Action bean into your JSP, for example:

```
<jsp:useBean class="com.ibm.wcp.analysis.beans.Action" id="action" scope="session"/>


<%
   // Note: Both the resource id and collection name were added to the request
   // by the referral page.
   action.log( request,
               request.getParameter( "resourceId" ),
               request.getParameter( "collectionName" ),
               "ItemView" );
%>
```

To log additional application data associated with the action, add key/value pair information to the log method call in. For example:

```
<jsp:useBean class="com.ibm.wcm.analysis.beans.Action" id="action" scope="session"/>

<jsp:useBean class="ShoppingCart" id="cart" scope="session"></jsp:useBean>

<%
   // Log last item added to shopping cart with quantity and size data.
   Hashtable actionInfo = new Hashtable();
   Product[] cartItems = cart.getItems();
   actionInfo.put( "quantity",
                   new String( cartItems[cartItems.length - 1].getQuantity() ));
   actionInfo.put( "size", cartItems[cartItems.length - 1].getSize() );
   action.log( request,
               cartItems[cartItems.length - 1].getId(),
               null,
               "Shopping Cart Insert",
               actionInfo );
%>
```

Action beans reference:

View some additional information related to Action beans and associated methods.

The `com.ibm.wcp.analysis.beans.Action` bean method signatures are:

Table 363. Descriptions for action bean method signatures

Action bean method signatures	Description
`public void log( HttpServletRequest request, String actionName )`	Logs a non-resource specific action
`public void log( HttpServletRequest request, String resourceId, String collectionName, String actionName )`	Logs a resource specific action
`public void log( HttpServletRequest request, String resourceId, String collectionName, String actionName, String key, String value )`	Logs a resource specific action with key/value action datum
`public void log( HttpServletRequest request, String resourceId, String collectionName, String actionName, Hashtable keyValueData )`	Logs a resource specific action with multiple key/value action data. Each key can have a single value specified by a string object or multiple values specified by a string array

Action beans should be instantiated as session beans. They maintain user action information for the current session including the actions logged by resource and their corresponding log counts.

The following methods are accessible from rules. These methods are provided through a custom application object.

```
public String[] getActionNames(HttpServletRequest request);
public int      getActionCount(HttpServletRequest request, String actionName );
```

Actions can be logged with or without respect to a specific resource. For example, the "OrderCancel" action does not apply to a specific resource whereas the "BrowseContent" action applies to a specific content resource.

If the call to the action log method specifies a `resourceId` and the `collectionName` is `null`, the name of a `ResourceCollection` is inferred. The `ResourceCollection` used will be any one containing a `resource` with the specified `resourceId`. The determination of the `ResourceCollection` used in this scenario is non-deterministic. Note that these variants are suitable for user implementations supporting one and only one `ResourceCollection` per `resource` class. If an implementation utilizes multiple `ResourceCollections` for the same `resource` class, the `collectionName` should be specified.

Category beans:

Content categories enable you to classify Web user interests. You use Category beans to log content categories.

As a Web site visitor navigates your Web pages, a Category bean logs the different categories viewed by the visitor during the session. For example, if a Web site visitor goes to your Web site and views information about the weather, the category "Weather" can be logged. Another Web site visitor might be more interested in sports, so you would log the "Sports" category for this user. Category beans maintain category information for the current session including the categories logged and their corresponding log counts. Category information is routed to all registered log listeners.

All categories are logged to the Feedback schema and to the user session. Category information is not logged to the LikeMinds schema.

You can create rules that access the category counts in the current session, for example:

```
News Interest is
    Weather when
      current Category Count.Weather > 5
    Sports when
      current Category Count.Sports > 5
   Otherwise Headlines
```

You can also create rules that access the category names from the current session, for example:

```
Select content
    whose News.Topics is included in current CategoryNamesCategory Names.Category Names
    order as is
```

"Implementing category logging"
To implement category logging, insert a Category bean into your JSP.

"Category beans reference" on page 2355
Learn about the various method signatures of Category beans.

Implementing category logging:

To implement category logging, insert a Category bean into your JSP.

About this task

Consider the following code for inserting a category bean in your JSP.

```
<jsp:useBean class="com.ibm.wcp.analysis.beans.Category" id="category" scope="session"/>
<% category.log( request, "Sports" ); %>
```

You can also pass an object to a JSP and implicitly log the content category. For example:

```
<jsp:useBean class="com.ibm.wcp.analysis.beans.Category" id="category" scope="session"/>

<%
   NewsManager newsManager = new NewsManager();
   News news = newsManager.findById( request.getParameter( "newsId" ), null );
   category.log( request, news );
%>
```

In the previous example, the category bean logs the news category by querying the `LoggableResource` interface to get the category.

Category beans reference:

Learn about the various method signatures of Category beans.

The `com.ibm.wcp.analysis.beans.Category` bean method signatures are:

Table 364. Descriptions for CustomLog bean method signatures

CustomLog bean method signatures	Description
`public void log( HttpServletRequest request, String category )`	Logs a single category literal.
`public void log( HttpServletRequest request, String[] categories )`	Logs an array of category literals.
`public void log( HttpServletRequest request, LoggableResource resource )`	Logs categories of interest by querying the `LoggableResource` interface of the object.

The `LoggableResource` interface can be implemented in addition to the Resource interface in order to facilitate logging. The categories of you resources can then be stored and retrieved from a database. By using this interface, you avoid the specification of category literals in your JSPs.

```
public interface LoggableResource
{
   String[] getTopics( );
}
```

Each of the topics returned by `getTopics` is logged as a category. If `getTopics` returns null, no category will be logged.

The log methods create a CategoryEvent with the request and category. The CategoryEvent is routed to all of the registered log listeners.

Category beans should be instantiated as session beans. They maintain category information for the current session including the categories logged and their corresponding log counts.

The following methods are accessible through rules. They can also be accessed directly through the category bean.

```
public String[] getCategoryNames( HttpServletRequest request)
public int      getCategoryCount( HttpServletRequest request,
                                  String category )
```

Rating beans:

The Rating bean is a specialized logging bean. You use Rating beans in applications that solicit user ratings for web content or other personalization resources.

The ratings logged through the Rating bean provide the user data necessary for the LikeMinds preference engine. This engine uses explicitly specified user ratings to predict user preferences. You can generate reports based on the ratings of your website content or resources.

Note: The concept of ratings logged through the Rating bean is different from the concept of ratings generated from tagging and rating. Tagging and rating allows users to assign ratings directly to an object to indicate likes and dislikes and is not used to collect data for future recommendations.

All rating data is logged to the Feedback schema and to the LikeMinds schema. Rating data are not logged to the user session.

"Implementing rating logging"
To implement rating logging, insert a Rating bean into your JSP.

"Rating beans reference"
Learn about the various method signatures of Rating beans.

Implementing rating logging:

To implement rating logging, insert a Rating bean into your JSP.

About this task

Following is an example that demonstrates implementation of rating logging by inserting a Rating bean in a JSP.

```
<jsp:useBean class="com.ibm.wcp.analysis.beans.Rating" id="pref" scope="session" />

<%
    // Note:  The mediaId, collectionName, and rating were added to the request
    //          by the referring page.
    pref.log( request,
              request.getParameter( "mediaId" ),
              request.getParameter( "collectionName" ),
              request.getParameter( "rating" ));
%>
```

Note: Web applications implementing preference logging must provide a user interface (UI) to enable preference setting. Once retrieved from the UI, the preference settings can be logged with the Rating bean.

Rating beans reference:

Learn about the various method signatures of Rating beans.

The `com.ibm.wcp.analysis.beans.Rating` bean method signatures are:

Table 365. Descriptions for rating bean method signatures

Rating bean method signatures	Description
`public void log( HttpServletRequest request,` `Resource resource,` `int prefRating );`	Logs a rating for a Resource object. A ResourceCollection with the same resource type will be determined and used for logging the collection name. If you have multiple ResourceCollections containing objects of the same class, you should use the log method that accepts the resourceId and collectionName.
`public void log( HttpServletRequest request,` `String resourceId,` `String collectionName,` `int prefRating );`	Logs a rating using a resource id and collection name.
`public void log( HttpServletRequest request,` `String resourceId,` `String collectionName,` `int prefRating,` `Hashtable ratingData )`	Logs a resource-specific rating with multiple key/value rating data. Each key can have a single value specified by a string object, or multiple values specified by a string array.

The log methods generate a RatingEvent object with the request and rating data. These events are routed to all of the registered log listeners.

Rating beans should be instantiated as session beans; however, they do not maintain rating information for the current session.

CustomLog beans:

CustomLog beans give you the flexibility to log data that does not easily map to the other logging beans. You use CustomLog beans to log application specific data. You define the required key name(s) as well as the value(s) for that key.

Logged data is stored in the Feedback schema in the HITPARMS, KEY_VALUE_COMBO, KEY_VALUE_PAIR, KEY, and VALUE tables. If you use CustomLog beans, you may want to create custom report elements.

All custom log data is logged to the Feedback schema. Custom log data is not logged to the LikeMinds schema nor to the user session.

"Implementing custom logging"
To implement custom logging, insert a CustomLog bean into your JSP.

"CustomLog beans reference" on page 2358
View the CustomLog bean method signatures.

Implementing custom logging:

To implement custom logging, insert a CustomLog bean into your JSP.

About this task

The following code provides an example:

```
<jsp:useBean class="com.ibm.wcp.analysis.beans.CustomLog" id="custom" scope="session" />

<% custom.log( request, "version", "1.0" ); %>
```

You can also log multiple values. For example:

```
<jsp:useBean class="com.ibm.wcp.analysis.beans.CustomLog" id="custom" scope="session" />

<%
   Hashtable customInfo = new Hashtable();
   customInfo.put( "version", "1.0" );
   customInfo.put( "custLevel", new String[] { "gold", "preferred" } );
   customInfo.put( "custRegion", "West" );
   custom.log( request, customInfo );
%>
```

CustomLog beans reference:

View the CustomLog bean method signatures.

The `com.ibm.wcp.analysis.beans.CustomLog` bean method signatures are:

Table 366. Descriptions for CustomLog bean method signatures

CustomLog bean method signatures	Description
`public void log( HttpServletRequest request,` `String key,` `String value );`	Logs a single pair of custom key/value data.
`public void log( HttpServletRequest request,` `Hashtable keyValueData );`	Logs multiple pairs of custom key/value data.

The log methods generate a CustomLogEvent with the request and custom data. The CustomLogEvent is routed to all of the registered log listeners.

CustomLog beans should be instantiated as session beans; however, they do not maintain any session data.

PageView beans:

You use PageView beans to log HTTP request data for JSPs (JavaServer pages).

The HTTP request data logged by PageView beans provides basic Web usage data that can be used during Web traffic analysis. The Action, Category, CustomLog, and Rating beans automatically collect the same basic HTTP data that is collected by the PageView bean. HTTP request data is also collected whenever a rule is executed. Use the PageView bean if the JSP of interest has no other logging beans or content spots, and you want to include the JSP in usage analysis.

"Implementing PageView logging"
To implement Web usage logging, insert a PageView bean in your JSP.

"PageView beans reference" on page 2359
Learn about the `com.ibm.wcp.analysis.beans.PageView` bean log method and instantiation of PageView beans.

Implementing PageView logging:

To implement Web usage logging, insert a PageView bean in your JSP.

About this task

Following is an example of implementing PageView logging:

```
<jsp:useBean class="com.ibm.wcp.analysis.beans.PageView"; id="pageView" scope="session"; />

<%
   pageView.log( request );
%>
```

PageView beans reference:

Learn about the `com.ibm.wcp.analysis.beans.PageView` bean log method and instantiation of PageView beans.

The `com.ibm.wcp.analysis.beans.PageView` bean log method signature is:
`public void log( HttpServletRequest request )`

The log method creates a new `PageViewEvent` with the request. The `PageViewEvent` is routed to all registered log listeners.

PageView beans should be instantiated as session beans; however, they will not maintain any session data.

LogManager:

When data is logged by either logging beans or rules, log events are generated and are routed to a controller for processing. LogManager is the class that implements this controller. There is a single instance of the LogManager within the Personalization run-time. It is responsible for receiving all logged events and distributing these events to listener objects that implement the LogListener interface and are registered with the LogManager.

The LogManager queues log events as they are received. In order to protect listeners from performance degradation due to inefficiencies and/or problems in other listeners, a separate notification thread is managed for each active listener. A nanny thread monitors the notification threads and suspends any non-responsive listener. Since the "queue" is processed by multiple consumers, an event is removed from the queue after all of the notification threads have read the event. You can alter the settings that govern the non-responsive time interval and event queue size by modifying the `FeedbackService.properties` file, located in the *wp_profile_root*`/PortalServer/config/config/services`.

Listeners and persistence:

Log listeners process the log events originating from either logging beans or rules.

Log listeners process the log events originating from either logging beans or rules and will typically store the data in some fashion (such as a database). Separate LogListener implementations are provided for processing log data into the Personalization Feedback schema and for processing data into the LikeMinds schema. These listeners always run whenever logging is enabled. Additionally, it is possible to extend the provided listeners and also write custom listeners from scratch to process logging events and log data as you see fit.

A log listener is registered with and receives log events from the LogManager. All log listeners must either implement the LogManager interface or extend the LogAdapter class. The LogAdapter class provides a default (empty) implementation of each of the handleEvent methods in the LogListener interface.

The logging component uses cache and multi-threading to optimize performance, and to allow each listener to process events at a difference rate without affecting other listeners.

The log listener interface

```
public interface com.ibm.wcp.analysis.event.LogListener extends java.util.EventListener
{
   /**
    * Method called by LogManager after adding the listener to the
    * set of active listeners.  This method can be used to perform
    * initialization during startup.
    */

   public void startHandlingEvents( );

   /**
    * Method called by LogManager when the listener is removed
    * from the set of active listeners.  This method can be
    * used to perform cleanup during termination processing.
    */

   public void stopHandlingEvents( );

   /**
    * Method to handle rule trigger events.
    */

   void handleEvent( RuleEvent event );

   /**
    * Method to handle category logging events.
    */

  void handleEvent( CategoryEvent event );

   /**
    * Method to handle action logging events.
    */

  void handleEvent( ActionEvent event );

   /**
    * Method to handle explicit rating events.
    */

   void handleEvent( RatingEvent event );

  /**
    * Method to handle custom logging events.
    */

 void handleEvent( CustomLogEvent event );

   /**
    * Method to handle basic Web traffic events.
    */
   public void handleEvent( PageViewEvent  event );

  /**
    * Method to handle Personalization audit events.
    */

  void handleEvent( AuditEvent event );
}
```

"FeedbackListener"
The FeedbackListener class routes data to the Feedback schema.

"LMListener"
The LMListener routes data of interest to the LikeMinds database layer.

"Custom log listeners"
When the log method of a logging bean is called, the feedback facility generates a log event. A log event is also generated when a rule is executed. Log listeners process these log events and either store the event data or perform custom processing with these events.

FeedbackListener:

The FeedbackListener class routes data to the Feedback schema.

The FeedbackListener processes all log event types. Each page request (JSP invocation) maps to one row in the Feedback schema HIT_FACTS table. Data for each log event is stored in a separate entry in the Feedback schema HITPARMS table. This distinguishes the data related to each rule or log method call and enables the reporting of information specific to individual content spots or method call. This storage design correlates of all of the data logged by the rules or method calls in a JSP enabling page level reporting.

An instance of the FeedbackListener is active on all run-time servers whenever logging is enabled.

Related concepts:

"Feedback database schema" on page 2372
The Feedback schema, within the Feedback database, stores data about your Web site visitors' actions. This schema is closely aligned with the Tivoli Site Analyzer schema.

LMListener:

The LMListener routes data of interest to the LikeMinds database layer.

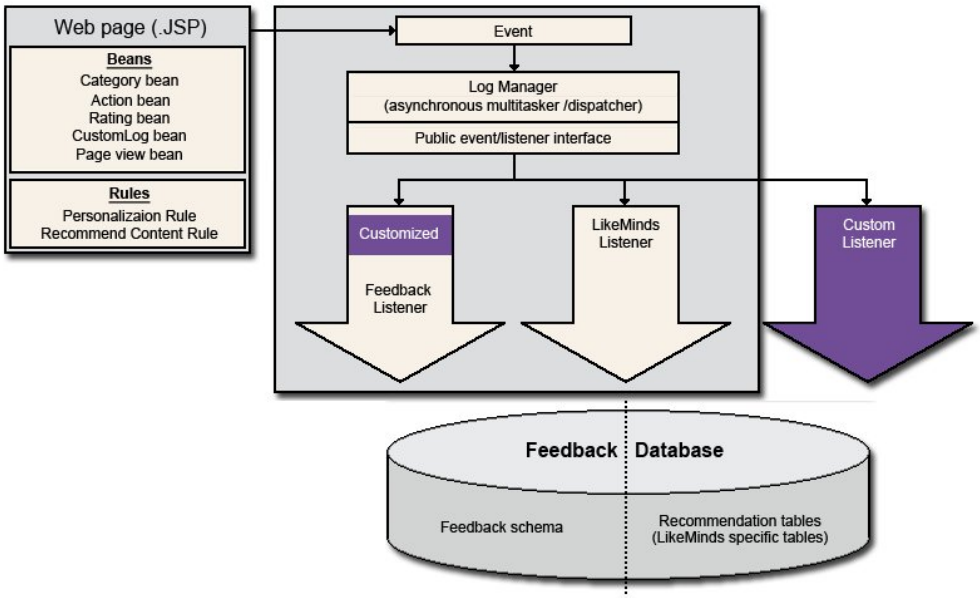
The LMListener only processes ActionEvent and RatingEvent data. These events result in LikeMinds addTransaction calls. Only action names defined in the LikeMinds lpx_trx_type table are logged by this listener. All other actions are ignored.

An instance of the LMListener is active after LikeMinds configuration tasks are run.

Custom log listeners:

When the log method of a logging bean is called, the feedback facility generates a log event. A log event is also generated when a rule is executed. Log listeners process these log events and either store the event data or perform custom processing with these events.

The Feedback component provides two default log listeners, the LMListener that collects data for use by the LikeMinds recommendation engine and the FeedbackListener that collects data for use in the feedback reports. Custom log listeners can be used to modify the default behavior of the FeedbackListener or to implement a listener that provides user-specific behavior.



There are a number of reasons that you might want to provide a customized feedback listener. Some of these are:

- To collect request parameters, referral parameters, or cookies. By default, the feedback listener does not collect this data. You must implement a customized feedback listener that enables the collection of this data.
- To prevent private information from being stored. For instance, you could mask the userid or IP address to ensure the privacy of your users.
- To augment the event data. Suppose your event data contains a product id number and you would like to report on products by product name and id. You could perform a custom lookup during event processing and add the product name to the event.

There are a number of reasons that you might want to provide a new custom listener class. Some of these are:

- To store event data in a separate database. Suppose that you want to capture rating data in a preference database. You can do this through your own web application or you could do this as by adding a custom listener and capturing rating events. The custom listener could facilitate real time rating results.
- To generate email when an event occurs. Perhaps you want to send email to customers after they purchase a large order. By processing an action event with the purchase amount included as action data, you could do this with a custom log listener.
- To generate notifications when an event occurs. You can detect the frequency of shopping cart abandons and generate a notification to the site administrator to check site availability and performance.

"Custom listener classes" on page 2363
A custom listener class is a class that implements the LogListener interface. View the steps to implement a custom listener class.

"Customized feedback listeners" on page 2364
A customized feedback listener is a subclassed FeedbackListener object that is registered with the feedback LogManager. View the steps to implement a custom feedback listener.

Custom listener classes:

A custom listener class is a class that implements the LogListener interface. View the steps to implement a custom listener class.

The custom listener class can implement the LogListener interface explicitly and provide implementations for all of the LogListener methods. Alternatively, the custom listener class can extend the LogAdaptor class. Since the LogAdaptor class contains default implementations of all of the LogListener methods, LogAdaptor subclasses only need to implement the LogListener methods of interest. To implement a custom listener class:

1. Implement a class that extends `com.ibm.wcp.analysis.event.LogAdaptor`. Override the handleEvent methods that accept the event type of interest. You can also provide an implementation for startHandlingEvents and stopHandlingEvents if your listener needs to perform initialization or cleanup, respectively.
2. Install the class file for your custom listener in the classpath of the server where the Personalization run-time is installed.
3. Add the class name to the logListeners property in the `FeedbackService.properties` file, located in the *wp_profile_root*`/PortalServer/config/config/services` directory.
4. Restart the Personalization run-time server.

Note that custom listener classes (other than customized feedback listeners) are always enabled when the Personalization run-time enterprise application is running.

The following example illustrates the implementation of a custom listener class. The listener in this example will generate an alert whenever the "MegaPurchase" action is logged.

```
import com.ibm.wcp.analysis.event.*;

public class SimpleCustomListener extends LogAdapter
{
   /**
    * Method to handle action events.
    */
   public void handleEvent( ActionEvent event )
   {
      if (event.getActionName().equals( "MegaPurchase" ))
         generateAlert();
   }

   /**
    * Method to generate an alert.
    */
   private void generateAlert( )
   {
      // Your custom code for generating an alert can go here.
      System.out.println( "We have a big purchase!" );
   }
}
```

Customized feedback listeners:

A customized feedback listener is a subclassed FeedbackListener object that is registered with the feedback LogManager. View the steps to implement a custom feedback listener.

Perform the following steps to implement a custom feedback listener:

1. Implement a class that extends com.ibm.wcp.analysis.event.FeedbackListener. Override the handleEvent methods that accept the event type of interest.
2. Install the class file for your custom feedback listener in the classpath of the server where the Personalization run-time is installed.
3. Add the class name to the **logListeners** property in the `FeedbackService.properties` file, located in the *wp_profile_root*`/PortalServer/config/config/services` directory.
4. Ensure that the **LoggingEnabled** is set to `true`.
5. Restart the Personalization run-time server.

Note that whenever logging is enabled, your customized FeedbackListener is enabled. Conversely, your customized FeedbackListener is disabled whenever logging is disabled. Only the default FeedbackListener or one custom feedback listener can be active at a time. The specification of a custom feedback listener will override the registration of the default FeedbackListener with the LogManager.

The following example illustrates the implementation of a customized feedback listener. The listener in this example will collect cookie data, request query data, and referral query data. Note that this data is not collected by the default FeedbackListener, but can be set in the event data with a customized feedback listener.

```
import com.ibm.wcp.analysis.event.*;

public class ParmCookieListener extends FeedbackListener
{
   /**
    * Method to handle action events.
    */
   public void handleEvent( ActionEvent event )
   {
      enableParmCollection( event );
      super.handleEvent( event );
   }

   /**
    * Method to handle category events.
    */
   public void handleEvent( CategoryEvent event )
   {
      enableParmCollection( event );
      super.handleEvent( event );
   }

   /**
    * Method to handle rule events.
    */
   public void handleEvent( RuleEvent event )
   {
      enableParmCollection( event );
      super.handleEvent( event );
   }

   /**
```

```
     * Method to set switches to collect cookie and
     * query string data.
     */
    private void enableParmCollection( LogEvent event )
    {
       event.enableLogCookies();
       event.enableLogQueryParms();
       event.enableLogReferralParms();
    }
}
```

Classes and APIs for writing custom listeners:

In order to write custom listeners, it is necessary to familiarize yourself with the classes representing logging events and info that is contained in those events. A reference of these classes is provided here.

"LogEvent class"
The LogEvent class is the base class for all run-time logging events. The methods in this class are used to access all basic http request information.

"ResourceInfo class" on page 2371
The ResourceInfo class is a wrapper class for a resource id and collection name tuple.

"RuleInfo class" on page 2372
The RuleInfo class is a wrapper class for a rule name and campaign name tuple.

LogEvent class:

The LogEvent class is the base class for all run-time logging events. The methods in this class are used to access all basic http request information.

All run-time log data is accessible through the log event sub classes. These log events are constructed by either a rule trigger or a logging bean. They are then routed to the registered log listeners.

The direct subclasses of the LogEvent class are ActionEvent, CategoryEvent, CustomLogEvent, PageViewEvent, RatingEvent, and RuleEvent.

```
public class com.ibm.wcp.analysis.event.LogEvent extends com.ibm.wcp.analysis.event.Event
                                        implements Serializable
```

Table 367. Method summary

Method	Explanation
`protected LogEvent( HttpServletRequest request )`	Constructor
`public Cookie[] getCookies( )`	Returns the cookies available with the request for this event.
`public void setCookies( Cookie[] )`	Sets the cookies for this event. Can be used by custom listeners in order to replace the cookie data received with the current JSP request.
`public void enableLogCookies( boolean enable )`	If enable is true, enables the collection of cookies for this event instance; otherwise, disables the collection of cookies for this event instance. If you want to collect cookie information with the FeedbackListener or a subclassed FeedbackListener, use this method.
`public void enableLogCookies( )`	Same as `enableLogCookies( true )`.

Table 367. Method summary (continued)

Method	Explanation
`public boolean logCookies( )`	Returns true if cookie information should be logged to the Feedback schema for this event; otherwise, returns false.
`public String getIPAddress( )`	Returns the Web client IP address.
`public void setIPAddress( String ipAddress )`	Sets the IP address for this event. Can be used by custom listeners in order to replace the IP address for the current JSP request.
`public String getMethod( )`	Returns the HTTP method used in the current JSP request (GET, POST). Note that the method is not stored in the Feedback schema.
`public void setMethod( String method )`	Sets the HTTP method for the current JSP request. Can be used by custom listeners in order to replace the method from the current JSP request. Note that the method is not stored in the Feedback schema.
`public String getProtocol( )`	Returns the protocol for the current JSP request (http or https).
`public void setProtocol( String protocol )`	Sets the protocol for the current JSP request. Can be used by custom listeners in order to replace the protocol from the current JSP request.
`public void enableLogReferralParms( boolean enable )`	If enable is true, enables the collection of referral parameters for this event instance; otherwise, disables the collection of referral parameters for this event instance. If you want to collect the referral query string parameters with the FeedbackListener or a subclassed FeedbackListener, use this method.
`public void enableLogReferralParms( )`	Same as `enableLogReferralParms( true )`.
`public boolean logReferralParms( )`	Returns true if referral parameter information should be logged to the Feedback schema for this event; otherwise, return false.
`public String getRemoteHost( )`	Returns the host name of the client machine that issued this JSP request. Note that the remote host name is not stored in the Feedback schema.
`public void setRemoteHost( String hostName )`	Sets the host name of the client machine that issued this JSP request. Can be used by custom listeners that perform DNS resolution. Note that the remote host name is not stored in the Feedback schema.
`public String getQueryString( )`	Returns the query string parameters for this event.
`public void setQueryString( String queryString )`	Sets the query string parameters for this event. Can be used by custom listeners in order to replace the query string parameters for the current JSP request.
`public void enableLogQueryParms( boolean enable )`	If enable is true, enables the collection of query parameters for this event instance; otherwise, disables the collection of referral parameters for this event instance. If you want to collect the query string parameters with the FeedbackListener or a subclassed FeedbackListener, use this method.
`public void enableLogQueryParms( )`	Same as `enableLogQueryParms( true )`.

Table 367. Method summary (continued)

Method	Explanation
`public boolean logQueryParms( )`	Returns true if query parameter information should be logged to the Feedback schema for this event; otherwise, return false.
`public String getReferrer( )`	Returns the URL of the referral page or null if there was no referrer.
`public void setReferrer( String referrer )`	Sets the referral page for this event. Can be used by custom listeners in order to replace the referrer for the current JSP request.
`public String getServerName( )`	Returns the host name of the server machine that is processing this JSP request. Note that the server host name is not stored in the Feedback schema.
`public void setServerName( String serverName )`	Sets the host name of the server machine that is processing this JSP request. Can be used by custom listeners to set the name of the current server. Note that the server host name is not stored in the Feedback schema.
`public String getSessionId( )`	Returns the id of the session for the current JSP request.
`public void setSessionId( String sessionId )`	Sets the id of the current user session. Can be used by custom listeners in order to replace the id of the current HttpSession object. This can be useful when an alternative session identification mechanism is used.
`public Date getTimestamp( )`	Returns the time that this log event was generated. Note that this timestamp differs slightly from the time the event was received by the IBM WebSphere Application Server. Note that if there are multiple logging beans or content spots in a JSP, the timestamps in the generated log events will be the same.
`public void setTimestamp( Date timestamp )`	Sets the time for this log event. Can be used by custom listeners in order to replace the timestamp used for this event.
`public String getUrl( )`	Returns the URL of the page request encapsulated by this log event.
`public void setUrl( String url )`	Sets the URL of the page request for this event. Can be used by custom listeners in order to replace the URL for the current JSP request.
`public String getUser( )`	Returns the HTTP authenticated user for the current session.
`public void setUser( String user )`	Sets the user for this event. Can be used by custom listeners in order to replace the user for the current JSP request. This can be useful when an alternative authentication mechanism is used.
`public String getUserAgent( )`	Returns the browser engine for the current session.
`public void setUserAgent( String userAgent )`	Sets the user agent for this event. Can be used by custom listeners in order to replace the user agent from the current JSP request.
`public String toString( )`	Returns a String representation of this event.

"RuleEvent class" on page 2368
A RuleEvent class is constructed whenever a rule is executed. It contains

information about the rule that was executed and the resulting resources. It is an implicitly constructed event; a logging bean is not necessary.

"CategoryEvent class" on page 2369
The CategoryEvent class is used to access the data logged with a Category bean.

"ActionEvent class" on page 2369
The ActionEvent class is used to access the data logged with an Action bean.

"CustomLogEvent class" on page 2370
Get an overview of the CustomLogEvent class and its methods.

"RatingEvent class" on page 2370
The RatingEvent class is used to access the data logged with a Rating bean.

"PageViewEvent class" on page 2371
The PageViewEvent class is used to access data logged by a PageView bean.

RuleEvent class:

A RuleEvent class is constructed whenever a rule is executed. It contains information about the rule that was executed and the resulting resources. It is an implicitly constructed event; a logging bean is not necessary.

```
public class com.ibm.wcp.analysis.event.RuleEvent extends com.ibm.wcp.analysis.event.LogEvent
                                                   implements Serializable
```

Get an overview of the methods of the RuleEvent class.

Table 368. Method summary

Method	Explanation
`public RuleEvent( HttpServletRequest request, RuleInfo ruleInfo, ResourceInfo[] resourceInfo )`	Constructor.
`public RuleInfo getRuleInfo( )`	Returns a rule information object containing the campaign and rule name for the rule that was executed in the content spot.
`public void setRuleInfo( RuleInfo ruleInfo )`	Sets the rule information for this event. Can be used by custom listeners in order to replace the rule execution data.
`public ResourceInfo[] getResourceInfo( )`	Returns a resource information array containing the results of the rule for this event. The resource information contains the collection name and resource ids of the results.
`public void setResourceInfo( ResourceInfo[] resourceInfo )`	Sets the resource information for this event. Can be used by custom listeners in order to replace the resource information.
`public String getResourceClass( )`	Returns the class name of the resources returned by the executed rule.
`public void setResourceClass( String className )`	Sets the class name of the resources returned by the executed rule. Can be used by custom listeners in order to replace the class name.
`public String toString( )`	Returns a String representation of this event.

CategoryEvent class:

The CategoryEvent class is used to access the data logged with a Category bean.

```
public class com.ibm.wcp.analysis.event.CategoryEvent extends com.ibm.wcp.analysis.event.LogEvent
                                                    implements Serializable
```

Get an overview of the methods of the CategoryEvent class.

Table 369. Method summary

Method	Explanation
`public CategoryEvent( HttpServletRequest request, String[] topics )`	Constructor.
`public String[] getTopics( )`	Returns the array of topics for this category event.
`public void setTopics( String[] topics )`	Sets the topics for this event. Can be used by custom listeners in order to replace the topics for this event.
`String toString( )`	Returns a String representation of this event.

ActionEvent class:

The ActionEvent class is used to access the data logged with an Action bean.

```
public class com.ibm.wcp.analysis.event.ActionEvent extends com.ibm.wcp.analysis.event.LogEvent
                                                  implements Serializable
```

Get an overview of the methods of the ActionEvent class.

Table 370. Method summary

Method	Explanation
`public ActionEvent( HttpServletRequest request, ResourceInfo resourceInfo, String actionName, Hashtable actionData )`	Constructor.
`public ResourceInfo getResourceInfo( )`	Returns the resource upon which the action in this event was taken.
`public void setResourceInfo( ResourceInfo resourceInfo )`	Sets the resource info for this action event. Can be used by custom listeners to replace the resource information in this event.
`public Hashtable getActionData( )`	Returns the supplemental data associated with this action. The action data is in key value format. Since a single key can have multiple values, the action data values are stored in the hashtable as String arrays.
`public void setActionData( Hashtable actionData )`	Sets the action data for this action event. Can be used by custom listeners to replace the action data in this event.
`public String getActionName( )`	Returns the name given to the current action.
`public void setActionName( String actionName )`	Sets the action name for this event. Can be used by custom listeners to replace the action name in this event.
`public RuleInfo getRuleInfo( )`	Returns the rule associated with this action. The subject rules for this action are determined by all rules in the current session that returned the target resource.

Table 370. Method summary (continued)

Method	Explanation
`public void setRuleInfo( RuleInfo ruleInfo )`	Sets the rule information for this action event. Can be used by custom listeners to replace the rule information in this event.
`public String toString( )`	Returns a String representation of this event.

CustomLogEvent class:

Get an overview of the CustomLogEvent class and its methods.

The CustomLogEvent class is used to access the data logged with a CustomLog bean.

```
public class com.ibm.wcp.analysis.event.CustomLogEvent extends com.ibm.wcp.analysis.event.LogEvent
                                                    implements Serializable
```

Table 371. Method summary

Method	Explanation
`public CustomLogEvent( HttpServletRequest request,` `Hashtable customData )`	Constructor.
`public Hashtable getCustomData( )`	Returns the custom data as key value information. Since a single custom data key can have multiple values, the custom data values are stored in the hashtable as String arrays.
`public void setCustomData( Hashtable customData )`	Sets the custom data for this event. Can be used by custom listeners to replace the data in this custom data event.
`public String toString( )`	Returns a String representation of this event.

RatingEvent class:

The RatingEvent class is used to access the data logged with a Rating bean.

```
public class com.ibm.wcp.analysis.event.RatingEvent extends com.ibm.wcp.analysis.event.LogEvent
                                                  implements Serializable
```

Get an overview of the methods of the RatingEvent class.

Table 372. Method summary

Method	Explanation
`public RatingEvent( HttpServletRequest request,` `ResourceInfo resourceInfo,` `int rating )`	Constructor.
`public ResourceInfo getResourceInfo( )`	Returns the resource that is the target of the current rating. The resource object returned contains the collection name and resource id.
`public void setResourceInfo( ResourceInfo resourceInfo )`	Sets the resource information for this rating event. Can be used by custom listeners to replace the resource information for this rating.

Table 372. Method summary (continued)

Method	Explanation
`public int getRating( )`	Returns the rating as an integer value. The value can be any valid integer as defined by the Web application implementor.
`public void setRating( int rating )`	Sets the rating for this event. Can be used by custom listeners to replace the rating in this event.
`public String toString( )`	Returns a String representation of this event.

PageViewEvent class:

The PageViewEvent class is used to access data logged by a PageView bean.

```
public class com.ibm.wcp.analysis.event.PageViewEvent extends com.ibm.wcp.analysis.event.LogEvent
                                                      implements Serializable
```

Get an overview of the methods of the PageViewEvent class.

Table 373. Method summary

Method	Explanation
`public PageViewEvent( HttpServletRequest request )`	Constructor.
`public String toString( )`	Returns a String representation of this event.

ResourceInfo class:

The ResourceInfo class is a wrapper class for a resource id and collection name tuple.

```
public class com.ibm.wcp.analysis.event.ResourceInfo extends Object
                                                     implements Serializable
```

Get an overview of the methods of the ResourceInfo class.

Table 374. Method summary

Method	Explanation
`public ResourceInfo( String resourceId, String collectionName )`	Constructor.
`public ResourceInfo( )`	Constructor.
`public String getResourceId( )`	Returns the id of the resource.
`public void setResourceId( String resourceId )`	Sets the id of the resource.
`public String getCollectionName( )`	Returns the name of the collection that contains this resource.
`public void setCollectionName( String collectionName )`	Sets the name of the collection that contains this resource.
`public String toString( )`	Returns a String representation of this resource tuple.
`public boolean equals( ResourceInfo resourceInfo )`	Returns true if and only if (1) the objects are the same or (2) the resource id and collection name information is equal.

Table 374. Method summary (continued)

Method	Explanation
`public int hashCode( )`	Returns a hash code enabling objects of this type to be used as hash keys.

RuleInfo class:

The RuleInfo class is a wrapper class for a rule name and campaign name tuple.

```
public class com.ibm.wcp.analysis.event.RuleInfo extends Object
                                                 implements Serializable
```

Get an overview of the methods of the RuleInfo class.

Table 375. Method summary

Method	Explanation
`public RuleInfo( String rule, String collectionName )`	Constructor.
`public RuleInfo( )`	Constructor.
`public String getRule( )`	Returns the name of the rule.
`public void setRule( String rule )`	Sets the name of the rule.
`public String getCampaign( )`	Returns the name of the campaign containing this rule.
`public void setCampaign( String campaign )`	Sets the name of the campaign containing this rule.
`public String toString( )`	Returns a String representation of this rule tuple.
`public boolean equals( RuleInfo ruleInfo )`	Returns true if and only if (1) the objects are the same or (2) if the rule names are equal and the campaign names are equal.

Reports:

Once data has been logged to the Feedback database schema, it is possible to use any reporting tool that can connect to a standard SQL database to generate reports based on this data. Reference the Feedback database schema section of the Information Center when writing such reports.

Related concepts:

"Feedback database schema"
The Feedback schema, within the Feedback database, stores data about your Web site visitors' actions. This schema is closely aligned with the Tivoli Site Analyzer schema.

Feedback database schema:

The Feedback schema, within the Feedback database, stores data about your Web site visitors' actions. This schema is closely aligned with the Tivoli Site Analyzer schema.

Any application that can connect to a standard SQL database can make use of the data that has been logged to the Feedback database schema.

For instance, you can use the data in the Feedback database schema to:

- Generate reports using reporting software.
- Integrate OnLine Analytical Processing (OLAP) tools.
- Create your own data integration tools.

"Feedback schema tables"
Learn about the feedback schema tables and the fields or columns they contain.

"Key value pairs" on page 2384
The additional information passed by rule logging events and certain bean logging events is logged to the Feedback database in the form of key value pairs.

Feedback schema tables:

Learn about the feedback schema tables and the fields or columns they contain.

The following tables are used by the Feedback schema:

Table Name: Browsers

Table 376. Table description: This table contains the client Web browser defined internally by a User Agent mapping.

Column Name	Column description
Browser_ID	The primary key. A unique number automatically generated by the database.
Name	The browser name defined internally by a User Agent mapping.

Table Name: Calendar

Table 377. Table description: Initially, this table is empty. The Feedback component calculates date stamp IDs, assuming a start date of 1/1/1995. The table can be populated with a new configuration task for applications that use custom logging or reports that reference this table.

Column Name	Column description
Date_ID	The primary key. A unique identifier automatically generated by the database.
Month	Month expressed as integer 1-12
Day	Day expressed as integer 1-31
Year	Year expressed as integer 1995-2030
Quarter	Quarter expressed as integer 1-4
Weekday	Weekday expressed as integer 1-7
WeekOfYear	The week number expressed as an integer 1-52
EpochDay	The number of days since the beginning date in this table.
EpochWeek	The number of weeks since the beginning date in this table.

Table 377. Table description: Initially, this table is empty. The Feedback component calculates date stamp IDs, assuming a start date of 1/1/1995. The table can be populated with a new configuration task for applications that use custom logging or reports that reference this table. (continued)

Column Name	Column description
EpochMonth	The number of months since the beginning date in this table.
EpochQuarter	The number of quarters since the beginning date in this table.
AggrWeekID	The identifier of the first day of the week that this day belongs to.
AggrMonthID	The identifier of the first day of the month that this day belongs to.
AggrQuarterID	The identifier of the first day of the quarter that this day belongs to.
AggrYearID	The identifier of the first day of the year that this day belongs to.

Table Name: Domain

Table description: This table is RESERVED for future Feedback component use.

Column Name	Column Description
Domain_ID	The primary key; a unique identifier automatically generated by the database.
Name	The domain name as resolved from the TCP/IP address.

Table Name: Entities

Table description: This table is RESERVED for Feedback component use.

Column Name	Column Description
Entity_ID	The primary key; a unique identifier automatically generated by the database.
Name	The name of the entity.
Type	Indicates the type of entity: 0=entity 1=category 2=key/value pair
ParentEntity_ID	For a category entity, indicates the entity the category is based on.

Table Name: EntityTraversal

Table description: This table is RESERVED for IBM Feedback component use.

Column Name	Column Description
EntityTraversal_ID	The primary key. A unique identifier automatically generated by the database.
Entity_ID	The identifier of the entity we start from.

Column Name	Column Description
Table1	The table this entity is stored in.
Column1	ID column for this entity.
Table2	The table of the parent entity.
Column2	Join column between entity and parent entity table.
PickThis	RESERVED for IBM Feedback component use.

Table Name: Hit_Facts

Table description: This table contains the information processed during data collection. Each row represents a hit to a page of your Web site--if that page contains a rule or logging bean. Not all fields are used on each entry.

Note: The first date stamp (date ID=1) corresponds to 1/1/1995. Time stamp IDs are calculated such that each second of the day has a separate ID.

Column Name	Column Description
JS_ID	A foreign key to the JavaScriptStatus table.
CookiesStatus_ID	A foreign key to the CookiesStatus table.
Hits	The actual number of hits.
Status_ID	A foreign key to the ResetStatus table.
ReturnCode_ID	A foreign key to the ReturnCodes table.
HTTPVersion_ID	A foreign key to the HTTPVersion table.
RecordType	Indicates what type of log file this hit came from. This field is for Feedback component use only.
LastUpdated	Time stamp of when this row was last created / updated.
PageViews	The actual number of page views. If the resource for this hit is a page, this field is set to 1.
TimeTaken	Time taken by the Web server to process this hit. This column is not implemented and is RESERVED for future Feedback component use.
Bytes	The number of bytes sent by the Web server to the client browser for this hit.
CorrelationKey	The key used to correlate information between the various types of logs that can be processed during data collection.
Session_ID	The identifier of the session this hit belongs to. A foreign key to the Session_Facts table.
HitTimestmp	Time stamp of this hit expressed as a Java timeInMillis value.

Column Name	Column Description
LocalDate_ID	The date of this hit. The first date stamp (date ID=1) corresponds to 1/1/1995. See also, Populating CALENDAR and TIMEOFDAY tables.
ImportHistory_ID	Not implemented. This field is RESERVED for future Feedback component use.
Hit_ID	The primary key. A unique identifier automatically generated by the database.
RefProtocol_ID	Protocol that the referrer used to request this hit. A foreign key to the Protocols table.
Referrer_ID	The referrer for this hit. A foreign key to the Referrer table.
Protocol_ID	The protocol the Web server used to process this hit. A foreign key to the Protocols table.
LocalTimeOfDay_ID	The time of this hit. See also, Populating CALENDAR and TIMEOFDAY tables.
GMTDate_ID	The GMT date of this hit. See also, Populating CALENDAR and TIMEOFDAY tables.
Resource_ID	The resource requested by the client browser. A foreign key to the Resources table.
GMTTimeOfDay_ID	The GMT date of this hit. See also, Populating CALENDAR and TIMEOFDAY tables.
Resource_ID	The resource requested by the client browser. A foreign key to the Resources table.
GMTTimeOfDay_ID	The GMT time of this hit. See also, Populating CALENDAR and TIMEOFDAY tables.

Table Name: HitParms

Table description: This table lists the personalization data (such as rules, campaigns, ratings), as well as query strings associated with a given hit.

Column Name	Column Description
Hit_ID	The identifier of the Hit that contains the query string. This is a foreign key to the ID column of the Hit_facts table.
Parms_ID	The identifier for a specific query string. This is a foreign key to the ID column of the Parms table.
Ordering	Ordering is used to group a set of keys during the processing of a Web page.
ParmType	Identifies the type of key value data. For example, personalization rules data, query strings, or referral query status.

Table Name: HTTPVersion

Table description: This table lists all known HTTP versions. It is pre-populated by the Feedback component when the table is created.

Column Name	Column Description
HTTPVersion_ID	1 - 'HTTP 1.0' 2 - 'HTTP 1.1' 99 - 'Unknown'
Name	The name of the HTTP version.

Table Name: ImportHistory

Table description: This table is RESERVED for future Feedback component use

Column Name	Column Description
ImportHistory_ID	The primary key. A unique number automatically generated by the database.
BeginTimeStamp	Start time of the import, expressed as a JAVA timeInMillis value.
EndTimeStamp	Time the import finished, expressed as a JAVA timeInMillis value.
Status	The status of the import.

Table Name: Key

Table description: Lists all Keys (part of Key/Value pairs) encountered during data collection.

Column Name	Column Description
Key_ID	The primary key. A unique identifier automatically generated by the database.
Name	the name of the key.

Table Name: Key_Value_combo

Table description: This table lists all Key/Value pairs that are grouped together.

Column Name	Column Description
Parms_ID	A foreign key to the Parms table.
KeyValuePair_ID	A foreign key to the Key_Value_Pair table.

Table Name: Key_Value_Pair

Table description: This table lists all Key/Value pairs processed during data collection.

Column Name	Column Description
KeyValuePair_ID	The primary key. A unique identifier generated automatically by the database.
Key_ID	A foreign key to the Key table.

Column Name	Column Description
Value_ID	A foreign key to the Value table.

Table Name: Networks

Table description: This table contains all the TCP/IP addresses processed during data collection.

Column Name	Column Description
Network_ID	The primary key; a unique number automatically generated by the database.
IP_Address	TCP/IP address as obtained from the page hit.
Subdomain_ID	The TCP/IP subdomain this address belongs to. A foreign key to the ID column of the Subdomains table.
Domain_ID	The TCP/IP domain this address belongs to. A foreign key to the ID column of the Domain table.

Table Name: Parms

Table description: This table lists all the key/value pairs strings processed during data collection.

Column Name	Column Description
KVCount	Indicates how many key/value pairs are in this query string.
ParmsString	This field is not used.
WebNode_ID	This field is not used.
Parms_ID	A unique identifier automatically generated by the database.

Table Name: Platforms

Table description: This table contains the client operating system defined internally by a User Agent mapping.

Column Name	Column Description
Platform_ID	The primary key. A unique number automatically generated by the database.
Name	The Operating System name as defined via the User Agent global settings.

Table Name: Protocols

Table description: Lists all network protocols processed during data collection.

Column Name	Column Description
Protocol_ID	The primary key. A unique identifier generated automatically by the database.
Name	The name of the network protocol.

Table Name: Referrer

Table description: This table contains all the referrers processed during data collection.

Column Name	Column Description
Referrer_ID	The primary key. A unique number automatically generated by the database.
RefHost_ID	A foreign key to the ID column of the REFERRERHOST table.
ReferrerURL_ID	A foreign key to the ID column of the REFERRERURL table.

Table Name: ReferrerHost

Table description: This table stores all the referrer site host names processed during data collection.

Column Name	Column Description
RefHost_ID	The primary key. A unique number automatically generated by the database.
IsLocal	Indicates whether this referrer is internal or external. 0 - external referrer 1 - internal referrer
Name	Host name part of referrer as parsed from the Web log record.

Table Name: ReferrerURL

Table description: This table contains the referrer pages processed during data collection. Note that the referrer site's host name is stored in the ReferrerHost table.

Column Name	Column Description
ReferrerURL_ID	The primary key. A unique number automatically generated by the database.
Name	The name of the referrer page as parsed from the Web log record.

Table Name: Resources

Table description: This table contains all resources (URLs) processed during data collection.

Column Name	Column Description
Resource_ID	The primary key. A unique number automatically generated by the database. IsPage Indicates whether this resource is a page. 0 = false 1 = true
IsImage	Indicates whether this resource is an image. 0=false 1=true
IsDirectory	Indicates whether this resources is a file directory. This column is not currently implemented and is RESERVED for future Feedback component use.
Name	The resource string as parsed from the Web log record.

Table Name: Session_Facts

Table description: This table contains all the visits calculated from the Web log data. Note: The first date stamp (date ID=1) corresponds to 1/1/1995. Time stamp IDs are calculated such that each second of the day has a separate ID.

Column Name	Column Description
firstHitTimestmp	Timestamp of the first hit in session expressed as a JAVA timeInMillis value.
lastHitTimestmp	Timestamp of the last hit in the session expressed as a JAVA timeInMillis value.
Result_ID	Foreign key to the ID column of the RESULT table. For your use in classifying this session.
User_ID	The visitor to your site. A foreign key to the Users table.
Referrer_ID	The referrer for this session. A foreign key to the Referrer table.
Sessions	The actual number of sessions. This field will always contain the value. It is used during Report Database processing.
LastUpdated	Timestamp of when this row was last created/updated.
PageViews	Number of page views that are part of this session.
Duration	Duration of the session up to the last hit. The Feedback component cannot calculate the full duration of the session because there is no explicit end-of-session signal from the Web log.
Hits	Number of hits that are part of this session.

Column Name	Column Description
SessionIdentifier	The session identifier string from the Web log. This is sometimes referred to as a session cookie. The SessionIdentifier contains information about the user's id and ip address. If this information needs to be protected steps must be taken to enable encryption on the database level for this column.
SessionTimestmp	Beginning of this session expressed as a JAVA timeInMillis value.
LocalDate_ID	This is the starting date of the session in the local time zone of the Web server that handled this session. See also, Populating CALENDAR and TIMEOFDAY tables.
LocalTimeOfDay_ID	This is the start time of the session in the local time zone of the Web server which handled this session. See also, Populating CALENDAR and TIMEOFDAY tables.
ImportHistory_ID	This column is not used.
Session_ID	The primary key; a unique number automatically generated by the database.
UserAgent_ID	The signature of the Web browser used for this session.
EntryResource_ID	The first resource viewed during this session.
Network_ID	Pointer to the Networks table.
GMTDate_ID	This is the start date of the session translated to the GMT time zone. See also, Populating CALENDAR and TIMEOFDAY tables.
ExitResource_ID	The last resource viewed during this session.
GMTTimeOfDay_ID	This is the start time of the session translated to the GMT time zone. See also, Populating CALENDAR and TIMEOFDAY tables.

Table Name: Result

Table description: For your use in classifying sessions.

Column Name	Column Description
Result_ID	The primary key. We recommend that you use numbers automatically generated by the database. In DB2/UDB, this column is defined as IDENTITY GENERATED BY DEFAULT. On ORACLE, the sequence RESULT_ID is created for your use with this table.
Name	For your use.

Table Name: SessionParms

Table description: This table lists all the Key/Value pairs associated with the session referrer.

Column Name	Column Description
Parms_ID	Foreign key to the ID column of the PARMS table.
Session_ID	Foreign key to the ID column of the SESSION_FACT table.
ParmType	Identifies the type of Key/Value data.

Table Name: Subdomains

Table description: This table is RESERVED for future Feedback component use.

Column Name	Column Description
Domain_ID	Foreign key to the ID column of the DOMAIN table. The TCP/IP domain this subdomain belongs to. Name Subdomain name string from a resolved TCP/IP address.
Subdomain_ID	The primary key, a unique number automatically generated by the database.

Table Name: TimeOfDay

Table description: Initially, this table is empty. Table rows are reserved to list all seconds in a day, when the database is populated. The table can be populated with a new configuration task for applications that use custom logging or reports that reference this table.

Column Name	Column Description
Minute	a number in the range 0..59
Hour	a number in the range 0..23
Second	a number in the range 0..59
AggrHourID	For each database row, indicates which hour it belongs to.
TimeOfDay_ID	The primary key. A unique number automatically generated by the database.
SecondsSinceMidnig	The number of seconds elapsed since midnight.
TimeSpan_ID	Foreign key to the ID column of the TIMESPAN table.

Table Name: TimeSpan

Table description: For your use in classifying time of day values.

Column Name	Column Description
TimeSpan_ID	The primary key. We recommend that you use numbers automatically generated by the database. In DB2/UDB, this column is defined as IDENTITY GENERATED BY DEFAULT. On ORACLE, the sequence TIMESPAN_ID is created for your use with this table.
Name	For your use.

Table Name: UserAgents

Table description: This table contains all UserAgents processed during data collection.

Column Name	Column Description
Browser_ID	Foreign key to the ID column of the BROWSERS table.
Name	The UserAgent string as available from the Web page hit.
Platform_ID	Foreign key to the ID column of the PLATFORMS table.
UserAgent_ID	The primary key. A unique number automatically generated by the database.

Table Name: Users

Table description: This table contains all userids processed from data collection.

Column Name	Column Description
FirstVisitDate_ID	Date of first visit to the site by this userid. See also, Populating CALENDAR and TIMEOFDAY tables.
Name	The userid string as parsed from the Web log record.
User_ID	The primary key. A unique number automatically generated by the database.

Table Name: Value

Table description: Lists all values (from Key/Value pairs) encountered during data collection.

Column Name	Column Description
Value_ID	A unique identifier automatically generated by the database.

Column Name	Column Description
Value	Value string of Key/Value pair parsed from a Web log record.

The following tables are included in the Feedback schema, however, the tables are not used and are not populated.

- Aggregate_Content
- AggregateKey
- Aggregates
- AggregateStatus
- Categories
- Category_Patterns
- Category_Sets
- CategoryMap
- CookiesStatus (Note: This table may be pre-populated, however, the table is not used.)
- JavaScriptStatus
- Linkage
- Log_file_status
- Logs
- ResetStatus
- ReturnCodes
- ServerNodes
- Web_Nodes

Key value pairs:

The additional information passed by rule logging events and certain bean logging events is logged to the Feedback database in the form of key value pairs.

Rule logging events and certain bean logging events pass additional information such as rule names, action names, resource collection names, resource identifiers, etc. when they are fired (refer to the individual rule logging and logging bean sections for more information on what information may be passed). This data is logged to the Feedback database in the form of key value pairs. Key strings representing each type of data are stored in the key table of the Feedback schema, while the values that each key references are recorded in the value table of the Feedback schema. Refer to the Feedback schema diagram for more information on how to join these tables to retrieve a mapping of values onto keys.

The following tables describe what each key represents:

Rule related keys:

Table 378. Key descriptions

Key	Description
wcpRule	The name of the rule that was fired
wcpCampaign	The campaign associated with the rule

Table 378. Key descriptions (continued)

Key	Description
wcpCollection	The name of the resource collection associated with the rule
wcpResourceId	An identifier for each individual resource the associated resource collection

Action bean related keys:

Key	Description
wcpAction	The name of the action being logged
wcpActionResId	The resource (if any) that has been acted upon
wcpActionCollection	The name of the resource collection associated with the resource
wcpActionRule	The name of the rule (if any) that surfaced the resource being acted upon
wcpActionCampaign	The campaign (if any) associated with the rule

Category bean related keys:

Key	Description
wcpCategory	A category string being logged

Rating bean related keys:

Key	Description
wcpRating	The rating value being logged
wcpRatingResId	The resource (if any) that is associated with the rating
wcpRatingCollection	The name of the resource collection associated with the resource

Developing a personalized portlet

This exercise demonstrates how to use Personalization features of WebSphere Portal and Rational Application Developer to build your first personalized portlet. Your final result is a working portlet that uses Personalization rules and content spots to display personal news based on user attributes (or profiles).

The demonstration Web site is organized into the following three pages:

1. **Front Page**
   - **Internal News** displays internal YourCo news for the user. This portlet displays the news upon clicking upon one of the links.
   - **General News** displays external worldwide news articles for users. This portlet messages to the GeneralNewsDetails upon clicking upon one of the links.

- **UserInfo** displays the current user's information. This information is used in rules to target content throughout the website to the particular preferences of your users. This user information can be modified using the edit feature.

2. **Partners Page**
   - **Products** displays products for the current user. This portlet messages to the PartnersDetails upon clicking upon one of the links.
   - **Promotions** runs a simple rule targeting the information in the portlet to the user's preferences.

   The promotions portlet also demonstrates running a campaign and the splitting of rules within a campaign. In this case, between March 1 and April 16, 2005 the Tax Season campaign will run. Within this campaign, one content spot is filled with 2 distinct rules, Get Tax Season Promotions and Get 3 Promotions about IRAs. Get Tax Season Promotions will be run 70% of the time and Get 3 Promotions about IRAs the remaining 30%.

   A second campaign is also running in the promotions spot between April 14 and April 15, 2005. This campaign has a higher priority than the Tax Season campaign so it will run at this time.
3. **Services Page**
   - **Services** displays services that are offered to the user. Clicking on one of the services will display the details of the service. If logging is enabled, clicking on one of the services will log the fact that this particular type of service was selected by the user.
   - **Offers** displays current offers for the user. This portlet demonstrates calling an analysis bean from a rule.

Personalization can use database content. This exercise uses the sample database that comes with the Personalization sample. The following tables are used:

- PZN_USER: contains user information, including profile attributes
- PZN_OFFERS: contains text for special offers categorized by customer type

Table 379. Sample data from PZN_USER table (primary key = USERNAME)

FIRST_NAME	LAST_NAME	DEPARTMENT	CUSTOMERTYPE	USERNAME*
Scott	Green	Loans	Gold	scott
Tawana	Streble	Human Resources	Platinum	tawana
Marge	Roorda	Human Resources	Platinum	marge
Andy	McPherson	IT	Titanium	andy

Table 380. Sample data from PZN_OFFERS table (primary key = OFFER_ID)

OFFER_ID*	CUSTOMERTYPE	TITLE	DETAILS
1	Platinum	2nd House Mortgage	Excellent Mortgage rates on your beach house.
2	Gold	Add other Cards	Add other members of your household to your account.
3	Platinum	ARM Mortgage	1/3 5.9% APR Mortgage for 27 years.

Table 380. Sample data from PZN_OFFERS table (primary key = OFFER_ID) (continued)

OFFER_ID*	CUSTOMERTYPE	TITLE	DETAILS
4	Titanium	Free Checking Deluxe	Free Checking with the attainment of a YourCo Financial Credit Card.

Notice the common element in these tables is the column CUSTOMERTYPE. This column is important when you invoke Personalization later in the exercise.

1. "Prerequisites for the Personalization portlet exercise" on page 2388
   This exercise is intended for users with significant WebSphere Portal Express administration experience and strong portlet Java development skills.
2. "Install the Personalization sample" on page 2389
   This exercise installs the Portal Personalization sample and configures your database for the Personalization sample. No additional database requirements are needed.
3. "Create the JSP file in Rational Application Developer" on page 2389
   Create a basic JSR 168 portlet file for the Personalized Offers project and add the welcome text to the portlet.
4. "Create the Personalization content resource classes and content spot" on page 2390
   View the steps to create the Personalization content resource classes and content spot using IBM Rational Application Developer.
5. "Creating the Personalization user resource classes and content spot" on page 2391
   Use the Project Explorer of IBM Rational Application Developer to create the user resource classes and content spot for the Personalization demo that creates the Personalized Offers portlet for different customer profiles.
6. "Coding the portlet JSP" on page 2392
   Use the Rational Application Developer Project Explorer to work with the pers_offers.Pzn_offersSpot.class and the pers_offers.Per_Offers_UserSpot.class to define the context parameters **offersSpot** and the **userSpot**. Code the setRequest calls to pass the user context to personalization.
7. "Export the WAR file and install the portlet" on page 2394
   Export the IBM Rational Application Developer project to a war file and install the portlet on a portal page.
8. "Import Personalization Workspace resource collections" on page 2394
   Use the Personalization Navigator to create Workspace folders for the resource collections. Then import the resource collections into the Workspace of the Personalization Navigator.
9. "Create a simple content rule" on page 2395
   Use the Personalization Editor to create a business rule for a resource collection.
10. "Create a content spot" on page 2395
    Use the Personalization Editor to create a placeholder for the rule that renders the selected content on a Web page. This placeholder is the *content spot*. Specify which rule to place in the content spot; this is called mapping the rule to the content spot or creating a *rule mapping*. In this example, the content is a resource collection.
11. "Enhance the Personalized Portlet" on page 2396
    Introduce enhancements in the Personalized Portlet.

12. “Insert dynamic table html/jsp code” on page 2397
    Use Rational Application Developer to code the dynamic table in the Personalized Offers portlet JSP. Export the project as a web archive (WAR) file. Open the **WebSphere Portal Administration** page for Portlet Management and update the web module that contains the Personalized Offers portlet.
13. “Modify resource collection properties” on page 2398
    Use the Personalization Navigator to specify the Translator Class for the resource collection.
14. “Create the user profiler rule” on page 2399
    Use the Personalization Editor to create a profiler rule for users who qualify as customers for Gold Offers in the Personalized Offers resource collection.
15. “Create additional advanced rules” on page 2399
    For each customer type specified by a profile, add to the business rule (the profiler) by selecting actions to build the syntax of the rule.
16. “Change content spot rule mapping” on page 2400
    Change the default rule mapping to the new binding rule that you created for different customer types who will use the Personalized Offers portlet. Test the portlet for use by all customer types. Verify that the portlet displays the personalized content that is specified by the business rule for each user profile.
17. “Personalized List portlet” on page 2401
    The Personalized List portlet provides a ready-to-use portlet for displaying personalized content from rules, content spots, or resources. In many cases, it eliminates the need to code new portlets and JSP files yourself. You can use this portlet instead of coding the IBM Rational Application Developer portlet.
18. “Uninstall Personalization sample and database” on page 2402
    View the steps to uninstall Personalization sample demo, database, and users.

Prerequisites for the Personalization portlet exercise:

This exercise is intended for users with significant WebSphere Portal Express administration experience and strong portlet Java development skills.

Before you begin

Software requirements:

- WebSphere Portal Express, including IBM Web Content Manager. You can use the default database or another supported database.
- Rational Application Developer Version 7.0.0.6 or 7.5 with the Portal Tools feature. No agent controller is required. No WebSphere Test Environment or integrated Portal environment is used in this example. For additional information about Rational Application Developer, visit the IBM software library.

Related concepts:

Chapter 11, “Administering,” on page 1041
Use the administration tools that are provided with the portal to do various day-to-day administration tasks. There are two methods for editing portal setup: using the administration portlets or the XML configuration interface. The administration portlets are a convenient way to make real-time updates to the portal's configuration. While the XML configuration interface is suited to more advanced administration, including batch processing of updates.

Related tasks:

 IBM software library

Install the Personalization sample:

This exercise installs the Portal Personalization sample and configures your database for the Personalization sample. No additional database requirements are needed.

Before you begin

Ensure you have the basic system requirements listed in the Prerequisites for the Personalization portlet exercise topic.

Procedure

1. Start the IBM WebSphere Portal Express server.
2. From a command prompt, navigate to *wp_profile_root*/ConfigEngine.
3. Enter the following commands to install the Personalization sample and create the users for this sample:
   - Linux:

     ```
     ./ConfigEngine.sh create-pzndemo-users -DPortalAdminPwd=password
     -DWasPassword=password
     ```

     ```
     ./ConfigEngine.sh install-pzndemo -DPortalAdminPwd=password
     -DWasPassword=password
     ```

   - Windows:

     ```
     ConfigEngine.bat create-pzndemo-users -DPortalAdminPwd=password
     -DWasPassword=password
     ```

     ```
     ConfigEngine.bat install-pzndemo -DPortalAdminPwd=password
     -DWasPassword=password
     ```

4. Stop the WebSphere Portal Express server.
5. Restart the WebSphere Portal Express server
6. Confirm that the users were created by logging in as the following users. Use pzndemo as the password for each user.
   - scott
   - marge
   - tawana
   - andy
7. Navigate to **Personalization** > **Demo**, then click through the different pages and portlets for each user. Notice the different information that displays for each user.
8. Log out of WebSphere Portal Express.
9. Stop the WebSphere Portal Express server. This action is necessary to free the single connection to the Derby database, so you can continue the next steps in this exercise.

Results

The sample demo and database configuration is complete. You can now begin coding a basic personalized portlet.

Create the JSP file in Rational Application Developer:

Create a basic JSR 168 portlet file for the Personalized Offers project and add the welcome text to the portlet.

Before you begin

Before you begin, complete the following prerequisites:

- Ensure you have properly installed the demo and databases.
- Ensure that the WebSphere Portal Server is stopped so that you can make a connection to the database.

Procedure

1. Launch Rational Application Developer.
2. Click **File** > **New** > **Project** and select **Portlet Project**.
3. Click **Next**. The Confirm Enablement window appears. Click **OK** to enable Portal Development.
4. Complete the New Portlet Project screen with the following information:
   a. In the **Project Name** field, type `Pers_Offers`.
   b. Use the default project contents.
   c. Select **WebSphere Portal v6.1 stub** from the **Target Runtime** list.
   d. Select **Add project to an EAR** and in the field **EAR Project Name** select *Pers_OffersEAR* from the list.
   e. In the **Portlet API** field, select *JSR 168 Portlet* from the list.
   f. Select **Create a portlet** and specify the following properties:
      - In the **Portlet Name** field, type `Pers__Offers`.
      - In the **Portlet Type** field, select *Basic Portlet* from the list.
5. Click **Next**.
6. Accept the default Portlet Settings and click **Next**.
7. No Actions are necessary. Clear any selected actions and click **Next**.
8. No Advanced Settings are necessary. Click **Finish**.
9. The JSP file opens automatically in Design mode. Edit the text of the portlet to display `Welcome to Personalized Offers!`.
10. Save and close the JSP.

Results

You have created a JSP file using Rational Application Developer.

What to do next

You can now create the Personalization content resource classes and content spot.

Create the Personalization content resource classes and content spot:

View the steps to create the Personalization content resource classes and content spot using IBM Rational Application Developer.

Procedure

1. In the Project Explorer tab in Rational Application Developer, right-click on the Pers_Offers project and choose **New** > **Other**.
2. From the New window, select **Portal** > **Personalization** > **Content or User Resource**.
3. Click **Next**.
4. Select the following options by clicking the appropriate radio button:

a. **SQL**
b. **Web Content**

5. Select **Create a new connection** and click **Next**.
6. Enter the following values:
   a. From the **JDBC driver** drop-down list, select **Derby Embedded JDBC Driver**.
   b. From the **Database location** field, click **Browse**. Navigate to and select *wp_profile_root*/PortalServer/derby/pzndemo_db/
   c. From the **Class location** field, click **Browse**. Navigate to and select *AppServer_root*/derby/lib/derby.jar
7. Click **Next**. The personalization resource wizard opens.
8. Expand PZNDEMO.
9. On the **Tables** tab, select **PZN_OFFERS**, and click the arrow to add the table to the list of selected tables. Click **Primary Table** to mark it as the primary table.
10. Select the **Columns** tab.
11. On the **Columns** tab, move all columns to the selected columns list by clicking the double arrow button. Notice the primary key is the column Offer_ID.
12. Click the **Mappings** tab.
13. On the **Mappings** tab, select **Customertype** and click **Populate**.
14. Click the **Select** buttons and expand **PZNDEMO** > **PZN_OFFERS** to select CUSTOMERTYPE for the **Description** and **Value** fields. Click **OK**.
15. Click the **Deployment** tab.
16. On the **Deployment** tab, change the datasource to jdbc/pzndemo. This datasource is defined in WAS by the Personalization demo program installation.
17. Click **Next**.
18. Set the package name as pers_offers. Select **Generate a Content Spot for this resource**. Select **Include schema names** in the generated Resource Runtime Manager.
19. Click **Finish**.

Results

You can now see the new JAVA classes in your project:

What to do next

You can now see the new JAVA classes in your project, and create the Personalization user resource classes and content spot.

Creating the Personalization user resource classes and content spot:

Use the Project Explorer of IBM Rational Application Developer to create the user resource classes and content spot for the Personalization demo that creates the Personalized Offers portlet for different customer profiles.

Procedure

1. In the **Project Explorer** tab of Rational Application Developer, right-click the **Pers_Offers** project and choose **New** > **Other**.

2. From the **New** window, select **Portal** > **Personalization** > **Content or User Resource**.
3. Click **Next**
4. Select one of the following radio buttons:
   - **SQL**
   - **Web users**
5. Select **Use an existing connection** and select **pzndemo_db** from the list of existing connections.
6. Click **Next**. The personalization resource wizard opens.
7. Expand **PZNDEMO**.
8. On the **Tables** tab, highlight **PZN_USER**. Click the appropriate arrow button to select the table.
9. Right-click **PZN_USER** and select **Edit Table**.
10. Change the display name of **PZN_USER** to **Per_Offers_User** to avoid naming conflicts with the previously installed demo code. Click the Primary Table button to mark it as the primary table.
11. Select the **Columns** tab
12. On the **Columns** tab, move all of the columns to the selected columns area by clicking the double arrow button. Notice the primary key is the column **USERNAME**.
13. Click the Mappings tab.
14. On the Mappings tab, select **Customertype** and click **Populate**.
15. Click the **Select** buttons and expand **PZNDEMO** > **PZN_USER** to select CUSTOMERTYPE for the **Description** and **Value** fields. Click **OK**.
16. Click the **Deployment** tab.
17. On the **Deployment** tab, change the datasource to `jdbc/pzndemo`. This datasource is defined in WAS by the Personalization demo program installation.
18. Click **Next**.
19. Set the package name as `pers_offers`. Select **Generate a Content Spot for this resource**. Select **Include schema names in the generated Resource Runtime Manager**.
20. Click **Finish**.

Results

You can now see the new JAVA classes in your project:

What to do next

You can now finish coding the portlet JSP file.

Coding the portlet JSP:

Use the Rational Application Developer Project Explorer to work with the pers_offers.Pzn_offersSpot.class and the pers_offers.Per_Offers_UserSpot.class to define the context parameters **`offersSpot`** and the **`userSpot`**. Code the setRequest calls to pass the user context to personalization.

Procedure

1. From the Rational Application Developer Project Explorer, open `PersOffers/WebContent/pers_offers/jsp/html/Pers_OffersPortletView.jsp`
2. Drag the `pers_offers.Pzn_offersSpot.class` file from `PersOffers/WebContent/WEB-INF/classes/pers_offers` onto the JSP file. This class is one of the classes you generated in the content resource wizard.
3. In the **Properties** window, select **offersSpot** from the **Id** list.
4. Drag the `pers_offers.Per_Offers_UserSpot.class` file from `PersOffers/WebContent/WEB-INF/classes/pers_offers` onto the JSP file.
5. In the **Id** field, type **userSpot**.
6. Switch to source mode in the JSP file editor.
7. Replace the existing text with the following sample. The `setRequest` calls pass the user context to personalization:

```
<%@ page session="false" contentType="text/html" import="java.util.*, pers_offers.*"%>

<%@ taglib uri="/WEB-INF/tld/portlet.tld" prefix="portletAPI" %>

<portletAPI:init/>

<jsp:useBean class="pers_offers.Pzn_offersSpot" id="offersSpot"></jsp:useBean>

<jsp:useBean class="pers_offers.Per_Offers_UserSpot" id="userSpot"></jsp:useBean>

<%

offersSpot.setRequest(request);

userSpot.setRequest(request);

%>
```

8. Save the file and check for syntax errors.
9. Insert the following code at the end of the JSP file:

```
<DIV style="margin: 6px">


<H3 style="margin-bottom: 3px">Welcome to Personalized Offers!</H3></DIV>

<BR>

Here is a personalized offer:<BR>

<%=offersSpot.getRuleContent()[0].getTitle() %>

<br>

<%=offersSpot.getRuleContent()[0].getDetails() %>
```

10. Save and close the JSP file.

What to do next

You finished the basic coding of the personalized portlet. You can now export the WAR file from Rational Application Developer and install the portlet in WebSphere Portal.

Export the WAR file and install the portlet:

Export the IBM Rational Application Developer project to a war file and install the portlet on a portal page.

Before you begin

Ensure that you have properly followed the steps in Coding the portlet JSP.

Procedure

1. Export the Rational Application Developer project to a war file called `PersOffers.war`.
   a. Right click Pers_Offers and select **Export** > **WAR file**.
   b. In the **Destination** field, type `filepath/PersOffers.war`.
   c. Select **Export source files**.
   d. Click **Finish**.
2. Start WebSphere Portal.
3. Log in as the Portal administrator (`wpsadmin`).
4. Click the **Administration menu** icon. Then, click **Portlet Management** > **Web Modules**.
5. Click **Install**.
6. Complete the installation of `PersOffers.war`. Verify that installation was successful.
7. Open **Portlet Management** > **Portlets**.
8. Search for the Pers_Offers portlet and grant **Privileged User** to **All Authenticated Portal Users**.
9. Create a new page called Pers Offers by completing the following steps:
   a. Select **Portal User Interface** > **Manage Pages**.
   b. Select **Content Root** > **Personalization**.
   c. Select **New Page**.
   d. Label the Page Pers Offer.
10. Add the Pers_Offers portlet to the page.
11. Click **Done**.

    Note: The portlet is not ready to run yet. If you try, you get an error.

What to do next

You can now import the Personalization workspace resource collections.

Import Personalization Workspace resource collections:

Use the Personalization Navigator to create Workspace folders for the resource collections. Then import the resource collections into the Workspace of the Personalization Navigator.

Before you begin

Before you can use the content and user resources in the Personalization Navigator, you must place these class files into a directory accessible by that portlet. To do this, export the pers_offers folder in RAD under `Pers_Offers/Java`

Resources/JavaSource as a JAR file. Make the target location *PortalServer_root*/pzn/prereq.pzn/collections/pers_offers.jar. Accept the defaults and click **Finish**, then restart the server.

Ensure you have properly followed the steps in Export the WAR file and install the portlet.

Procedure

1. Click the **Personalization** tab.
2. In the Personalization Navigator portlet, click **New** > **Folder**
3. Enter the name Pers Offers and click **Create**.
4. Change to the Pers Offers folder.
5. Click **Import**.
6. Browse to find the Pzn_offers.hrf file in your installed Pers_Offers directory under Pers_Offers.war/WEB-INF/pzn-resourceCollections/pers_offers.
7. Click **Import**.
8. See the resource collection in the Workspace.
9. Do the same to import the Pers_Offers_User.hrf file.

What to do next

You can now create a simple content rule.

Create a simple content rule:

Use the Personalization Editor to create a business rule for a resource collection.

Before you begin

Before you begin this procedure, ensure you have imported the resource collections into the Personalization Workspace.

Procedure

1. While still in the Pers Offers folder, click **New** > **Rule**.
2. Type Show Gold Offers in the New Rule field.
3. Select **Select Action** from the **Rule Type** drop-down list.
4. Click **content** next to **Select**.
5. Select **Pzn_offers** from the drop-down list.
6. Click **attribute** and select **Customertype** in the drop-down list.
7. Hover over **value** and click the greater than symbol, **>**, then select **Gold** in the drop-down list.
8. Click **Save**.

What to do next

You can now create a content spot for your rule.

Create a content spot:

Use the Personalization Editor to create a placeholder for the rule that renders the selected content on a Web page. This placeholder is the *content spot*. Specify which

rule to place in the content spot; this is called mapping the rule to the content spot or creating a *rule mapping*. In this example, the content is a resource collection.

Before you begin

Before you begin this procedure, ensure you have created the simple content rule *Show Gold Offers* for the *Personalized Offers* resource collection.

About this task

Note: You must use the same name as the original content spot's display name. Do not place the new content spots into a folder, unless your display name already includes slashes. Place them directly into the root Workspace.

Procedure

1. Return to the root directory in the Personalization Navigator.
2. Click **New** > **Content Spot** to display the Personalization Editor.
3. Type `Pzn_offersSpot` in the title field of the content spot.
4. Select **Pzn_offers** from the **Output Type** drop-down list.
5. Click the button in the **Default Mapping** section.
6. Expand the **Pers Offers** folder in the tree, select **Show Gold Offers** and click **OK**.
7. Click **Save**.
8. Test the content spot with the Personalization Editor portlet by selecting it in the Personalization Navigator and clicking the **Preview** tab in the Personalization Editor.

   Note: If the previous values appear on the Preview tab, the rule and content spot work.
9. View the page with your Pers_Offers portlet.

Results

You can now edit the JSP file to contain dynamic code.

Enhance the Personalized Portlet:

Introduce enhancements in the Personalized Portlet.

Before you begin

Ensure you have properly followed the steps in “Create a content spot” on page 2395.

About this task

We need a few more things in this portlet to make it interesting:

- User profiling
- Special userid handling, so we can build the user profiles
- A more complex binding rule
- A dynamic html table, to display any number of special offers

Insert dynamic table html/jsp code:

Use Rational Application Developer to code the dynamic table in the Personalized Offers portlet JSP. Export the project as a web archive (WAR) file. Open the **WebSphere Portal Administration** page for Portlet Management and update the web module that contains the Personalized Offers portlet.

Before you begin

Before you begin this procedure, ensure that you created a content spot.

Procedure

1. Open Rational Application Developer and open `Pers_OffersPortalView.jsp` in edit mode.
2. Insert the following code at the end of the JSP page:

```
<HR>

Here are all your personalized offers:

<br>

<%

try {

pers_offers.Pzn_offers[] items = offersSpot.getRuleContent();

pers_offers.Pzn_offers item = items[0]; // throws an exception if empty. %>

<TABLE WIDTH="100%">

<TBODY>

<%

for (int i = 0, c = 0; ; ) {

if (c == 0) { %>

<TR bgcolor="e7e7e7"><%

}

else { %>

<TR><%

} %>

<TD><%= item.getTitle() %>: <%= item.getDetails() %>

</TD>

</TR><%

c = c == 0 ? 1 : 0;

i++;

try {

item = items[i];
```

```
}

catch (java.lang.ArrayIndexOutOfBoundsException _e0) {

break;

}

} %>

</TBODY>

</TABLE><%

}

catch (java.lang.ArrayIndexOutOfBoundsException _e0) {

%><FONT>There are no current articles to display. </FONT><%

} %>
```

3. Save and close the file. Verify no errors other than exception handling errors.
4. Export the project as a war file. Close Rational Application Developer.
5. Log in to WebSphere Portal as `wpsadmin`. Click the **Administration menu** icon. Then, click **Portlet Management** > **Web Modules**.
6. Click the **Update** button to update the web module.
7. Complete the installation process by using the newly created war file.

What to do next

You can now modify the resource collection properties for Pzn_Offers_User.

Modify resource collection properties:

Use the Personalization Navigator to specify the Translator Class for the resource collection.

Before you begin

Before you begin this procedure, ensure that you have inserted the dynamic table code into the Personalized Offers portlet.

Procedure

1. Open the Personalization Navigator.
2. Navigate to the `Pers_Offers` folder.
3. Select the **Per_Offers_User** resource collection.
4. In the Personalization Editor, click **Edit**.
5. Set the Translator Class to `com.ibm.websphere.personalization.security.RegularExpressionSecurityTranslator` and click **Save**.

What to do next

You can now create a user profiler rule.

Create the user profiler rule:

Use the Personalization Editor to create a profiler rule for users who qualify as customers for Gold Offers in the Personalized Offers resource collection.

Before you begin

Before you begin this procedure, ensure that you have modified the resource collection properties to specify the Translator Class.

About this task

In the Personalization Editor, follow these steps:

Procedure

1. Within the Pers Offers folder, click **New** > **Rule**.
2. Enter the following values:
   a. In the **New Rule** field, type `Pers Offers User Profiler`.
   b. Select **Profiler** in the Rule Type menu.
3. Click **Profile**, type **Gold** in the Profile field, and click **Submit**.
4. Click **attribute** and select **Per_Offers_User** in the drop down, then **Customertype** in the expanded drop down.
5. Click the greater than symbol, >, next to value and select **Gold** in the drop down list.
6. Click **add Profile** to add another conditional expression to the profiler rule.
7. Complete the profiler rule by adding profiles that define distinct sets of users. Each profile expresses the conditions of the user type as illustrated in the following example.

   Note: Select **No value** for the **Otherwise** field.
8. Click **Save**.

Results

You can now create some additional advanced rules for your personalized portlet.

Create additional advanced rules:

For each customer type specified by a profile, add to the business rule (the profiler) by selecting actions to build the syntax of the rule.

Before you begin

Ensure you have properly followed the steps in "Create the user profiler rule."

Procedure

1. Within the Pers Offers folder, select **New** > **Rule**
2. Type `Show Platinum Offer` in the **New Rule** field.
3. Select **Select Action** from the **Rule Type** drop-down list.
4. Edit the rule so that its `Customertype is Platnum` and click **Save**.
5. In the Pers Offers folder, create another rule called Show Titanium Offers. Select Pzn offers whose Customertype is titanium.

6. Within the same folder, select **New** > **Rule**
7. Type `Show Offers by Customer Type` in the **New Rule** field.
8. Select **Binding** from the **Rule Type** drop-down list.
9. Click **Profiler** > **Select a Profiler...**.
10. Expand the Pers Offers folder, select **Pers Offers User Profiler** and click **OK**.
11. Click **Profile** and select **Gold** from the drop-down list.
12. Click **do Action** > **Select Actions...**.
13. Select **Show Gold Offers** and click **OK**.

    Tip: To find **Show Gold Offers**, you may reorder the columns to make navigation easier or page through the list of rules.
14. Click **Profile** and choose **Platinum**. Complete the fields as follows:

What to do next

You can now change your content spot mapping.

Change content spot rule mapping:

Change the default rule mapping to the new binding rule that you created for different customer types who will use the Personalized Offers portlet. Test the portlet for use by all customer types. Verify that the portlet displays the personalized content that is specified by the business rule for each user profile.

Before you begin

Before you begin this procedure, ensure you have properly created additional advanced rules.

Procedure

1. In the Personalization Navigator, select the content spot **Pzn_offersSpot**, located in the **Workspace Root**.
2. In the Personalization Editor portlet, click **Edit**.
3. Change the default rule mapping to the new binding rule and click **Save**.
4. Log in to the Portal as `Scott`, with password `pzndemo`, a Gold customer.
5. Run the enhanced Personalized portlet
6. Log out, then log in as `Marge`, with password `pzndemo`, a Platinum customer.
7. Run the portlet.

Results

If you run the portlet as *wpsadmin*, it fails because *wpsadmin* does not exist in the user resource database, and the programmer who coded the JSP did not code the proper error handling.

Congratulations! You have finished building a Personalized portlet.

What to do next

The next topic, *Personalized list portlet*, shows how to use this portlet instead of coding the Rational Application Developer portlet.

Personalized List portlet:

The Personalized List portlet provides a ready-to-use portlet for displaying personalized content from rules, content spots, or resources. In many cases, it eliminates the need to code new portlets and JSP files yourself. You can use this portlet instead of coding the IBM Rational Application Developer portlet.

About this task

Restriction: The Personalized List portlet is not intended to be used with the Web Content resource collection or with rules that involve the Web Content resource collection. To display Web Content rules, use a Portal Personalization Component in IBM Web Content Manager. When the Web Content resource collection is used with the Personalized List portlet, certain attributes like authoring template shows raw values from the repository that cannot be translated to a readable form. The details page of the Personalized List portlet does not show the results of the rendered content. The details page shows some attribute from the content, such as the creator or last modified date, which is not suitable for production use of content from Web Content Manager.

The intended use of the Personalized List portlet is to display personalized lists of documents. The Personalized List can also be used with some generated or custom resource collections.

Procedure

1. Copy the pers_offers folder from the /Pers_Offers.war/WEB-INF/classes directory under your Pers_Offers portlet in the installedApps location into *wp_profile_root*/PortalServer/pzn/collections. You might need to create this classes folder first.
2. Restart IBM WebSphere Portal Express.
3. Log in as the admin user.
4. Click the **Administration menu** icon. Then, click **Portlet Management** > **Portlets**.
5. Search for the Personalized List portlet.
6. Make a copy of the Personalized List portlet, named **Personalized List Special Offers**.
7. Set **access control** for the new portlet so **All Authenticated Portal Users** are Privileged Users.
8. Add the Personalized List Special Offers portlet to a new page under Personalization called **Pers List Portlet - Rules**.
9. Display the new page with the new portlet. Click the portlet menu on the Personalized List Special Offers portlet, and select **Configure** from the drop-down list.
10. Click the menu icon under **Which Personalization resources are retrieved** and select **Select a Rule, Content Spot or Resource Collection**.
11. Expand the **Pers Offers** folder and select **Pzn_offers** and click **OK**.
12. Click **Display Options**.
13. Select the following options and click **OK**.
14. Click **OK** again to see the portlet.
15. Click one of the title links to see the details of that resource.
16. Click **Back**.
17. Configure the portlet again to show more personalized offers.

18. From the Personalization Picker, select the content spot **Pzn_offersSpot** under the Workspace Root and click **OK**.
19. Click **Display Options**, set the following values, and click **OK**.
20. Set the Title Attribute and Detail Attribute values to **Fixed**. Click **OK**.
21. Complete the configuration and see that the Personalized List Special Offers portlet is empty. This is because the administrator did not enter customer details in the pzndemo database for the user resource that is being used.
22. Log in as Scott and view the Personalized List Special Offers portlet. The portlet displays the personalized offers that are based on the rule, *Show Offers By Customer Type*, which is mapped to the content spot *Pzn_offersSpot*.
23. Test this portlet, logging in as each pzndemo user having a different profile. You must get the same content results that you saw previously by using your own custom-built portlet, *Pers_Offers*, except you do not have to code a portlet or a JSP file.

Uninstall Personalization sample and database:

View the steps to uninstall Personalization sample demo, database, and users.

About this task

Procedure

1. From a command prompt, navigate to *wp_profile_root*/ConfigEngine.
2. From this command prompt, run the following commands:
   - Linux:

     ```
     ./ConfigEngine.sh remove-pzndemo -DPortalAdminPwd=password
     -DWasPassword=password
     ./ConfigEngine.sh remove-pzndemo-users -DPortalAdminPwd=password
     -DWasPassword=password
     ```

   - Windows:

     ```
     ConfigEngine.bat remove-pzndemo -DPortalAdminPwd=password
     -DWasPassword=password
     ConfigEngine.bat remove-pzndemo-users -DPortalAdminPwd=password
     -DWasPassword=password
     ```

Results

The sample demo, database, and users will be removed.

Note: The database connection to the included Apache Derby database may stay open and prevent removal of the database directory. Restart IBM WebSphere Portal Express and run `ConfigEngine.bat/sh remove-pzndemo-database -DPortalAdminPwd=password -DWasPassword=password` if this is a problem. The directory `PznDemo` in the root of the Portal Personalization Workspace may need to be removed manually.

Personalization programming reference

IBM WebSphere Portal Express provides the programming model, processes, and APIs for the Personalization rules and resource engines.

"Preparing your personalized application" on page 2404
Before deploying applications that take advantage of the features of Portal Personalization, certain features must be configured in order to work properly. The Feedback and LikeMinds components of Personalization both communicate

with their databases using Java data sources. Before using either of these components, you must create resource references to the data sources in your project.

"Programming model" on page 2405
Personalization builds on the same programming model used by WebSphere Application Server. Two components of that model are Java Server Pages (JSP) and business logic (JavaBeans and enterprise beans). JSP files enable you to effectively separate HTML coding (presentation) from business logic. You isolate your business logic in beans that the Web page author embeds in the JSP.

"User and content models" on page 2405
The first step in developing a Personalization solution is to analyze your business conditions and events to determine the users and content to be targeted. The business users and administrators are primarily responsible for this task.

"How the rules engine works" on page 2406
The rules engine processes and delivers the results of a rule execution to a content spot contained in a web page. The content spot is marked by a content spot bean, which is placed in either a JSP file or a servlet. The JSP file or servlet is then linked to the web page.

"Workload management" on page 2409
Only one Portal Personalization engine can be installed on an application server. Consequently, there is one Personalization engine for each servlet engine. Personalization supports application server clustering, because each Personalization instance in the cluster shares the same IBM Java Content Repository. Therefore, each Personalization instance accesses the same customer data stores.

"Using the Personalization APIs" on page 2409
Personalization provides open APIs that enable the Personalization run-time environment and Rational Application Developer to access user and content data in customer data stores.

"Generic query framework" on page 2412
The generic query framework enables resource collection developers to convert a property-based query object into a language specific executable query string. It contains query component classes, and builder and callback interfaces.

"Request context interface" on page 2414
This is the interface used to access various attributes for rules. For HTTP contexts, it provides access to the HttpRequest and HttpSession attributes. For non-HTTP contexts, it provides the same interface to a surrogate for the request and session.

"Sample Personalization resources XML file" on page 2415
Use the sample file `ExportFromServlet.xml` as a reference for coding other Personalization actions.

"Content spot exits" on page 2415
Content spot exits provide the ability to alter the default flow of processing of content spots, making it possible in the runtime environment to override the rule to process, the current user, and the results of rule processing. Instances of the same exit class are instantiated for all content spots. The exit class interface name is public interface RuleExit.

"Resource cache" on page 2417
Personalization uses the WebSphere Application Server Dynamic Cache service to cache the results of select rules and to cache the rules themselves in a DistributedMap. When publishing, importing or saving rules, the cache is flushed automatically to ensure that the site is current.

"Programmatically starting rules" on page 2418
All types of rules can be accessed programmatically within a Java application. For example, a profiler can be used to determine the behavior that an application must exhibit depending on the current user, or an action can return content to your application for further processing before the content is displayed. Rules are mapped to content spots, and because a content spot is an implementation of a JavaBeans, it can be programmatically declared and implemented.

"Rule Exception Handling in the run-time environment" on page 2419
The default method of error or exception handling within the Personalization run-time environment is for the engine to print out a trace error to the application server's stdout log. Using an exception handling utility, it is possible to specify the type of output (error message or stacktrace), the output log file to use (stdout or stderr), and whether the exception should be rethrown to the JSP. The additional exception handling capabilities can be set on a per-request basis or globally.

Preparing your personalized application:

Before deploying applications that take advantage of the features of Portal Personalization, certain features must be configured in order to work properly. The Feedback and LikeMinds components of Personalization both communicate with their databases using Java data sources. Before using either of these components, you must create resource references to the data sources in your project.

About this task

If you are using IBM Rational Application Developer to add resource references to the Feedback and LikeMinds data sources, complete the following steps:

Procedure

1. From the WAR application `WEB-INF/` directory, open the `web.xml` file. Click the **References** tab. Under the list of defined references, click **Add**.
2. In the Add Reference panel, select **Resource Reference** and click **Next**.
3. Complete the following fields in the Add Resource Reference panel and click **Finish**.
   - **Name:** `jdbc/feedback`
   - **Type:** `javax.sql.Datasource`
   - **Authentication:** Container
   - **Sharing scope:** Sharable
   - **Description:** This field is optional.
4. In the section marked **WebSphere Bindings**, enter the JNDI name of your Feedback Datasource.
5. Save the file.

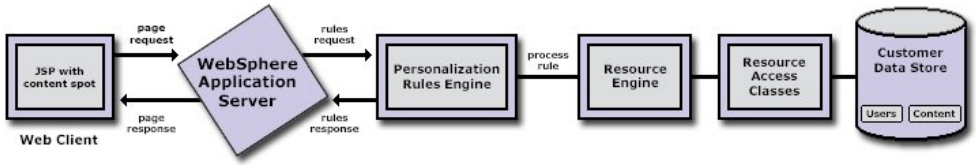
What to do next

Repeat this process for the LikeMinds data source. Change the name of the reference to jdbc/lmdbDS. All other settings remain the same.

Programming model:

Personalization builds on the same programming model used by WebSphere Application Server. Two components of that model are Java Server Pages (JSP) and business logic (JavaBeans and enterprise beans). JSP files enable you to effectively separate HTML coding (presentation) from business logic. You isolate your business logic in beans that the Web page author embeds in the JSP.

The term *developer* could refer to Web page authors or programmers. The developer does not need programming skills to author JSPs that access databases and reusable Java components. The author only needs to know the type of content that the bean, servlet, or other Java component adds to the page. The programmers create the reusable Java components and provide the Web page authors with the component names and properties. The database administrators provide the Web page programmers with the database access and table information.



The model expands three Web team roles:

- The business users

  The business users understand the business goals that the Web site or Web application must achieve. They understand the business conditions and business rules that must be factored into the Personalization solution. For example, the business users know the characteristics by which to categorize users for effective implementation of the business rules.

- The developer

  The developer understands Web site development and can use Rational Application Developer (or similar tools) to create and publish Web pages and sites. The developer works with the business user to determine where personalized content should appear and creates content spots of the appropriate type at the proper locations on the Web pages.

- The administrator

  This team member is the Web content expert and/or server administrator. The administrator knows what content already exists, where it is stored, the people who provide the content, how often the content is updated and other details. This person understands how to categorize content and helps the business users organize and target content for categories of users.

User and content models:

The first step in developing a Personalization solution is to analyze your business conditions and events to determine the users and content to be targeted. The business users and administrators are primarily responsible for this task.

After this analysis is completed, focus on the user, content and other data for your Personalization solution. The subsequent tasks include:

- Developing a user model and a content model

  The user model consists of the properties (attributes) of the user (that is, the typical Web site visitor), such as name and address. Similarly, the content model consists of properties of the Web content, such as title and author. Another term for model is *schema*.

  If you have already stored user data, some level of user model already exists. In such cases, the decision becomes whether to create a new model, or adapt the existing model to meet the requirements for your Personalization solution. You will also need to ensure that the user model and content model facilitate mapping users to content. For example, if you want to display news headlines that are of interest to the site visitor, the user model might include a property called `user_interests` that is a list of the topics in which the user is interested. The content model, for article content, could include a property `topics` that is a list of the subjects covered in the article. To support mapping a user interest to an article topic, you would ensure that there is an understandable relationship between possible values of `user_interests` and values of `topics`.
- Implementing the user and content models

  You must implement the user and content data models. If database tables matching the data models do not exist, implementing the data model involves creating and populating the data store for the user and content data. If the database tables exist, you might need to combine data from multiple tables, or add columns to create a user or content resource that matches the corresponding data model. Reusing existing tables might involve combining data from multiple tables to create a user or content resource.
- Implementing the Personalization APIs for accessing user, content and other resources

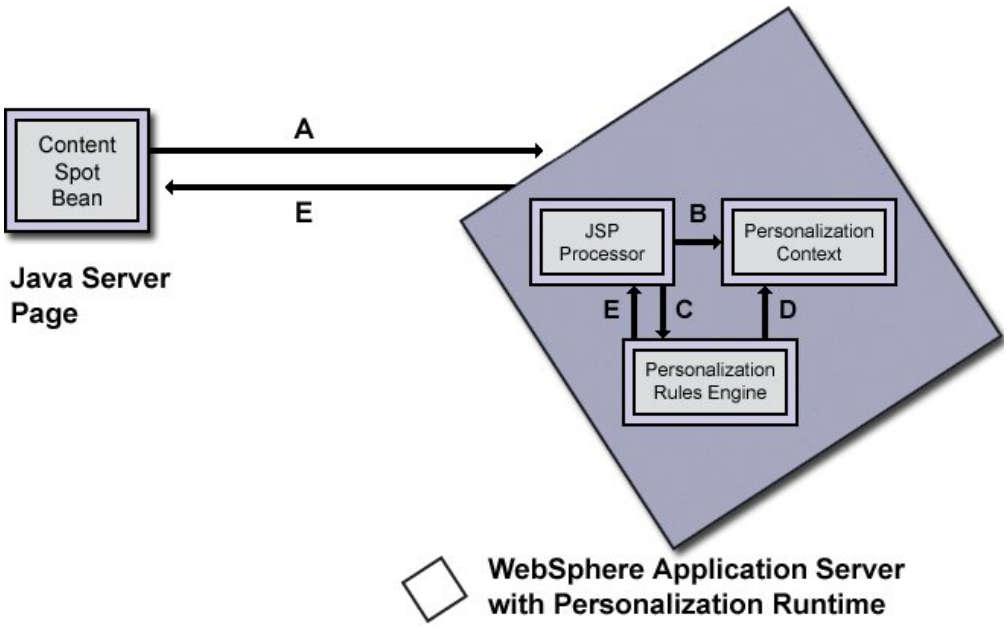
  After you implement the user and content data stores, you must enable the Personalization runtime environment to access the data stores. This task involves creating implementations of the Personalization APIs for accessing resources in the customer data store. You can either implement the `ResourceManager3` and `ResourceDomain3` interfaces manually or you can use the wizards provided with Rational Application Developer to generate resources for LDAP repositories. There are also implementations of the Portal user as a Personalization resource and a resource collection implementation for accessing the DB2 Content Manager.

How the rules engine works:

The rules engine processes and delivers the results of a rule execution to a content spot contained in a web page. The content spot is marked by a content spot bean, which is placed in either a JSP file or a servlet. The JSP file or servlet is then linked to the web page.

At run time, the content spot bean searches for the best rule to fill the spot with personalized data. The best rule to use depends on the campaigns that are set up to run. The rule that is mapped to the content spot in the campaign with the highest priority is run to personalize your web page.

Rule processing results in the return of a set of resources or a profiler. The returned resources or profiler can be used for generating a partial or entire web page. The figure illustrates how a rule is processed.



Step A: The process begins when a user requests a JavaServer Page (JSP) or servlet in which a content spot bean is embedded. The content spot bean contains the code to find and run the rule. When the web server receives the client request for the JSP or servlet, the web server passes the JSP or servlet request to the IBM WebSphere Application Server, which then starts its JSP or servlet processor.

The content spot bean can be embedded in the JSP by using any JSP editor. The following example code demonstrates how to embed and use a content spot bean in a JSP.

Note: IBM Rational Application Developer provides a visual JSP editor (Page Designer) that simplifies the development task and generates the JSP scriptlet coding for you.

The bean is embedded by using the JSP useBean tag. The HTTPServletRequest object is passed to the bean within the body of the useBean tag. The content spot bean properties are retrieved in the same manner as retrieving properties for any JavaBeans. The getRuleContent method of the contactsByLocation content spot bean determines the appropriate rule to run. This action is based on campaigns that have a rule that is mapped to the spot, runs the rule, and returns the results. The bean returns an array of Personnel objects.

Table 381. Example code that demonstrates embedding and by using a content spot bean in a JSP

```
<jsp:useBean id="contactsByLocation" class="GetContactsByLocation">
<% contactsByLocation.setRequest(request); %>
</jsp:useBean>
<%
try {
Personnel[] contentArray = contactsByLocation.getRuleContent();
Personnel theContent = contentArray[0]; // throws an exception if empty
%>
<TABLE border="1">
<TBODY> <TR>
<TD>Last Name</TD>
<TD>First Name</TD>
<TD>Role</TD>
<TD>Site</TD>
</TR>
<% for (int _i0 = 0; ; ) { %>
<TR>
<TD><%= theContent.getLastName() %></TD>
<TD><%= theContent.getFirstName() %></TD>
<TD><%= theContent.getRole() %></TD>
<TD><%= theContent.getSite() %></TD>
</TR>
<% _i0++;
try {
theContent = contentArray[_i0];
}
catch (java.lang.ArrayIndexOutOfBoundsException _e0) {
break;
}
} %>

</TBODY>
</TABLE>
<% }
catch (java.lang.ArrayIndexOutOfBoundsException _e0) {
} %>
```

Note: When you decide to add personalized content to a web page, all that is needed to develop the JSP is the content spot bean. The content spot can display personalized data of a single data type (for example, Personnel in this example). The web developer does not need to know where or how the content is retrieved only that personalized content of type Personnel is returned and it has a set of properties to be displayed.

Step B: When the embedded content spot bean is started, the processor passes the HTTP servlet request object to the content spot bean. The client request is used to initialize the RequestContext. The RequestContext provides access to the resources needed for rule processing. Those resources include collections, application objects, requests, and sessions. The RequestContext is applicable for the life of the HTTP request.

Step C: The Personalization rules engine find the appropriate rules in the IBM Java Content Repository and runs them

Step D: The rules engine processes the rule to obtain the results of the rules execution.

Step E: The rules engine returns the result of the rule execution to the content spot bean. The result can be a list of valid content from which a user can make a selection, a string, a Boolean value, or no results. The JSP scriptlet or servlet uses the rule result to derive specific web content for use in the generated web page. The web server forwards the page to the client.

Workload management:

Only one Portal Personalization engine can be installed on an application server. Consequently, there is one Personalization engine for each servlet engine. Personalization supports application server clustering, because each Personalization instance in the cluster shares the same IBM Java Content Repository. Therefore, each Personalization instance accesses the same customer data stores.

IBM WebSphere Application Server dynamic caching is used to cache resource instance and the results of queries used in rules. The dynamic cache shares expiration notification for the cache across clones in a cluster. Although Personalization uses the dynamic cache internally to cache the results of rules, it is also possible to use the WebSphere Application Server dynamic cache to cache the entire response from a JSP or servlet.

Note: Care must be taken when using Personalization and the dynamic cache of JSPs, servlets, or portlets. When using the dynamic cache to cache JSPs or servlets, the cache key must take into account all the inputs into any rules on that page. If rules on the page use an employee department attribute of a user resource, the cache key must be configured to contain this employee department attribute.

Using the Personalization APIs:

Personalization provides open APIs that enable the Personalization run-time environment and Rational Application Developer to access user and content data in customer data stores.

The Javadoc for Personalization APIs is available from the *PortalServer_root*/doc/Javadoc directory.

"Resource interface" on page 2410
The interface com.ibm.websphere.personalization.resources.Resource enables mapping your user model, content model, or other resource model to data in your customer data store. Get an overview of the methods defined by this interface that you must implement.

"APIs for multivalue properties" on page 2410
Use the com.ibm.websphere.personalization.resources.IMVResource interface to enable mapping multi-value properties. Use the com.ibm.websphere.personalization.resources.IMultiValueUtils interface for retrieving multi-value properties when those resources are in a database.

"General tips" on page 2411
View some general tips related to resource classloading when using the Personalization API.

"Personalization jar files that use public API" on page 2412
Learn about the .jar files that contain public APIs documented in Personalization JavaDoc information. Unless they are included in the JavaDoc documentation, classes are not public API even if they are included in these .jar files. When you use IBM Rational Application Developer and the Portal Personalization tools, these classes are added to your project classpath automatically and no further action is necessary.

Resource interface:

The interface com.ibm.websphere.personalization.resources.Resource enables mapping your user model, content model, or other resource model to data in your customer data store. Get an overview of the methods defined by this interface that you must implement.

Table 382. Explanation of methods used by com.ibm.websphere.personalization.resources.Resource

Method	Explanation
getId()	Returns the primary key or identifier for this resource. The primary key must be a string and unique within the resource collection. This method behaves in coordination with the findById method of the associated resource manager class such that the following method returns true: `manager.findById(resource.getId(), context).getId().equals(resource.getId())`
get(String name)	Returns the value of the specified dynamic property for this resource
keys()	Returns all (an Enumeration) of the dynamic property keys associated with this resource
put(String name, Object value)	Sets the specified dynamic property for this resource
remove(String name)	Removes the specified dynamic property

In addition to the methods listed in this table, your implementation must contain methods for setting and getting each fixed property in the data model. For example, if your user model includes a fixed property userName, you would define the methods getUserName() and setUserName().

Given an implementation of fixed properties, dynamic properties are optional. The get, keys, put, and remove methods may be implemented to perform no operation. If the content schema or resource attributes are known when the Java classes are developed, fixed properties are preferred. If the attributes of a resource are not determined until the resource is instantiated in the application server, dynamic properties are preferred. Dynamic and fixed properties may be used together in a single resource.

Rules support nested method calls. For example, a resource interface implementation could define a `user` object with a fixed property `employer` for which there is a fixed property `name`.

APIs for multivalue properties:

Use the com.ibm.websphere.personalization.resources.IMVResource interface to enable mapping multi-value properties. Use the com.ibm.websphere.personalization.resources.IMultiValueUtils interface for retrieving multi-value properties when those resources are in a database.

The com.ibm.websphere.personalization.resources.IMVResource interface extends the Resource interface and enables mapping multi-value properties. Implementing this interface is only required when an `IMultiValueUtils` implementation is used by the `ResourceManager3`. This is the default case for resources generated with the Portal tools, but may not be necessary for custom resources.

Table 383. Explanation of methods for mapping multi-value properties

Method	Explanation
setMultiValueUtils(IMultiValueUtils instance)	Saves the reference to the instance of MultiValueUtils. That reference is used when invoking the fillinMultiValueProperties method of the MultiValueUtils class. This method does not return output.
addMultiValuePropertyValue(String propertyname, Object propertyvalue)	Enables the `IMVResource` instance to add values for multi-value properties. This method does not return output.

The interface com.ibm.websphere.personalization.resources.IMultiValueUtils is a set of utilities for retrieving multi-value properties when those resources are in a database. This class supports mapping multi-value properties to the corresponding database tables.

Table 384. Explanation of methods for retrieving multi-value properties

Method	Explanation
convertSQLtoMultiValue(String query)	Converts the SQL query string for `ResourceManager3` classes that need to search on single value and multi-value properties. Returns an enumeration that contains the converted where clause followed by one or more elements that contain the table names that are involved in the query.
populateJoinedProperties(`IMVResource` theResource, RequestContext context)	Retrieves the value for all multi-value properties and calls the addMultiValuePropertyValue method of `IMVResource`. This method does not return output.
populateJoinedProperty(IMVResource theResource, String propertyName, RequestContext context)	Retrieves the value for all single value properties and calls the addMultiValuePropertyValue method of `IMVResource`. This method does not return output.
syncJoinedProperty(IMVResource theResource, String propertyName, List values, RequestContext context)	Populates the multi-value property into the resource repository.

General tips:

View some general tips related to resource classloading when using the Personalization API.

Resource classloading

- The recommended location for generated content spot classes and resource collection classes is in a WebSphere Application Server shared library. When a rule is executed, Personalization uses the class loader of the content spot to load any resource collections or application objects required for that rule. If you are using the default `com.ibm.websphere.personalization.ContentSpot` class to execute rules, then that class loader is the Personalization shared library class loader. In this scenario, your resource collections and application objects must be visible to the class loader of the `com.ibm.websphere.personalization.ContentSpot` class. With a generated

content spot class, you have the flexibility to place this class and any associated resource collections directly in a web module or portlet application, but they must also be available to the Personalization portlets. Whichever mechanism you use for content spots, the best way to ensure that your resource collection classes are available is to put them in a shared library on the application server. By default, Personalization is shipped with a shared library that may be used for resource collections. Place your resource collection classes in the *wp_profile_root*/PortalServer/pzn/collections directory and reload the server.

- In previous versions of Personalization, the user interface classes were loaded dynamically out of other web applications, out of databases, and out of other paths entered by the user. In the current version of Personalization, the content spot and resource collections must be on the classpath for the Personalization portlets. The best way to achieve this is through use of a shared library.
- If you use the Personalization resource wizard in Rational Application Developer to create the resource classes in a Web or portlet project, those generated classes will be deployed in the Web module and not be available to the Personalization portlets. You will have to put these classes on a shared library and make that shared library available to the Personalization portlets.
- Classes in a shared library are only reloaded when the application server is reloaded. Classes in a Web module are reloaded when the Web module is reloaded.
- Ensure the global uniqueness of the class names by using package names.
- In a cluster of application servers, be sure to copy your content spot and resource classes to each application server in the cluster.

Personalization jar files that use public API:

Learn about the .jar files that contain public APIs documented in Personalization JavaDoc information. Unless they are included in the JavaDoc documentation, classes are not public API even if they are included in these .jar files. When you use IBM Rational Application Developer and the Portal Personalization tools, these classes are added to your project classpath automatically and no further action is necessary.

If you are compiling your application outside of Rational Application Developer and you are using Personalization APIs, you may need to add the following classes to your classpath when compiling your application. These .jar files are located in *PortalServer_root*/pzn/prereq.pzn/lib:

- pznquery.jar
- pznresources.jar
- pznruntime.jar
- pznwpsruntime.jar

Generic query framework:

The generic query framework enables resource collection developers to convert a property-based query object into a language specific executable query string. It contains query component classes, and builder and callback interfaces.

- Query framework classes, such as the Query class, Predicate class, and Condition class provide an object representation of a query.
- Builder and callback interfaces together facilitate a query string generation mechanism that delegates operations to domain specific callbacks for property-to-attribute resolution and query string syntax conversion.

"Using the Generic Query Framework"
The resource engine constructs a generic query object and passes it to domain developers through the ResourceDomain3 interface method findResourceByQuery(). Get an overview of how this query object can be converted into a meaningful domain query string.

Using the Generic Query Framework:

The resource engine constructs a generic query object and passes it to domain developers through the ResourceDomain3 interface method findResourceByQuery(). Get an overview of how this query object can be converted into a meaningful domain query string.

The developer can take one of the following approaches to convert this query object into a meaningful domain query string.

1. Walk through the query object.

   Note: The com.ibm.websphere.query.base.Query class contains query components that the developer can cover to generate domain specific query string. For detailed information on query hierarchy and components, see the Portal Personalization API documentation.

2. Use a system provided builder callback.

   There are nine builder callbacks:

   a. Microsoft SQL Server Enterprise Edition: (com.ibm.websphere.query.callbacks.SqlSelectQueryCallback)

      Note: Although the generic SQL callback can be used for most SQL Server databases, there are minor differences in SQL syntax and availability of functions which require specific subclasses for some databases.

   b. IBM DB2 Universal Database Enterprise Server Edition: (com.ibm.websphere.query.callbacks.DB2SqlSelectQueryCallback)

   c. IBM DB2 Universal Database for z/OS: (com.ibm.websphere.query.callbacks.DB2390SqlSelectQueryCallback)

   d. IBM DB2 Universal Database for i: (com.ibm.websphere.query.callbacks.DB2400SqlSelectQueryCallback)

   e. Apache Derby: (com.ibm.websphere.query.callbacks.DerbySqlSelectQueryCallback)

   f. Oracle Enterprise Edition: (com.ibm.websphere.query.callbacks.OracleSelectQueryCallback)

   g. LDAP: (com.ibm.websphere.query.callbacks.LdapSelectQueryCallback)

      Note: The LDAP callback supports a set of function common to many LDAP repositories. Users may subclass this callback to support more advanced vendor specific functions.

   Property resolution and query syntax conversion are handled in the callbacks. The developer can prepare a property mapping hash table and use it with one of the previous callbacks to build the executable query string. Here is the sample code for SQL query string generation:

   ```
   String s=q.buildString(new SqlSelectQueryCallback(h));
   ```

   where *q* is a query object, and *h* is a property mapping hash table.

   For detailed information on these builder callbacks, see the Personalization API documentation.

3. Develop and use a domain specific builder callback.

   If the system provided builder callbacks do not satisfy resource domain requirements, a domain specific builder callback can be created and used as long as it implements ISelectQueryCallback. The developer can decide the mechanisms to interpret properties and derive the proper query syntax in their own callbacks. The code would look like:

   ```
   String s=q.buildString(new MySelectQueryCallback(myParameter));
   ```

   where *q* is a query object, and *MySelectQueryCallback* is the custom builder callback that takes *myParameter* as parameter.

Request context interface:

This is the interface used to access various attributes for rules. For HTTP contexts, it provides access to the HttpRequest and HttpSession attributes. For non-HTTP contexts, it provides the same interface to a surrogate for the request and session.

The request context, and any request values accessed via the request context, are only valid for the life of the request.

The Request Context string is used in caching lookups. The lookup is created by using a user-specified string combined with query values as the lookup key. The user-specified string should uniquely represent the current Request Context. This key is stored under ibm.wcp.cache.user.key as a request attribute or as a session attribute with the request attribute taking precedence.

Accessing the Request Context

The Request Context provides the Personalization rules engine with the data and environmental information needed for rules processing. In other words, the Request Context contains all the input to execute Personalization content spots and rules. This includes simple inputs like request and session data, and more advanced input like the user object.

You can access the Request Context from a content spot by first using the setRequest method of the content spot to back the content spot with a request, and then by calling getContext to retrieve the context. You can also use the Request Context to call directly into the ResourceDomain3 and ResourceManager3 APIs.

The Request Context allows you to retrieve session, request, portlet attribute, date, cookie, and other data and environmental information from the resource layer.

The Request Context includes:

- Session

  The session information identifies the HttpSession object that is associated with the current user.
- Request date

  This request date is the date the HTTP request was received. This information supports rules that have date-dependent actions.

Since Personalization uses the Request Context to contain all the rule input, the Request Context must be set onto the content spot prior to rule execution. The code with the content spot's useBean tag must be similar to:

```
<jsp:useBean id="gold_promo_bean"
class="yourco.goldpromo.BannerSpot" />
<% gold_promo_bean.setRequest (request); %>
```

In the previous section, the jsp:useBean tag constructs the yourco.goldpromo.BannerSpot class and stores an instance of that class into the local variable gold_promo_bean. The next line calls setRequest to put the HttpServletRequest or PortletRequest onto the newly constructed content spot bean. The content spot then implicitly constructs a Request Context which is backed by the given HttpServletRequest or PortletRequest. This Request Context then provides access to that request's parameters and attributes and attributes of the session through the com.ibm.websphere.personalization.RequestContext interface.

In some cases, it may be useful to call into a content spot without having access to an HttpServletRequest or PortletRequest. The interface com.ibm.websphere.personalization.PznRequestObjectInterface can be used in these situations. A implementation of this class called com.ibm.websphere.personalization.PznRequestObjectImplementor is provided for convenience.

Sample Personalization resources XML file:

Use the sample file `ExportFromServlet.xml` as a reference for coding other Personalization actions.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE ibm-websphere-personalization>
   <ResourceCollection action="create" name="News">
     <ResourceType>Content</ResourceType>
     <ResourceManagerClass>com.mycompany.personalization.ContentResourceManager</ResourceManagerClass>

     <ResourceDomainClass>com.mycompany.personalization.ContentResourceDomain</ResourceDomainClass>
     <ResourceClass>com.mycompany.personalization.Content</ResourceClass>
  </ResourceCollection>
  <SecurityMappings>
    <collection EnableSecurity="false" name="News"></collection>
  </SecurityMappings>
</ibm-websphere-personalization>
```

Content spot exits:

Content spot exits provide the ability to alter the default flow of processing of content spots, making it possible in the runtime environment to override the rule to process, the current user, and the results of rule processing. Instances of the same exit class are instantiated for all content spots. The exit class interface name is public interface RuleExit.

The content spot exit class can provide the following actions:

1. Access request and session information
2. Set information in the personalization context, including changing the user
3. Get the campaign name
4. Get the rule name to be executed for campaign
5. Override and specify the rule to be executed
6. Bypass rule processing; no rule is processed.

After rule processing, the content spot exit class can provide the following actions:

1. Add result items

2. Remove result items
3. Completely replace result items

RuleExit methods:

void aboutToExecuteRule(RuleTrigger contentSpot, RequestContext requestContext)
Get the results of the rule execution.

Object[] getFilteredResults(RuleTrigger contentSpot, RequestContext requestContext, Object[] originalResults)
Exit to allow for changing the results of the rule execution. originalResults: the original result array is supplied to the exit. filteredResults: the exit should return the original array if no changes are needed

RuleTrigger methods:

String getCampaignName()
Get the campaign name for the content spot.

String getRuleName()
Get the name of the rule to be executed in this spot. Can return null if the name is not yet established.

void setRuleName(String ruleName)
Set the name of the rule to be executed in this spot. If the name is set to null, no rule will be executed.

void setRuleExit(RuleExit ruleExit instance)
Set the rule exit for this particular content spot instance.

The `RuleTrigger` also supports a static method that contains an instance of `RuleExit` that is set up at initialization time. You can implement `setRuleExit(RuleExit)` on a per-content spot basis to override only the content spots that you specify.

The class name of the default `RuleExit` implementor is read from the PersonalizationService.properties file.

Example Usage Scenario

1. Specify a `RuleExit` class in PersonalizationService.properties, as shown:
   ```
   rulesEngine.defaultRuleExit=com.ibm.websphere.personalization.RuleExitSample
    ...
   ```
2. At startup time, the Personalization run-time environment creates an instance of that class (`aRuleExit` in this scenario) and caches it in a private `RuleTrigger` static method
3. As each content spot is triggered, the Personalization rule engine determines the rule name to be used
4. `aRuleExit.aboutToExecuteRule()` is invoked, passing the spot and the request context
5. `aRuleExit` has several options:
   - Access request context information (includes http request and session)
   - Get the campaign and rule names
   - Specify a request user ID
   - Change the rule name to be executed
   - Bypass the rule by setting the rule name to null

6. The rule is executed
7. `aRuleExit.getFilteredResults()` is invoked; the rule exit modifies results as required, then returns the updated set
8. The updated results are stored

Resource cache:

Personalization uses the WebSphere Application Server Dynamic Cache service to cache the results of select rules and to cache the rules themselves in a DistributedMap. When publishing, importing or saving rules, the cache is flushed automatically to ensure that the site is current.

Configuration settings available for caching rules

The `PersonalizationService.properties` file contains cache control settings. The following table summarizes the configuration settings available for caching.

Table 385. PersonalizationService.properties properties and descriptions. Each PersonalizationService property is listed by name and described.

Property	Description
rulesEngine.cache.enabled	Global setting to disable all caching. Setting to *false* will override any collections that are individually set to be cached.
rulesEngine.cache.jndiName	The Dynamic Cache DistributedMap to use, as specified by its JNDI name.
rulesEngine.cache.maxEnumSize	Although the DistributedMap may be configured in the WebSphere Integrated Solutions Console to limit the maximum number of entries in the cache, the collection of resources that is returned for each rule is counted as one entry in the cache. When a rule returns a very large number of results, caching the results may result in extra memory overhead. Therefore, when the result set of a rule exceeds the number specified in this property, it will not be cached.
rulesEngine.cache.timeout	The number of seconds before an entry in the cache expires.
rulesEngine.cache.priority	The priority of a cache entry, relative to other entries in the same DistributedMap.
rulesEngine.cache.enabled.*collectionName*	The same as **rulesEngine.cache.enabled**, except that this property applies to a specific resource collection. The variable *collectionName* is the fully qualified path to the resource collection in Personalization. For instance, the *GeneralNews* resource collection that is included in the sample folder called `PznDemo` has the *collectionName* of `/PznDemo/`*GeneralNews*.
rulesEngine.cache.jndiName.*collectionName*	The same as **rulesEngine.cache.jndiName**, except that this property applies to a specific resource collection.

Table 385. PersonalizationService.properties properties and descriptions (continued). Each PersonalizationService property is listed by name and described.

Property	Description
rulesEngine.cache.maxEnumSize.*collectionName*	The same as **rulesEngine.cache.maxEnumSize**, except that this property applies to a specific resource collection.
rulesEngine.cache.timeout.*collectionName*	The same as **rulesEngine.cache.timeout**, except that this property applies to a specific resource collection.
rulesEngine.cache.priority.*collectionName*	The same as **rulesEngine.cache.priority**, except that this property applies to a specific resource collection.

Caching business rules for resources

You might need to programmatically flush the Personalization cache, for example when a resource is updated outside of Personalization rules through some other application. A programming interface is provided to flush the cache. Since the timeout interval for the cache can be specified in the properties file, in many cases it may be adequate to wait for the cache timeout before updates are seen. The class com.ibm.websphere.personalization.resources.cache.CacheManager can be used to invalidate the cache for a particular resource, a particular resource collection, or the entire cache. Personalization uses this class internally to flush the cache when updates occur.

Flushing the cache for a particular resource may require that all cached queries be flushed. Flushing the cache on a collection may flush the cache for all collections using the same dynamic cache map. When the application frequently flushes the cache for a particular collection, isolating that collection in its own cache map through the use of the **ruleEngine.cache.jndiName**.*resourceCollectionName* property will result in better cache utilization.

Caching occurs before any rule exits are called.

For more information about using the DistributedMap and DistributedObjectCache interfaces for the dynamic cache, refer to the IBM WebSphere Application Server Information Center.

Related tasks:

Using the DistributedMap and DistributedObjectCache interfaces for the dynamic cache

Programmatically starting rules:

All types of rules can be accessed programmatically within a Java application. For example, a profiler can be used to determine the behavior that an application must exhibit depending on the current user, or an action can return content to your application for further processing before the content is displayed. Rules are mapped to content spots, and because a content spot is an implementation of a JavaBeans, it can be programmatically declared and implemented.

To programmatically start a rule, follow these steps:

1. Instantiate the bean. If the class name of your content spot was ProfilerSpot, you would instantiate the bean by using one of the following options:

   ```
   com.ibm.websphere.personalization.ContentSpot contentSpot =
     new com.ibm.websphere.personalization.ContentSpot("ProfilerSpot");
   ```

   or

   ```
   ProfilerSpot contentSpot = new ProfilerSpot();
   ```

2. Call the method `setRequest()` and pass the HttpServletRequest object or an object that implements PznRequestObjectInterface. This action makes the current information that is stored in the request object to be known to the Personalization runtime Engine and the Resource Engine.

   ```
   contentSpot.setRequest(request);
   ```

3. Trigger the rule and, if applicable, get the content from the rule. Use one of the following methods, depending on the type of rule:
   - Select content actions, bindings, and recommendation actions are mapped to content spots and return content from a resource.
     - `getRuleContent()` - returns an array of results
     - `getRuleContent(int which)` - returns the resource at the index
   - Profilers are mapped to empty content spots, which do not declare a return type.
     - `boolean isProfiledAs(string value)` - returns true or false depending on whether the string passed to the method matches a profile that is given to the user
     - `String getProfile(integer value)` - returns the profile in the location that is specified by the integer passed

       A user can have more than one profile. For example, a user might fit profiles that are named "young," "hip," and "sporty"; when the user is looking for the profile at location 0, "young" is returned.
     - `String[] getProfiles()` - returns an array of profiles.

       For example, if a user is in profiles "young," "hip," and "sporty," this method returns an array of the profiles.
     - `boolean isProfiledAsAll(String[] profiles)` - returns True or False depending on whether all the profiles match all the profiles in the list of profiles that are passed to the method
     - `boolean isProfiledAsAny(String[] profiles)` - returns True or False depending on whether the profile is in the list of profiles that are passed to the method
   - Update actions and email actions are mapped to empty content spots.
     a. `trigger()` - runs the rule

Rule Exception Handling in the run-time environment:

The default method of error or exception handling within the Personalization run-time environment is for the engine to print out a trace error to the application server's stdout log. Using an exception handling utility, it is possible to specify the type of output (error message or stacktrace), the output log file to use (stdout or stderr), and whether the exception should be rethrown to the JSP. The additional exception handling capabilities can be set on a per-request basis or globally.

Per-request

To set the scheme on a per-request basis, include the following code within a content spot on a JSP:

```
request.setAttribute(RuntimeUtils.PZN_RUNTIME_EXCEPTION_HANDLING_KEY,
RuntimeUtils.HANDLING_SCHEME);
```

Where HANDLING_SCHEME is one of the options specified for the setRuntimeExceptionHandlingScheme method described in the following section.

This method will only change the exception handling scheme for rules running on that JSP.

Global

To set the same scheme for every rule running on a server, execute the following code:

```
RuntimeUtils.setRuntimeExceptionHandlingScheme(RuntimeUtils.HANDLING_SCHEME);
```

To reset the global exception handling scheme, call:

```
RuntimeUtils.resetRuntimeExceptionHandlingScheme();
```

Note: Restarting the application server will automatically reset the exception handling scheme.

You can also change the global behavior of the server by changing the following value in PersonalizationService.properties:

```
rulesEngine.exceptionHandling=logMessage_stdout
```

Note: To make any changes effective, you must restart the server.

RuntimeUtils methods

public static void setRuntimeExceptionHandlingScheme(String value)

Sets the exceptions handling scheme to the specified value. Options for value include:

- RuntimeUtils.IGNORE
- RuntimeUtils.LOG_MESSAGE_STDOUT
- RuntimeUtils.LOG_MESSAGE_STDERR
- RuntimeUtils.LOG_MESSAGE_STDOUT_AND_RETHROW
- RuntimeUtils.LOG_MESSAGE_STDERR_AND_RETHROW
- RuntimeUtils.LOG_STACKTRACE_STDOUT
- RuntimeUtils.LOG_STACKTRACE_STDERR
- RuntimeUtils.LOG_STACKTRACE_STDOUT_AND_RETHROW
- RuntimeUtils.LOG_STACKTRACE_STDERR_AND_RETHROW
- RuntimeUtils.LOG_MESSAGE_AND_STACKTRACE_STDOUT
- RuntimeUtils.LOG_MESSAGE_AND_STACKTRACE_STDERR
- RuntimeUtils.LOG_MESSAGE_AND_STACKTRACE_STDOUT_AND_RETHROW
- RuntimeUtils.LOG_MESSAGE_AND_STACKTRACE_STDERR_AND_RETHROW
- RuntimeUtils.RETHROW_EXCEPTION

Note: Settings with RETHROW will pass the exception to the screen and are recommended for use within a testing environment only.

public static void resetRuntimeExceptionHandlingScheme()
Resets the current exception handling scheme to LOG_MESSAGE_STDOUT.

public static String getRuntimeExceptionHandlingScheme()
Returns the current exception handling scheme.

Exception Handling Process

When an exception occurs, if the code can continue, it will do so while logging the exception. If not, the exception is wrapped in a PersonalizationException which is passed to RuleTrigger. RuleTrigger looks for a request override for the exception handling scheme, processes the scheme, and returns to the JSP. Any subclasses of Throwable might be wrapped in a PersonalizationException.

Tracing

To trace the runtime classes, enable trace for `com.ibm.websphere.personalization.*=all`. To trace the authoring portlet classes, enable trace for `com.ibm.wps.caf.*=all`.

Social Media Publisher

The Social Media Publisher for Web Content Manager is an extension to Web Content Manager that allows businesses to promote their web content on social networks, and provide some basic statistics about the promoted content.

Requirements

- WebSphere Portal version 8.0.0.1 or later, or Web Content Manager version 8.0.0.1 or later.
- A social media service:
  - IBM Connections version 3.0.2, version 4.0 CF2, version 4.5, or version 5.0.
  - Facebook (API level as at September 2012)
  - LinkedIn (API level as at September 2012)
  - Twitter API level v1.1

"Overview of the Social Media Publisher" on page 2422
The Social Media Publisher for Web Content Manager is used to manage social media posts for published content items and, if configured, draft content items.

"Installing the Social Media Publisher" on page 2423
Follow these steps to install the Social Media Publisher on your authoring server, and then on any servers that subscribe from your authoring server.

"Social network configuration" on page 2425
Your social network configuration settings are stored in separate content items based on the **Social Network Configuration** authoring template that is supplied with the Social Media Publisher.

"The social network information table" on page 2437
When you create a content item by using a content authoring template that is mapped to a social network configuration document, an element that is named **Social Network Information** is displayed on the content item.

"Social Media Workflow Actions" on page 2437
A set of social media workflow actions are supplied to with the Social Media Publisher and are found in the **Social Configuration** library. These actions can be added to workflows to run social media actions as part of a workflow.

Overview of the Social Media Publisher

The Social Media Publisher for Web Content Manager is used to manage social media posts for published content items and, if configured, draft content items.

- The Social Media Publisher supports IBM Connections, Facebook, LinkedIn and Twitter.
- The Social Media Publisher supports both manual and automatic social media posts, and also manages the life-cycle of social media posts.
- Social media posts are only applied to published content and, if configured, draft content items.
- The Social Media Publisher does not include or manage the social media "badges" you can add to websites to allow end-users to link or share web content to their social network followers.

Table 386. Messages supported by each social network

Social Network	Message type	Statistics supported	Message URL addressable	Supports end user tracking	Supports untrack	Supports delete
IBM Connections	Blog post	Like, comment count	Yes	No	Yes	Yes
IBM Connections	Message board post	Comment count	Yes	No	Yes	Yes
IBM Connections	Wiki post	Like, comment count	Yes	No	Yes	Yes
Facebook	Page wall post	Like, comment count	Yes	Yes	Yes	Yes
Facebook	Profile wall post	Like, comment count	Yes	Yes	Yes	Yes
LinkedIn	Network update	None	No	No	Yes	No
LinkedIn	Share	None	No	No	Yes	No

Table 386. Messages supported by each social network (continued)

Social Network	Message type	Statistics supported	Message URL addressable	Supports end user tracking	Supports untrack	Supports delete
Twitter	Tweet	Reshare count	Yes	Yes	Yes	Yes

Note: Page notes have been deprecated by Facebook.

Before you begin

You should become familiar with your social media of choice and be aware of limitations such as:

- throttle limits
- maximum posts per timeframe

Installing the Social Media Publisher

Follow these steps to install the Social Media Publisher on your authoring server, and then on any servers that subscribe from your authoring server.

Before you begin

Note: If you migrated from a WebSphere Portal version 7.0 or Web Content Manager version 7.0 server that was already using the Social Media Publisher, you also need to run these steps on the migrated system to enable the Social Media Publisher. All your data from your version 7.0 system is maintained.

Procedure

1. Ensure that you enable the following options correctly:
   a. That the **WasPassword** and **PortalAdminPwd** passwords are set in the `wkplc.properties` file.
   b. That the **jcr.DbPassword** password is set in `wkplc_dbdomain.properties` file.
   c. That a proxy server is configured if your server is behind a proxy.
2. Edit *wp_profile_root*`/PortalServer/wcm/social/smp.properties` file and to ensure that the configuration is correct for the current server or cluster node.

   Note: The parameter settings in `smp.properties` are case-sensitive.
3. Run the following registration command from the *wp_profile_root*`/ConfigEngine` directory:

   Windows
   ```
   ConfigEngine.bat register-wcm-social
   ```
   Linux
   ```
   ./ConfigEngine.sh register-wcm-social
   ```
   IBM i
   ```
   ConfigEngine.sh register-wcm-social
   ```
4. Run the following deployment command from the *wp_profile_root*`/ConfigEngine` directory:

   Windows
   ```
   ConfigEngine.bat deploy-wcm-social
   ```
   Linux
   ```
   ./ConfigEngine.sh deploy-wcm-social
   ```

IBM i `ConfigEngine.sh deploy-wcm-social`

5. If you are publishing to an IBM Connections server, then import the SSL certificate from the IBM Connections server into the WebSphere Portal server. See the WebSphere Application Server documentation for instructions on importing certificates.

   Note: If your server is located behind a firewall, you might also need to import certificates for Facebook, LinkedIn, and Twitter. For more information, see "Social media certificates" on page 2440 for further information.
6. Restart the server.
7. Add the **Social Configuration** library to all syndication relationships. Repeat these steps for each subscriber server in your environment. The settings that are defined for **Shared Social database properties** in the `smp.properties` file must be the same as the settings defined on the syndicator server.

   Note: The parameter settings in `smp.properties` are case-sensitive.

   When you install on a server other than the authoring server, if your JCR database user name or password is different from the authoring server, then run the following command from the *wp_profile_root*`/ConfigEngine` directory:

   Windows
   ```
   ConfigEngine.bat action-set-wcm-social-datasource-credentials
   -DWcmSocialDSNewUserName=AUTH_DB_USERNAME
   -DWcmSocialDSNewPassword=AUTH_DB_PASSWORD
   ```

   Linux
   ```
   ./ConfigEngine.sh action-set-wcm-social-datasource-credentials
   -DWcmSocialDSNewUserName=AUTH_DB_USERNAME
   -DWcmSocialDSNewPassword=AUTH_DB_PASSWORD
   ```

   IBM i
   ```
   ConfigEngine.sh action-set-wcm-social-datasource-credentials
   -DWcmSocialDSNewUserName=AUTH_DB_USERNAME
   -DWcmSocialDSNewPassword=AUTH_DB_PASSWORD
   ```
8. If your server contains virtual portals, you must also run the following task for each virtual portal on your server:

   Windows
   ```
   ConfigEngine.bat import-wcm-social-data
   -DVirtualPortalHostName=VirtualPortalHostName
   -DVirtualPortalContext=VirtualPortalContext
   ```

   Linux
   ```
   ./ConfigEngine.sh import-wcm-social-data
   -DVirtualPortalHostName=VirtualPortalHostName
   -DVirtualPortalContext=VirtualPortalContext
   ```

   IBM i
   ```
   ConfigEngine.sh import-wcm-social-data
   -DVirtualPortalHostName=VirtualPortalHostName
   -DVirtualPortalContext=VirtualPortalContext
   ```
9. Restart WebSphere Portal.
10. Repeat these steps on every server and cluster node.

Additional installation and configuration

- When you import or migrate a Social Media Publisher library, you must also migrate credential vault data by using the XML configuration interface. See the WebSphere Portal product documentation for further details.
- Two credential vault slots are required for Social Media Publisher configuration:

- A credential vault slot that is called **socialPostUser** is required for post workflow actions. See "Social Media Workflow Actions" on page 2437. The **socialPostUser** slot is not displayed in the list of available vault slots when you create a Social Network Configuration document.
- You must also set up a credential vault for each social network. This action is selected when you create a social network configuration document. See "Creating a social network configuration document."

Both of these credential vault slots must be shared credential vault slots and store the admin user login information of your social network.

- Anonymous access to web content libraries, content items, and images are required for posting to external social networks. Single signon can also be enabled for IBM Connections.
- The Social Media Publisher uses the built-in JMS function of Web Content Manager to monitor and respond to events that are occurring within a system. When the Social Media Publisher is running within a cluster, each server needs to be configured to match the WebSphere Application Server JMS policy type in effect. The `jms.processRemoteMsgs global setting` must be set to either true or false depending on the WebSphere Application Server JMS policy type. For more information, see "Global configuration settings" on page 2433 for further information.
- When you install the Social Media Publisher into a cluster that has a web-server in front of it, add the following Web Content Manager configuration setting. The settings allow the Social Media Publisher to make loop back requests to the current server:

  ```
  wcm.local.host.port=LOCAL_PORTAL_PORT
  ```

  Use the WebSphere Integrated Solutions Console to add this setting to the **WCM WCMConfigService** service. `LOCAL_PORTAL_PORT` is the local WebSphere Application Server port for WebSphere Portal. The default port is 10039.

Social network configuration

Your social network configuration settings are stored in separate content items based on the **Social Network Configuration** authoring template that is supplied with the Social Media Publisher.

"Creating a social network configuration document"
To create a social network configuration document, you create a content item that uses the **Social Network Configuration** authoring template.

"Global configuration settings" on page 2433
Although the Social Media Publisher uses social network configuration documents to specify most of its configuration properties, it also uses a global settings document for configuring server-wide options.

"Adding the social information table to an authoring template" on page 2437
If the **Automatically add social information element** setting on a social network configuration document is disabled, you must manually add the social information element to any authoring template you want to use the social media publisher on.

Creating a social network configuration document:

To create a social network configuration document, you create a content item that uses the **Social Network Configuration** authoring template.

About this task

The configuration settings for each social network you want to post to are stored in separate content items. These content items are never displayed on a website and are only used to store configuration settings. To create a new social network configuration document:

Procedure

1. Create a content item that uses the **Social Network Configuration** authoring template. This template is in the **Social Configuration** library.
2. Type a name and display title.
3. Enter the name of your production domain in the **Site Domain** field. For example: `www.ibm.com`
   - If this site is a virtual portal, then the site domain must be either the virtual portal host name or a public domain name that resolves to the virtual portal host name.
   - If your virtual portal does not define a virtual portal host name, then the site domain field can also be set by using the following format: `host/portal_context/vp_url_context`. For example: `localhost/wps/portal/myvp`
   - When you post content to Facebook, LinkedIn and Twitter, the site domain must be internet accessible, otherwise your social media posts might not function correctly.
   - If a web server is not being used, then the domain must include the portal port. For example: `myserver:10039`.
   - If the domain is not accessible by using port 80, then the domain must also include the HTTP port.
4. In the **Social Network Information** section, select a social network and enter configuration settings for the chosen network.
5. When the authentication settings are specified, click the **Authorize** button.
6. Map the social network configuration document to one or more content authoring templates by clicking **Add** under the **Mapped Authoring Template** section.
   - These templates should be authoring templates that are associated with content types you would like to post to social networks.
   - You can map multiple social network configuration documents to the one authoring template, and multiple authoring templates to the one social network configuration document.
   - Select **Automatically add social information element** to automatically add the **Social Information** element to each new content item created by using a mapped authoring template.
7. Specify extra settings:

   Display social action buttons
   Select which social action buttons are displayed in the social media table.

   Action to perform on document delete
   When an item is deleted, you can choose to either delete the item only, or delete the item and also delete any social media posts related to that item.

   Configuration enabled
   This is enabled by default. If disabled, social media publishing is

stopped for this social media configuration, and this social media configuration is hidden from the social information table in content items, and hidden from selection dialogs in social media workflow actions.

Enable Posting for draft documents
If enabled, social media posts are created for draft items and published items. This setting is disabled by default. If enabled, social media posts can be sent from draft items, either by using the social media table or social media workflow actions, in the same way as published items.

Show enabled social actions in read mode
When enabled, the functions available in the social information table on content forms can be used in both read and edit mode.

Link Mode
This option determines the type of links that are used in social media posts depending on whether your content is published in a portlet, or servlet.

Language
This option is used to select the language that is used to render social media posts.

"Defining social network settings for IBM Connections"
Complete these steps to set up a social network connection for IBM Connections.

"Configuration settings for Facebook" on page 2428
Complete these configuration parameters to setup a social network connection for Facebook.

"Defining social network settings for LinkedIn" on page 2430
Complete these configuration parameters to set up a social network connection for LinkedIn.

"Defining social network settings for Twitter" on page 2432
Complete these configuration parameters to set up a social network connection for Twitter.

Defining social network settings for IBM Connections:

Complete these steps to set up a social network connection for IBM Connections.

Before you begin

Before you begin you must have already created a new credential vault from the WebSphere Portal administration view where:

- Your IBM Connections user name is specified as the shared user ID.
- Your IBM Connections password is specified as the shared password.

About this task

When you create a social network configuration document for IBM Connections:

Procedure

1. Select **IBM Connections** as the social network.
2. Define authentication settings for the social network:
   a. Select the **Credential Vault** that contains your IBM Connections account credentials.

b. Type the name of the Connections server to connect to.
c. Click **Authorize** to bind the IBM Connections account credentials to this configuration.

3. Select a message type:

Blog post:

a. Select the name of the Blog.
b. Enter a default name and message template for Blog posts. A predefined message is included as a guide. For example:

Name: This is posted as text.

```
[Property field="title" context="current" type="content" format="length:100"]
```

Message:
This is posted as HTML.

```
[Element context="current" type="content" key="Message" format="length:100"]
See more: [URLCmpnt context="current" type="content" mode="standalone"
start="<a href='" end="'>Link</a>"]
```

Message board post:

a. Enter a default message template for message board posts. A predefined message is included as a guide. For example:

Message:
This is posted as text.

```
[Element context="current" type="content" key="Message" format="length:100"]
See more: [URLCmpnt context="current" type="content" mode="standalone"]
```

Wiki post:

a. Select the name of the Wiki.
b. Enter a default name, and message template for Wiki posts. A predefined message is included as a guide. For example:

Name: This is posted as text.

```
[Property field="title" context="current" type="content" format="length:100"]
```

Message:
This is posted as HTML.

```
[Element context="current" type="content" key="Message" format="length:100"]
See more: [URLCmpnt context="current" type="content" mode="standalone"
start="<a href='" end="'>Link</a>"]
```

Configuration settings for Facebook:

Complete these configuration parameters to setup a social network connection for Facebook.

Before you begin

Before you begin you must register an application in order to obtain a username and password that grants access to the Facebook API. To register a new application that the WCM Social Media Publisher can use:

1. Go to https://developers.facebook.com/apps
2. Click **Create new App**.
3. A form will open. Enter values for the following parameters:

Display Name:
This is the name of your application that will be displayed at the end of each post. For example, your company name.

Namespace:
This is an optional name that is used to access Facebook Insights information. It is recommended that a unique name is used for each application you setup.

Category:
Select a category that matches your business type. If unsure select **Apps for Pages**.

4. Click **Create App**.
5. A configuration screen for your application is displayed. Select the **Settings** option and enter the following:

   App Domain:
   Enter a comma separated list of domains or server addresses that you want to use to bind this application to individual Facebook accounts. You will only be able to authorize against this application from one of the domains listed here.
6. Click the **Add Platform** button and select **WebSite**.
7. Set the **Site URL** and **Mobile Site URL** fields to your website. For example: http://www.ibm.com
8. Click **Save Changes**.
9. The **App ID/API Key** and **App Secret Key** are now displayed.

You must then create a new credential vault from the WebSphere Portal administration view where:

- The **App ID/API Key** is specified as the shared user ID
- The **App Secret Key** is specified as the shared password.

About this task

When you create a social network configuration document for Facebook:

Procedure

1. Select **Facebook** as the social network.
2. Define authentication settings for the social network:
   a. Select the **Credential Vault** that contains your Facebook application credentials.
   b. Click **Authorize** to bind the credentials to a specific social network account.
3. Select a message type:

   Note: It is not recommended to reference rich text elements or HTML elements in any fields that do not support HTML.

   Page wall post:
   a. Select a page.
   b. Enter a default name, description, image, caption, and message template. Predefined tags for the image, caption and message are included as a guide. For example:

      Name: This is posted as text.

      ```
      [Property field="title" context="current" type="content" format="length:100"]
      ```

      Description:
      This is posted as text.

```
[Property field="description" context="current" type="content"
format="length:100"]
```

Image:
This is posted as a URL.
```
[Element context="current" type="content" key="Image" format="URL"]
```

Caption
This is posted as text.
```
[Element context="current" type="content" key="Caption"]
```

Message
This is posted as text.
```
[Element context="current" type="content" key="Message" format="length:100"]
```

Profile wall post:

a. Select the appropriate visibility option.

   Note: The visibility setting selected here will not override the default visibility setting of your Facebook application if the application has a more secure level of visibility. For example, if the visibility setting of your Facebook application is set to "Friends", you cannot change this to "Public" by selecting "Public" in this option.

b. Enter a default name, description, image, caption, and message template. Predefined tags for the image, caption and message are included as a guide. For example:

   Name: This is posted as text.
   ```
   [Property field="title" context="current" type="content" format="length:100"]
   ```

   Description:
   This is posted as text.
   ```
   [Property field="description" context="current" type="content"
   format="length:100"]
   ```

   Image:
   This is posted as a URL.
   ```
   [Element context="current" type="content" key="Image" format="URL"]
   ```

   Caption:
   This is posted as text.
   ```
   [Element context="current" type="content" key="Caption"]
   ```

   Message:
   This is posted as text.
   ```
   [Element context="current" type="content" key="Message" format="length:100"]
   ```

Note: Page notes have been deprecated by Facebook.

Defining social network settings for LinkedIn:

Complete these configuration parameters to set up a social network connection for LinkedIn.

Before you begin

Before you begin you must register an application to obtain a user name and password that grants access to the LinkedIn API. To register a new application that the WCM Social Media Publisher can use:

1. Go to https://developer.linkedin.com
2. Sign in and click the twistie next to your login name and select **API Keys**.
3. Click **Add New Application**.
4. Under **Company Info** either select an existing company or select **New Company** and enter a company name.
5. Under **Application Info** enter values for the following parameters:

   Application Name:
   This is the name of your application that is displayed at the end of each post. For example, your company name.

   Application Description:
   Enter a short description of the application.

   Website URL:
   This is the URL to your website. For example: http://www.ibm.com

   Application Use:
   Social Aggregation

   Live Status:
   During testing, you can set the application to **Development**. Once ready, modify the status to **Live**.

   Contact Info:
   Add your contact information here.
6. Under **Author User Agreement** select the display language of the user agreement screen. **Browser Locale Setting** is recommended.
7. Click **Add Application**.
8. A success screen is displayed showing your API Key and Secret Key. Record these values and then click **Done**.

You then must create a new credential vault from the WebSphere Portal administration view where:

- The API Key is specified as the shared user ID
- The Secret Key is specified as the shared password.

About this task

When you create a social network configuration document for LinkedIn:

Procedure

1. Select **LinkedIn** as the social network.
2. Define authentication settings for the social network:
   a. Select the **Credential Vault** that contains your LinkedIn application credentials.
   b. Click **Authorize** to bind the credentials to a specific social network account.
3. Select a message type:

   Note: It is not recommended to reference rich text elements or HTML elements in any fields that do not support HTML.

   Network Update:
   a. Enter a default message template. Predefined tags for the message are included as a guide. For example:

Message:
This is posted as HTML.

```
[Element context="current" type="content" key="Message" format="length:200"] See more:
[URLCmpnt context="current" type="content" mode="standalone" start="<a href='" end="'>Link</a>"]
```

Share:

a. Select the appropriate visibility option.

b. Enter a default name, description, image, and message template. Predefined tags for the image, description, and message are included as a guide. For example:

Name: This is posted as text.

```
[Property field="title" context="current" type="content" format="length:100"]
```

Description:
This is posted as text.

```
[Property field="description" context="current" type="content" format="length:100"]
```

Image:
This is posted as a URL.

```
[Element context="current" type="content" key="Image" format="URL"]
```

Note: The URL to the image must be internet accessible.

Message:
This is posted as text.

```
[Element context="current" type="content" key="Message" format="length:200"]
```

Defining social network settings for Twitter:

Complete these configuration parameters to set up a social network connection for Twitter.

Before you begin

Before you begin, you must register a Twitter application to obtain a user name and password that grants access to the Twitter API. To register a new application that the WCM Social Media Publisher can use:

1. Go to https://dev.twitter.com
2. Log in and click **Create an app**.
3. Click **Add New Application**.
4. Under **Application Info** enter values for the following parameters:

Application Name:
This is the name of your application that is displayed at the end of each post. For example, your company name.

Application Description:
Enter a short description of the application.

Website:
This is the URL to your website. For example: http://www.ibm.com

Callback URL:
Set this to `http://domain/wps/wcmsocial/servlet/oAuthCB/twitter` where "domain" is your domain name.

5. Read the terms and conditions and select **I Agree**.
6. Enter the security/captcha information if required.

7. Your Consumer Key and Consumer Secret is displayed. Make a record of these.
8. Click the **Settings** tab.
9. In the **Application Type** section, set the **Access** property to `Read and write`.
10. Click the **Update this Twitter application's settings**.

When you have registered your application, you then create a new Credential Vault from the WebSphere Portal administration view where:

- The Consumer Key is specified as the shared user ID.
- The Consumer Secret is specified as the shared password.

About this task

When you create a social network configuration document for Twitter:

Procedure

1. Select **Twitter** as the social network.
2. Define authentication settings for the social network:
   a. Select the **Credential Vault** that contains your Twitter application credentials.
   b. Click **Authorize** to bind the credentials to a specific social network account.
3. Enter a default message template. This is posted as text. Predefined tags for the message are included as a guide. For example:

```
[Property field="title" context="current" type="content" format="length:100"]
[URLCmpnt context="current" type="content" mode="storable"]
[profilecmpnt type="content" context="current" field="keywords" separator=" #" include="exact" start="#"]
```

   Note: Rich Text is not supported for Twitter posts, so it is not recommended to reference rich text elements in your post message.

   Note: If your posts are exceeding the 140 character limit for Twitter posts, you can adjust length of the title by reducing the length option in the format parameter as shown in this example. Long category names, or a large number of categories, can also lead to exceeding the 140 character limit.

Global configuration settings:

Although the Social Media Publisher uses social network configuration documents to specify most of its configuration properties, it also uses a global settings document for configuring server-wide options.

To use a global setting, you create a text component that is called **Global Settings** within the **Social Configuration** library. Type the settings that you want to configure into the text field, one per line.

Table 387. Global social media settings

Setting	Details
`actions.post.credentialvault=`	This is used to specify the name of the credential vault that contains the user name and password that is used in post workflow actions. The credential vault user requires edit access to all content that is posted.

Table 387. Global social media settings (continued)

Setting	Details
`library.actions.post.credentialvault=`	This is used to specify the name of the credential vault that contains the user name and password that is used in post workflow actions for a specific library, where "library" is the name of the library you want to assign a credential vault to. The credential vault user requires edit access to all content that is posted.
`actions.disableAllSocialWorkflowActionsOnServers=`	You disable all social media actions on specific servers by entering a comma-separated list of servers. You can specify either node-names, host names or cluster-names. It is recommended that this setting is set to the host name or cluster name of all test, preview, and staging servers.
`actions.disablePostSocialWorkflowActionsOnServers=`	You disable all social media post actions on specific servers by entering a comma-separated list of servers. You can specify either node-names, host names or cluster-names. It is recommended that this setting is set to the host name or cluster name of the authoring server or servers.
`actions.disableDeleteSocialWorkflowActionsOnServers=`	You disable all social media delete actions on specific servers by entering a comma-separated list of servers. You can specify either node-names, host names or cluster-names.
`actions.disableUntrackSocialWorkflowActionsOnServers=`	You disable all social media untrack actions on specific servers by entering a comma-separated list of servers. You can specify either node-names, host names or cluster-names.
`disableDeleteHandlerOnServers=`	You disable all social media delete handlers on specific servers by entering a comma-separated list of servers. You can specify either node-names, host names or cluster-names. The delete handler responds to content deletions and untracks or deletes associated social messages that have already been posted to social media. It is recommended that this setting is set to the host name or cluster name of all test, preview, and staging servers.

Table 387. Global social media settings (continued)

Setting	Details
For Facebook: • facebook.endUser.statistics.enabled=true • facebook.endUser.statistics.domain=YOUR_DOMAIN For Twitter: • twitter.endUser.statistics.enabled=true • twitter.endUser.statistics.domain=YOUR_DOMAIN	These settings are used to enable the tracking of end user social activity. Set these parameters to false to disable statistic tracking for a specific social network. These settings can be individually specified per-library by adding the name of the library to the beginning of the parameter. For example: myLibrary.twitter.endUser.statistics.enabled=true When using virtual portals, or not performing servlet-rendering on Web Content Manager V7.0, then the domain' properties must include the virtual portal context. For example: twitter.endUser.statistics.domain=localhost/wps/por
socialTable.showFullRefreshButton=false	If set to true, the social information table displays a **Refresh All** button that will refresh both statistics and table data. If set to false, a **Refresh Table** button is displayed, and only table data is refreshed.
format:network_name.message_type.postfield.default=value to use the default_value	Use this to define the default value for a Post format field. For example: connections.blog_post.message.defaultValue= [Element context="current" type="content" key="Mess format="length:100"] See more: [URLCmpnt context="current" type="content" mode="st start="Link"

Table 387. Global social media settings (continued)

Setting	Details
jms.processRemoteMsgs=	The Social Media Publisher uses the built-in JMS functions of Web Content Manager to monitor and respond to events that are occurring within a system. When the Social Media Publisher is running within a cluster, each server needs to be configured to match the WebSphere Application Server JMS policy type in effect. To determine the current WebSphere Application Server JMS policy type, open the WebSphere Application Server Administration Console and navigate to **Resources > JMS > Topics > IWKTopics_Items > Buses > IWKBus > Bus Members > PortalCluster** **Note:** This is only available if the Social Media Publisher is installed or if Web Content Manager JMS support has been manually enabled. The jms.processRemoteMsgs setting is set to either true or false depending on the WebSphere Application Server JMS policy type: **High Availability (Default)** Set jms.processRemoteMsgs to true. **Scalability** Set jms.processRemoteMsgs to false. **Scalability with high availability** Set jms.processRemoteMsgs to false. **Custom** Set jms.processRemoteMsgs to true if there is a single JMS engine per cluster, or false if there is a JMS engine per node. When this setting is syndicated to all environments, all clusters in your environment will share the same WebSphere Application Server JMS policy type.

Note: These settings are also documented in the **Global Settings Reference** component that is stored in the **Social Configuration** library.

Adding the social information table to an authoring template:

If the **Automatically add social information element** setting on a social network configuration document is disabled, you must manually add the social information element to any authoring template you want to use the social media publisher on.

Procedure

1. Open the authoring template that you want to add the social information element to in edit mode.
2. Add a text element named **SocialNetworkInformation**.
3. Set the **Custom JSP** property of the element to:
   ```
   readMode=/wps/wcmsocial;/jsp/html/ItemField.jsp,editMode=/wps/wcmsocial;/jsp/html/ItemField.js
   ```
4. Save and close the authoring template.

What to do next

To apply the element to all existing content that uses a specific authoring template, open the authoring template in read mode and click **Apply Authoring Template**, ensuring that the **Add new elements** option is selected.

The social network information table

When you create a content item by using a content authoring template that is mapped to a social network configuration document, an element that is named **Social Network Information** is displayed on the content item.

- To post information about the content item to a specific social network, click **Post** and select the social network of choice.
- To post information about the content item to all social networks, click **Post to All**.
- When posted, information about the social media is listed next to each active social media post. You can refresh this list by using the **Refresh Statistics** action.
- You can stop receiving information about the social media post by clicking the **Untrack** action.

  Note: When you stop tracking an item, you cannot go back and track the item from the time it was published. All your old tracking data is lost. You might not be able to post to social media again as the new post can be considered as duplicate content.
- If you have view access to social configuration documents, you use the **View configuration** action to display the configuration document for a social network.
- Click **Delete** to delete a post from the social network altogether.

Social Media Workflow Actions

A set of social media workflow actions are supplied to with the Social Media Publisher and are found in the **Social Configuration** library. These actions can be added to workflows to run social media actions as part of a workflow.

Preinstalled social media workflow actions

These social media workflow actions are included with the Social Media Publisher:

- Post to all registered configurations
- Untrack message from all registered configurations
- Delete message from all registered configurations

Custom social media workflow actions

You can also create the following social media workflow actions for individual configuration documents:

- Post to specific configuration
- Untrack message from specific configuration
- Delete message from specific configuration

To use these actions, you must:

1. Create a custom workflow action:
2. Enter the name of the configuration document you want to use this action with in **Description** field. This description must exactly match the name of the configuration document, not the display title.
3. Select an action from the **Social Custom Workflow Action Factory**.

Post Action Credentials

Post actions require a credential vault slot called **socialPostUser** populated with credentials of a user that has the following access. This is normally an administration user:

- Edit access to all content that will be posted.
- Contributor access to all required social configuration documents.
- User access to the social configuration library. This must be set on both syndicator and subscriber servers.

The name of the credential vault used can be customized via the global setting **actions.post.credentialvault**. You can also specify a library specific vault via the global setting **library.actions.post.credentialvault**, where library is the name of the library the action is saved in.

Using Social Media Workflow Actions In Projects

When using social media workflow actions with projects, you can:

- Add social media workflow actions directly to a project to run them on all content items in the project. This requires IBM Web Content Manager version 8.
- Add social media workflow actions to the publish stage of individual workflows used by content items. The social media workflow actions are automatically executed once the project is published.
- Use both project level and item level social media workflow actions. Project level actions will run before item level actions.

Workflow Scheduling and Syndication

Social media actions are enabled by default on all the servers that subscribe to your authoring server and social media posts will be sent from all the servers in your system when a social media action is run.

It is best practice to disable social media actions on all servers except your rendering server. To do this, update the following parameter in your global settings:

- `actions.disableAllSocialWorkflowActionsOnServers`
- `actions.disablePostSocialWorkflowActionsOnServers`
- `actions.disableDeleteSocialWorkflowActionsOnServers`

- `actions.disableUntrackSocialWorkflowActionsOnServers`

See "Global configuration settings" on page 2433 for further details.

Publishing content directly to a Facebook page

The Social Media Publisher is used to post status updates about your content to Facebook profiles and pages. You can also post content items directly to a Facebook page.

Procedure

1. Go to `https://developers.facebook.com/apps`
2. Click **Create new App**.
3. A form opens. Enter values for the following parameters:

   App Name:
   This is the name of your application that is displayed at the end of each post. For example, your company name.

   App Namespace:
   This is name that is used to access Facebook Insights information. It is recommended that a unique name is used for each application.
4. Click **Continue**.
5. A configuration screen for your application is displayed. Select **Page Tab** under **Select how your app integrates with Facebook** and complete the following parameters:

   Page tab name:
   Enter the name for the page tab.

   Page tab URL:
   This is the URL to the content item you want to add to the Facebook page.

   Secured Page tab URL:
   This is an https version of the Page tab URL.
6. Click **Save Changes**.
7. Click **Authorize** to bind the credentials to a specific social network account.

What to do next

- To post your content to the Facebook page, use the following URL where `APP_ID` is the Application-Id of Facebook Page Tab application:
  `http://www.facebook.com/dialog/pagetab?app_id=APP_ID&next=https%3A%2F%2Fwww.facebook.com%2Fconnect%2Flogin_success.ht`
- Publishing content directly to a Facebook page requires a Secured Page tab URL to be provided. This means that your server must be SSL enabled.

Social media for users

While the current version for the Social Media Publisher focuses on functions for you and your company as website owners, you might also consider adding social features to your rendered website for use by your users.

To enable this feature:

1. Review the documentation for each social network for the markup they require to be added to a page to integrate their social network badges:

   Facebook:
   http://developers.facebook.com/docs/guides/web/

Twitter:
https://dev.twitter.com/docs/twitter-for-websites

2. When you specify the content url to use as an input to the Social badges, use the following tag: `http://YOUR_DOMAIN[URLCmpnt context="current" type="content" mode="static"]`, where `YOUR_DOMAIN` is the production domain of your website.

   For example: `http://www.ibm.com[URLCmpnt context="current" type="content" mode="static"]`

   The domain that is specified in the URL must be the same as the domain specified in the *facebook.endUser.statistics.domain* and *twitter.endUser.statistics.domain* global settings.
3. Enable the tracking of user's social activity within the Social Media Publisher by setting the global settings `Facebook.endUser.statistics.enabled` and `Twitter.endUser.statistics.enabled` to `true`. See "Global configuration settings" on page 2433 for details.

Social media certificates

To communicate to Facebook, LinkedIn and Twitter, your server requires the SSL certificates for these social networks to be installed.

When these certificates expire, you must refresh them on your server by removing then and adding them again.

Viewing certificates

To view a list of current certificates for Facebook, LinkedIn and Twitter, run the following command from the *wp_profile_root*/ConfigEngine directory:

Windows
```
ConfigEngine.bat action-list-wcm-social-certs
```

Linux
```
./ConfigEngine.sh action-list-wcm-social-certs
```

IBM i
```
ConfigEngine.sh action-list-wcm-social-certs
```

Removing certificates

To remove all current certificates for Facebook, LinkedIn and Twitter, run the following command from the *wp_profile_root*/ConfigEngine directory:

Windows
```
ConfigEngine.bat action-remove-wcm-social-certs
```

Linux
```
./ConfigEngine.sh action-remove-wcm-social-certs
```

IBM i
```
ConfigEngine.sh action-remove-wcm-social-certs
```

Installing certificates

To install the certificates for Facebook, LinkedIn and Twitter, run the following command from the *wp_profile_root*/ConfigEngine directory:

Windows
```
ConfigEngine.bat action-install-wcm-social-certs
```

Linux
```
./ConfigEngine.sh action-install-wcm-social-certs
```

IBM i
```
ConfigEngine.sh action-install-wcm-social-certs
```

Note: If your server is located behind a firewall, you might need to manually import these certificates. See the WebSphere Application Server documentation for instructions on importing certificates. The following locations are for each certificate type:

- Facebook: `graph.facebook.com`
- LinkedIn: `api.linkedin.com`
- Twitter: `api.twitter.com`

SSL port 443 is the default SSL port for these certificates.

Uninstalling the Social Media Publisher

The following procedures to uninstall the Social Media Publisher must be run on every server and cluster node.

About this task

Before starting:

- Ensure that the **WasPassword** and **PortalAdminPwd** passwords are set in the `wkplc.properties` file.
- Ensure that the **jcr.DBPassword** password is set in `wkplc_dbdomain.properties` file.

Procedure

1. To remove the Social Media Publisher tracking data tables, run the following command from the *wp_profile_root*/ConfigEngine directory:

   Windows
   ```
   ConfigEngine.bat action-remove-wcm-social-tables
   ```

   Linux
   ```
   ./ConfigEngine.sh action-remove-wcm-social-tables
   ```

   IBM i
   ```
   ConfigEngine.sh action-remove-wcm-social-tables
   ```

2. To completely remove the Social Media Publisher, run the following command on all servers or nodes in your system from the *wp_profile_root*/ConfigEngine directory:

   Windows
   ```
   ConfigEngine.bat remove-wcm-social
   ```

   Linux
   ```
   ./ConfigEngine.sh remove-wcm-social
   ```

   IBM i
   ```
   ConfigEngine.sh remove-wcm-social
   ```

3. Restart all servers to complete the uninstallation.

What to do next

Note: All SSL certificates for Twitter, Facebook and Linkedin are removed when the uninstall script is run.

Known issues with the Social Media Publisher

These are the current known issues with the Social Media Publisher and their solutions.

Hung tread message in SystemOut.log when running SMP traffic with 10 Virtual Users

WebSphere Application Server provides a detection feature that attempts to locate and report on potentially hung threads in the system. Hung threads can be

difficult to diagnose. The WebSphere Application Server thread monitor architecture monitors managed threads and is enabled by default. When a thread is suspected to be hung, a notification message is written to Sysprint. False alarms do occur. When a false alarm occurs, a followup notification is sent to Sysprint.

Symptom

The following warning from WebSphere Application Server may be observed in the System log:

```
[7/19/12 13:26:39:031 EDT] 00000033 ThreadMonitor W   WSVR0605W: Thread "WorkManager.wpsDefaultWorkManager : 72" (000000ee
    has been active for 682689 milliseconds and may be hung.  There is/are 1 thread(s) in total in the server that may be
    at sun.misc.Unsafe.park(Native Method)
    at java.util.concurrent.locks.LockSupport.parkNanos(LockSupport.java:224)
    at java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject.awaitNanos(AbstractQueuedSynchronizer.java:20
    at java.util.concurrent.LinkedBlockingQueue.poll(LinkedBlockingQueue.java:435)
    at com.ibm.workplace.wcm.messaging.data.JMSTopicContext.getNextMessage(JMSTopicContext.java:232)
```

Cause The Social Media Publisher makes use of JMS background threads that are long running by nature. There is no facility for the Social Media Publisher application to pre-warn the WebSphere Portal server, or the WebSphere Application Server. As a result of this limitation, false alarms can be observed relating to JMS threads, which are used by Social Media Publisher.

Solution

Until this design limitation is corrected, warnings about potential hung threads can be observed, but can be ignored. WebSphere Application Server reports on falsely alerted threads that resume normally. It is possible, but not necessary, to adjust these intervals and thresholds. Note the following custom properties and refer to the WebSphere Application Server documentation for more information:

com.ibm.websphere.threadmonitor.interval
Defaults to 180 seconds. This is the interval at which the thread pools are polled for hung threads.

com.ibm.websphere.threadmonitor.threshold
Defaults to 600 seconds. This is the length of time that a thread can be active before being marked as "potentially hung".

com.ibm.websphere.threadmonitor.false.alarm.threashold
Defaults to 100 false alarms. This is the number of false alarms that can occur before automatically increasing the threshold by 50%.

IBM Web Content Manager Multilingual Solution

The Web Content Manager Multilingual Solution is a set of tools that are used to manage translated versions of localized and regionalized websites.

The multilingual solution requires:

- WebSphere Portal version 8.0.0.1 or later, or Web Content Manager version 8.0.0.1 or later.

"Overview of a multilingual site" on page 2443
A multilingual site consists of content that is localized to support multiple audiences with different languages. Your multilingual site consists of a set of localized and regionalized subsites, including a single base site.

"Deployment, installation, and configuration" on page 2443
Before you use the multilingual solution, you must deploy, install and configure the multilingual solution extensions, and the also update and configure your system.

"How to use the multilingual solution" on page 2452
The multilingual solution is used to author, manage, and publish multilingual content.

"Extensions for multilingual sites" on page 2464
The multilingual solution uses a set of extensions that can be used with existing features to author, manage, and publish your multilingual websites.

Overview of a multilingual site

A multilingual site consists of content that is localized to support multiple audiences with different languages. Your multilingual site consists of a set of localized and regionalized subsites, including a single base site.

Types of sites

The base site

- The base site defines the site structure and contains the shared content that is translated.
- Some base sites are continually updated, and fresh translations are made.
- Other base sites are created once, and used as templates for new sites in other languages.

Translated sites

- In most cases, the entire content and structure of a base site are replicated to a translated site.
- In some cases, only a subset of the original base site is reused on the translated site.

Regional sites

- A regional site is similar to a translated site, but involves no translation.
- Regional sites reuse content from a base site, but the content is updated for each region. For example, different states in a single country.

Translated Regional Sites

- These types of sites both reuse and update content from a base site, but also translates the updated content.

No base-locale sites

- In this type of site, content is written separately in different sites for different languages, but then synchronized and translated to all the sites and languages.

Mixed language sites

- This type of site includes content that is written in multiple languages, but stored in a single site.
- Content is grouped by category, not language.

Deployment, installation, and configuration

Before you use the multilingual solution, you must deploy, install and configure the multilingual solution extensions, and the also update and configure your system.

"Multilingual deployment" on page 2444
Deployment of multilingual sites can be centralized, running out of single environment, or decentralized where locales are served from different environments.

Multilingual deployment

Deployment of multilingual sites can be centralized, running out of single environment, or decentralized where locales are served from different environments.

Centralized deployment

In a centralized deployment, all locales are served from the same environment and all of your localized content is syndicated. Web server mappings to each locale home page are created either as domains or as paths. These mappings provide easily remembered URLs for users to link to a localized version of your site.

Using a single domain name

When a single domain name is used, each path is mapped to the home page in a localized library, by using URLs like `http://mydomain/en` or `http://mydomain/es` for different localities.

Using multiple domain names

When multiple domain names are used, each domain is mapped to the home page in a localized library, by using URLs like `http://mydomain.com` or `http://mydomain.com.es` for different localities.

Your localized navigation links change the domain, not just the path, and use a fully qualified path, not a relative path.

Decentralized deployment

In a decentralized deployment, locales are served from separate environments, one per environment, or potentially multiple locales per environment. Multiple domains must be used, where each locale or combination of locales is served from its own domain.

- When you implement a decentralized deployment, users either switch between localized content on an item by item basis, or users select the locale to use and are taken to the home page for that locale.

- To support a decentralized environment, you need to have all content in all locales that are deployed to every locale environment. The navigation code can then check which content is available before links are rendered, and the user can browse to that localized version without losing their session.
- When using an anonymous site, you can redirect to the localized version of a site on another server. This strategy requires fully qualified links to be generated.

Locale home page navigation

If your site requires a link to go to the home page of each locale instead of item-by-item navigation, you can create links to your other servers. The advantage of implementing your site this way is that you do not need to deploy all content to all servers. You only need the content for the locale you intend to serve out of that environment and any shared assets.

The links to your other localized domains need to be hard-coded because these links cannot be automatically generated.

Installing the multilingual extensions

The multilingual solution consists of a set of extensions to Web Content Manager that can be used to manage the authoring, workflow, and configuration of your multilingual system.

Procedure

1. Ensure that the **WasPassword** and **PortalAdminPwd** passwords are set in the `wkplc.properties` file.
2. Run the following registration command from the *wp_profile_root*/`ConfigEngine` directory:

   Windows
   ```
   ConfigEngine.bat register-wcm-mls
   ```

   Linux
   ```
   ./ConfigEngine.sh register-wcm-mls
   ```

   IBM i
   ```
   ConfigEngine.sh register-wcm-mls
   ```
3. Run the following deployment command from the *wp_profile_root*/`ConfigEngine` directory:

   Windows
   ```
   ConfigEngine.bat deploy-wcm-mls
   ```

   Linux
   ```
   ./ConfigEngine.sh deploy-wcm-mls
   ```

   IBM i
   ```
   ConfigEngine.sh deploy-wcm-mls
   ```
4. If your server contains virtual portals, you must also run the following task for each virtual portal on your server:

   Windows
   ```
   ConfigEngine.bat import-wcm-mls-data
   -DVirtualPortalHostName=VirtualPortalHostName
   -DVirtualPortalContext=virtual_portal_context_url
   ```

   Linux
   ```
   ./ConfigEngine.sh import-wcm-mls-data
   -DVirtualPortalHostName=VirtualPortalHostName
   -DVirtualPortalContext=virtual_portal_context_url
   ```

   IBM i
   ```
   ConfigEngine.sh import-wcm-mls-data
   ```

```
-DVirtualPortalHostName=VirtualPortalHostName
-DVirtualPortalContext=virtual_portal_context_url
```

5. Restart WebSphere Portal.
6. Repeat these steps on every server and cluster node.

Configuring a multilingual system

After you install the multilingual extensions, you then must configure your system to support multilingual authoring and rendering.

Procedure

1. Log into WebSphere Portal as an administrator.
2. Create a new group for your locale owners:
   a. Click the **Administration menu** icon. Then, click **Access** > **Users and Groups**.
   b. Create a new group for your locale owners. For example: **Locale Owners**. The base locale owner must belong to this group.
3. Modify the security of the **MLConfiguration_v7** library:
   a. Click the **Administration menu** icon. Then, click **Portal Content** > **Web Content Libraries**.
   b. Assign the **Locale Owners** group to the **Editor** role on the library.
   c. Assign the **Locale Owners** group to the **Contributor** role on the library.
   d. This is an optional step. To hide all sections in the authoring portlet, except the content and component views, disable inheritance for all item types except content and components.
   e. Assign the **Administrators** group to the **Administrators** role on the library and all item types.
4. Run the Update Member Fixer tool by running the following command from the *wp_profile_root*/ConfigEngine directory:

   Windows
   ```
   ConfigEngine.bat run-wcm-admin-task-member-fixer
   -DPortalAdminId=username -DPortalAdminPwd=password
   -DWasUserId=username -DWasPassword=password
   -Dlibrary=MLConfiguration_v7 -Dfix=true -DinvalidDn=update
   -DmismatchedId=update -DaltDn=update
   ```

   Linux
   ```
   ./ConfigEngine.sh run-wcm-admin-task-member-fixer
   -DPortalAdminId=username -DPortalAdminPwd=password
   -DWasUserId=username -DWasPassword=password
   -Dlibrary=MLConfiguration_v7 -Dfix=true -DinvalidDn=update
   -DmismatchedId=update -DaltDn=update
   ```

5. Go to the authoring portlet and update the Configuration settings to add the **MLConfiguration_v7** library to the list of selected libraries.
6. Create multilingual configuration file for each set of localized or regionalized libraries.
   a. From the authoring portlet, create a new content item:
      - use the **LocalizedConfigurationFileAT** content template from the **MLConfiguration_v7** library for localized sites.
      - use the **RegionalizedConfigurationFileAT** content template from the **MLConfiguration_v7** library for regionalized sites.
   b. Type a name and display title to represent the set of multilingual sites.
   c. Complete the following fields:

Base Content Library
Type the name of the library that is used to store items from the default locale, or leave the field blank to indicate that no base locale is used.

Content Libraries
Type the names of all the libraries in your multilingual system, including the base content library, separated by commas. This list must not include any shared design libraries. The order that the libraries are entered in this field determines the output of the render-time and edit-time extensions, so it is recommended to place the base content library first then add the remaining libraries in the order that you want them to be displayed.

Has Regionalizations
Used for localized sites only, this setting determines whether any of the localized sites have associated regionalized versions. Change this setting to true if any regionalized sites are associated with the current set of libraries.

Content Library Owners
For each library listed in the **Content Libraries** field, specify the library name and email address of the user who will be the owner for that library, placing each entry on a new line. For example:
`MyLibraryFR=wpsadmin@portal.com`

Table 388. Configuration file examples

Site Type	Language Tree	Libraries	Configuration files
Localized site	• English • Spanish	• English • Spanish	**Localized Configuration file** • Base Content Library: English • Content Libraries: English, Spanish

Table 388. Configuration file examples (continued)

Site Type	Language Tree	Libraries	Configuration files
Regionalized site	• English – English US (Primary) – English Australia • Spanish – Spanish Spain (Primary) – Spanish Mexico	• English • English US • English Australia • Spanish • Spanish Spain • Spanish Mexico	**Localized Configuration file** • Base Content Library: English • Content Libraries: English, Spanish **Regionalized Configuration file 1 (English)** • Base Content Library: English • Content Libraries: English, English US, English Australia **Regionalized Configuration file 2 (Spanish)** • Base Content Library: Spanish • Content Libraries: Spanish, Spanish Spain, Spanish Mexico

7. For each library referenced in each multilingual configuration file, create a link component directly under the Components folder that references the multilingual configuration file:
   - For references to a localized multilingual configuration file, the link component name must be **MLConfFileReference**.
   - For references to a regionalized multilingual configuration file, the link component name must be **RegionalizedMLConfFileReference**.
   - The **ALL_USERS** group must be assigned to the **User** role in the component access controls.
8. This step is optional. Activate email notifications for the workflow synchronization extensions:

a. Open a multilingual configuration file from the **MLConfiguration_v7** library.
b. Update the following settings in the **General Workflow Synchronization Settings** section:

emailServer
Type the name of your email server.

emailFromAddress
The email address that is entered here is used to set the "From" address on all email notifications. This field must be set to a valid email address.

authoringUIURL
Type the URL of your Authoring server. For example, `http://localhost:10045/wps/myportal/wcmAuthoring`

c. Enable email notifications for each multilingual extension:

Localized Sites
Set the **localize.emailNotificationsEnabled** field in the **Localize Workflow Synchronization Settings** section to **true**.

Regionalized Sites
Set the **regionalize.emailNotificationsEnabled** field in the **Regionalize Workflow Synchronization Settings** section to **true**.

Synchronized publishing
Set the **SyncPublish.emailNotificationsEnabled** field in the **SyncPublish Workflow Synchronization Settings** section to **true**.

Synchronized expiration
Set the **SyncExpire.emailNotificationsEnabled** field in the **SyncExpire Workflow Synchronization Settings** section to **true**.

Synchronized deletion
Set the **SyncDelete.emailNotificationsEnabled** field in the **SyncDelete Workflow Synchronization Settings** section to **true**.

9. This step is optional. To enable project integration for synchronized publishing, set the **SyncPublish.useProjects** field in the **SyncPublish Workflow Synchronization Settings** section to **true**.
10. Enable syndication for the **MLConfiguration_v7** library.

Note: The multilingual solution must be installed on both the syndicator and subscriber before syndication can be enabled.

Optimizing the performance of the extensions

How to optimize your system for best performance of the multilingual solution.

Procedure

1. Add the following settings to the **WCM WCMConfigService** service by using the WebSphere Application Server.
   a. Enable the missed path cache by adding `missed_absPath.cache.enable=true`.
   b. Enable the user cache by setting `user.cache.enable=true`.
   c. Restart the server.
2. Update these settings in the WebSphere Application Server to enable "disk-offload" and "flush to disk" for the following caches:

ML Config File
`services/cache/iwk/mlconffile`

Abs Path
`services/cache/iwk/abspath`

Abs Path Reverse
`services/cache/iwk/abspathreverse`

Missed Path
`services/cache/iwk/missed`

Object Summary
`services/cache/iwk/summary`

Strategy
`services/cache/iwk/strategy`

Note: These changes require manual flushing of the cache offload directories whenever an iFix, Cumulative iFix or Fixpack is installed.

3. Periodically run the clear history tool to improve page load and save performance.
4. Disable workflow actions on subscribers to reduce load and reduce the number of versions that are created for each item. See "Disabling Workflow Actions" on page 434 for further information.

Capacity planning for multilingual sites

It is important to consider the additional load that is placed on the authoring server in comparison with single language websites.

The workload that is imposed by each additional language can be considered equivalent to the workload imposed by an extra set of authoring users. So if you have 50 users that support a single language website, then adding four extra languages is equivalent to having 250 users creating and publishing individual content items.

- When synchronized publishing is required, enabling project integration is recommended as this strategy greatly reduces the load that is generated by this function.
- To further reduce the load that the multilingual extensions generate, the following settings can be modified in each configuration file, with higher values reducing the load:

Localize / Regionalize Workflow Synchronization Settings

`localize.processingDelay`
Specifies the delay in milliseconds between processing each item. Defaults to 100 milliseconds.

`regionalize.processingDelay`
Specifies the delay in milliseconds between processing each item. Defaults to 100 milliseconds.

SyncPublish Workflow Synchronization Settings
This is required only for the legacy synchronized publishing implementation where project publishing synchronization is not used.

`syncPublish.processingDelay`
Specifies the delay in milliseconds between processing each item. Defaults to 2000 milliseconds.

syncPublish.rescheduleInterval
Specifies the interval in seconds to reschedule the action if documents are outstanding. Defaults to 300 seconds.

SyncExpire Workflow Synchronization Settings

syncExpire.processingDelay
Specifies the delay in milliseconds between processing each item. Defaults to 2000 milliseconds.

syncExpire.rescheduleInterval
Specifies the interval in seconds to reschedule the action if documents are outstanding. Defaults to 300 seconds.

SyncDelete Workflow Synchronization Settings

syncDelete.processingDelay
Specifies the delay in milliseconds between processing each item. Defaults to 5000 milliseconds.

Changing the multilingual solution context root

The default context root for the multilingual solution application is /wps. With Version 8.5 CF3, you can change this setting to match the WebSphere Portal context root to better suit the requirements of your organization.

Procedure

1. Ensure that the WebSphere®Portal context root is modified as described in "Changing the portal URI after an installation" on page 368.
2. Ensure that the *WasPassword* and *PortalAdminPwd* passwords are set in the **wkplc.properties** file.
3. Run the following registration command from the *wp_profile_root*/ConfigEngine directory:

   Windows
   ```
   ConfigEngine.bat mls-modify-servlet-path
   ```

   Linux
   ```
   ./ConfigEngine.sh mls-modify-servlet-path
   ```

   IBM i
   ```
   ConfigEngine.sh mls-modify-servlet-path
   ```
4. Restart WebSphere Portal.
5. Repeat these steps on every server and cluster node.
6. If the **Edit-Time Navigation/Creation** extension is being used, for each authoring template that uses the extension:
   a. Edit the **ML Translations** text element properties and update the custom JSP field with dynamic context path tokens:

      For the Auto load version:
      ```
      readMode=[wcm.mls.context.path];/jsp/html/MLAuthorTimeRead_AutoLoad.jsp,
      editMode=[wcm.mls.context.path];/jsp/html/MLAuthorTimeEdit_AutoLoad.jsp
      ```

      For the Manual load version:
      ```
      readMode=[wcm.mls.context.path];/jsp/html/MLAuthorTimeRead_ManualLoad.jsp,
      editMode=[wcm.mls.context.path];/jsp/html/MLAuthorTimeEdit_ManualLoad.jsp
      ```

   a. Save the authoring template.
7. If the **Domain Locale Redirection** extension is being used, update the HTTP Server with the new context root. For example, if you are using the IBM HTTP Server with the following configuration settings and the new context root is /abc:

```
RewriteRule ^/$ /wps/wcmml/jsp/html/MLServletHomePageRedirection.jsp [PT]
RewriteRule ^/en$ /wps/wcm/connect/english/Internet.jsp [PT]
```

Should be updated to:

```
RewriteRule ^/$ /abc/wcmml/jsp/html/MLServletHomePageRedirection.jsp [PT]
RewriteRule ^/en$ /abc/wcm/connect/english/Internet.jsp [PT]
```

Uninstalling the multilingual solution extensions

The following steps are required to uninstall the multilingual solution extensions. These must be run on every server on non-clustered systems, or on the primary cluster node.

Procedure

1. Remove all multilingual workflow stages from each of your workflows.
2. If the **Edit-Time Navigation/Creation** extension is being used:
   a. Manually remove the field from each authoring template that uses the extension.
   b. Save and then reapply any updated authoring templates by using the Remove existing elements option.
3. If the **Servlet Render-Time Navigation** extension is used, remove it from your presentation templates.
4. Ensure that the **WasPassword** and **PortalAdminPwd** passwords are set in the `wkplc.properties` file.
5. To completely remove the multilingual solution, run the following command on all servers or nodes in your system from the *wp_profile_root*`/ConfigEngine` directory:

   Windows
   `ConfigEngine.bat remove-wcm-mls`

   Linux `./ConfigEngine.sh remove-wcm-mls`

   IBM i `ConfigEngine.sh remove-wcm-mls`
6. Delete the **MLConfiguration_v7** library on base portal and all virtual portals.
7. Restart the Portal Server.

How to use the multilingual solution

The multilingual solution is used to author, manage, and publish multilingual content.

"Framework for multilingual site management" on page 2453
The framework covers a data model for storing multilingual sites, processes for managing multilingual sites, and a model for delivering multilingual sites to site visitors.

"Multilingual authoring" on page 2454
Nearly all authoring tasks occur in the base site, and are then replicated, if appropriate, to all of the localized sites. Authoring occurs on translated sites for language or region-specific content, or to fix errors in translation.

"Lifecycle and synchronization" on page 2455
Workflow is used to keep the base site and localized sites synchronized. The creation, modification, publishing, expiry, deletion, and moving of site areas and content can be synchronized across locales by using a workflow.

"Multilingual presentation" on page 2457
Localizing the presentation layer in your site involves deciding which parts of the presentation layer need to be localized, and then separating out those pieces into components that can be localized.

Framework for multilingual site management

The framework covers a data model for storing multilingual sites, processes for managing multilingual sites, and a model for delivering multilingual sites to site visitors.

The key elements of this framework are:

Use one library per locale

A library is created for the base site, and one library for each localized site, and another library for shared templates and components. When you create libraries, ensure that the correct locale is specified during the creation process, as this locale is used during search and in the multilingual user interface extensions.

Note: If a language does not exist in the list of languages available when you create a library, you can add that language to the list of supported languages.

Same site structure in each locale

For content to be recognized as being equivalent between sites, they must have the same URL path. This means the names and structure of all the site areas and content items in your site framework must be identical in each site if they are to be equivalent. The display titles can be different, but the names must match.

Use workflow for all modifications

Creating, updating, publishing, expiring, deleting, and moving content and components are made by using workflow. This strategy allows code to be triggered by the workflow to notify localized sites owners, automate parts of the localization process, and synchronize important stages in the workflow.

Workflows can be localized by creating an equivalently named workflow in each locale library. Localizing the workflow allows separate approval processes to be run per locale and is also a requirement for regionalized sites. Localizing the workflow is part of synchronization process, and the provided extension for workflow synchronization has this ability.

When you create new templated portal pages that are associated with site areas and content items, the templated items are published. The items that are created by copying the site areas and content items are also published. In this case, the items need to be copied to all of the localized sites.

Use shared templates and components

Authoring templates, presentation templates, and components are shared across locales as much as possible. Variability can be built in using

workflows or workflow code that differs across locales, and presentation components that are picked up from the current locale.

Authoring templates can be shared across localizes whenever possible with text providers used to provide localized template display titles, element names, and help-text.

Presentation templates can be different across locales by having different mappings in each locale.

Multilingual authoring

Nearly all authoring tasks occur in the base site, and are then replicated, if appropriate, to all of the localized sites. Authoring occurs on translated sites for language or region-specific content, or to fix errors in translation.

Localization of authoring content

While the initial creation of both the base locale item and any localizations can be done manually by using the authoring portlet, it is recommended to use either a custom authoring field or workflow synchronization code to automate the initial creation of localized documents in the base locale language that are then translated by the appropriate team.

Names and display titles

- The **Name** field in the **Identification** section must not be translated. Doing so breaks the link between the translated item and its base copy.
- Translators are free to translate the **Display Title** field. This is recommended so that links are displayed by using the correct translation of the content name.

Other fields

- Other fields can be different between locales, but for the purposes of consistency of availability and access, typically many of the metadata fields would be kept in sync, including: authors, categories, publish date, expiry date, and security.
- The **owner** field may be localized to indicate the owner of the specific localization. This might be used during automated workflow code to notify the appropriate users of updates.
- The multilingual extensions do not resynchronize fields during updates, nor does it use the owner field. This can be done by using extra custom workflow actions added to the **Localize** and **Regionalize** workflow stages.

Authoring Templates

There are multiple ways in which authoring templates can be set up in a multilingual environment. The common authoring template model is recommended.

Common Authoring Templates

Common authoring templates are usually created in the common library and used for all locales. Text providers can then be written to provide localized template display titles, element names, and help-text. The disadvantage of this approach is that the default values of any elements cannot be translated. You can work around this limitation by moving the recommended default value into the element help text.

Localized Authoring Templates

Each locale uses separate authoring templates with manually translated element names. This approach is not recommended for the following reasons.

- Querying of content is more complicated. Each separate authoring template must be selected in queries that are used by menus or personalization rules.
- Content creation is more complicated. Users must choose the appropriate template not just for the type of content, but also for the appropriate language or locale.
- Maintaining localized authoring templates is more complicated than maintaining common authoring templates. Changes to the base authoring template needs to be duplicated in each of the localized authoring templates, although the multilingual workflow extensions can help with coordinating these changes.

If there is a valid reason to translate the default values of the base authoring template, then it is recommended that this is implemented by creating localized authoring templates in each localized site library that all have the same name, and then set the *localize.supportLocalizedAuthoringTemplates* and *regionalize.supportLocalizedAuthoringTemplates* properties of the **Localize Workflow Synchronization Settings** and **Regionalize Workflow Synchronization Settings** sections of each multilingual configuration file to *true*. Setting the *supportLocalizedAuthoringTemplates* property to *true* enables localized authoring template support and automatically associates each created document with its localized authoring template.

Single Authoring Templates

This approach involves the use of a single authoring template with fields for each locale. This approach is not recommended for the following reasons:

- This approach is not scalable. Only a limited number of languages can be supported due to the number of fields on the authoring template.
- You cannot have content authors editing localizations concurrently.
- Localizations cannot be syndicated separately.
- Only a single workflow can be used.
- Localized content cannot be published or expired independently.
- You cannot use non-Unicode character sets.

Previewing

To preview content to see how it will look in a specific locale or language, including the appropriate encoding, preview the content from within the appropriate library.

Lifecycle and synchronization

Workflow is used to keep the base site and localized sites synchronized. The creation, modification, publishing, expiry, deletion, and moving of site areas and content can be synchronized across locales by using a workflow.

Creating, updating, and moving translated content and components

To keep content and components synchronized between locales, a special workflow stage that runs after the approval stage is created in the base library. This stage is

used for the updating, moving, and creation of content. If the current content item has already been published, the action must be an update or move action. Translated content is treated as new content if no published version exists.

Creating

When new content and components are created in the base library, one of the following actions is run against each configured localized library and a notification sent to localized content owners:

- The item is copied into the library for later translation.
- The item is linked into the library if no translation is required. This applies to content items only.
- Nothing happens and the library is ignored.

Updating

When content or components are updated in the base library, draft copies of existing published localized items are created and a notification sent to localized content owners.

Moving

When you move content in the base library, rather than move the published content, a draft copy of the published content must be created first and then the draft moved to the correct location.

When the draft content is detected as moved within the base library, draft copies of the existing published localized content must be created and moved to the corresponding location in the localized library. A notification is then be sent to localized content owners.

Publishing synchronization

- To ensure that the base item and localized items are published at the same time, project integration can be enabled for the *Update Notification* workflow synchronization extension.
- When project integration is enabled, then all associated draft localized documents are added to a temporary project that is automatically published when all documents are approved.
- When the project is published, a notification is sent to all localized content owners. When the project is successfully published, the project is deleted.

Expiry synchronization

- To ensure that the base item and localized items are expired at the same time, another special workflow stage is required. This workflow stage is run after the publish workflow stage.
- If every published localized copy is in the publish workflow stage, and all their expiry dates are in the past, then items are moved to the next stage that contains the expire action.
- A notification is sent to all localized content owners when documents leave this stage.

Deletion synchronization

- If a base item is deleted the localized content must also be deleted. To handle this task, another special workflow stage is required. This stage is placed after the expire stage if content expiration is used, or after the publish stage if content expiration is not used.
- To delete an item, you need to push the item to the final workflow stage. In most systems, only an administrator has access to do this. When processing

documents in this stage, if the base item is detected, or no base item exists, then all draft, published and expired localized copies are removed and a notification sent to all localized content owners to inform them of the deletion.
- If a localized item is detected and the base item is in another stage, then a deletion request notification is sent to the localized content owners to inform them that various localized copies are ready to be deleted.

Multilingual presentation

Localizing the presentation layer in your site involves deciding which parts of the presentation layer need to be localized, and then separating out those pieces into components that can be localized.

Presentation templates

In many cases the same presentation templates are used in all locales, with fragments of the template being varied by using one of these two options:

- Localized elements can be referenced from the current site area.
- Localized components can be referenced by prefixing the name attribute in the component tag with `name="./item"`. By using dot notation in component tags, the library name is not resolved from the current library until the item is rendered. For example:

```
<component name="./my_component" />
```

In other cases however, a locale requires such different presentation that an entirely different template is required. For example, to support locales with languages that read right to left, or for locales with different branding.

These strategies can all be used together, by setting up template mappings in each localized site, and by using site area elements or dot notation where appropriate.

Site area elements

Using site area elements to vary the locale presentation is used for parts of the presentation that are site-specific content.

Reasons to use site area elements:

- A user must be a site manager to edit these elements.
- The site manager has the option to select different components.
- If the site area is workflowed, all of the modified elements can be previewed at once.
- Suitable for multiple sites per locale.

Reasons to not use site area elements:

- Not a good solution for fragments of the presentation template that are under the control of the designer rather than the site manager.
- Using more than 12 elements can cause scalability issues.

Using dot notation to reference components from the current library

The `name="./item"` parameter is used in component tags for locale-specific design variations in a presentation template.

Reasons to use dot notation in tags:

- Changes can be limited to the users that manage presentation.
- Scales better than using site area elements.

Reasons to not use dot notation in tags:

- Changes cannot be easily previewed.
- References cannot be easily tracked.

Localized navigator components

When creating a navigator, if the **Current site area** or **Current content** options are used, you can reuse navigators across multilingual sites.

Navigators that use selected start positions cannot easily be shared across locales. To use these types of navigators, the `name="./item"` parameter must be used to retrieve localized components.

Another way of localizing a navigator is to use a JSP fragment to set a different context before rendering the navigator. The navigator would be set to use the **Current site area** as the starting point, and by changing the context that you can manipulate this start area by using code. Because the code uses a path lookup, the same JSP fragment works in all of your localized sites.

```
<%@ page import="com.ibm.workplace.wcm.api.*" %>
<%@ taglib uri="/WEB-INF/tld/wcm.tld" prefix="wcm" %>
<wcm:initworkspace user="<%= request.getUserPrincipal() %>" />
<%
RenderingContext context = (RenderingContext)request.getAttribute(Workspace.WCM_RENDERINGCONTEXT_KEY);
String currentLibrary = context.getLibrary().getName();
String navStart = currentLibrary+"/mySiteArea";
%>
<wcm:setExplicitContext path="<%=navStart%>"/>
<wcm:libraryComponent name="theNavigator" library="theLibrary" />
```

In this example:

- `./mySiteArea` is the path.
- `theNavigator` refers to the navigator component.
- `theLibrary` refers to the library the navigator is stored in.

Localized menu components

Menus that use the **Current content site area** option can be reused across multilingual sites.

Menus that use selected site areas cannot easily be shared across locales. To use these types of menus, the dot notation parameter must be used to retrieve localized versions.

Another way of localizing a menu is to use a JSP fragment to allow a list of site areas to be specified at render time. Because the path is the same for each locale, the same JSP fragment works in all of your localized sites.

```
<%@ page import="com.ibm.workplace.wcm.api.*, java.util.*"%>
<%@ taglib uri="/WEB-INF/tld/wcm.tld" prefix="wcm" %>
<wcm:initworkspace user="<%= request.getUserPrincipal() %>" />
<%
RenderingContext context = (RenderingContext)request.getAttribute(Workspace.WCM_RENDERINGCONTEXT_KEY);
String currentPath = context.getPath();
String currentLibrary = context.getLibrary().getName();
Map myparams = new HashMap();
myparams.put("SiteAreas", currentLibrary+"/mySite/mySiteArea1,"+currentLibrary+"/mySite/mySiteArea2");
%>
<wcm:setExplicitContext path="<%=currentPath%>" requestParameters="<%=myparams%>"/>
<wcm:libraryComponent name="theMenu" library="theLibrary" />
```

In this example:

- `theMenu` refers to the menu component.
- `theLibrary` refers to the library the navigator is stored in.
- `SiteAreas` contains the list of site areas to specify in the code.

Localized text within common design components

If you have common design components that you want to reference localized text into, then it is recommended to write a Rendering plug-in that wraps a Java Resource Bundle, with the Resource Bundle Key name and the bundle name that is supplied to the plug-in as arguments.

Localized site search

1. Create a search collection for each locale specifying the appropriate language.
2. Create a content source for each site.
3. Create a search component for each locale library selecting the relevant search collection, or use the Search and Browse portlet.

Localized rendering

Localized rendering is provided by either automatically matching the user locale with the content locale, or by providing a navigation option in the site to allow the user to choose the locale for themselves. These two strategies might also be used together in some cases.

Automatic localization

Automatic localization is when there are different versions of a page for each page and a user is automatically redirected to one of them, based on their locale. The locale can be determined by the following methods:

URL The url can contain the locale. For example, ibm.com.au or ibm.co.uk.

Browser preference
Browsers allow you to specify your language preferences. Language preference information is then sent by the browser in the Accept-Language http header. For example: Accept-Language: en-ca,en-us;q=0.7,en;q=0.3

Portal The portal determines the language for rendering portal content by a search process along the following sequence at login time:

1. If the user has logged in, the portal displays the preferred language that is selected by the user.
2. If no preferred user language can be found, the portal looks for the language in the browser. If the portal supports that language, it displays the content in that language. If the browser has more than one language, the portal uses the first language in the list to display the content.
3. If no browser language can be found, for example if the browser used does not send a language, the portal uses its own default language.
4. If the user has a portlet that does not support the language that was determined by the previous steps, that portlet is shown in its own default language.

User-selected localization

User selected localization is when the user selects a language manually from a link or selection list. The selection can then be stored in a cookie for subsequent visits to the website.

Locale selection page
In this scenario, a launch page is shown when a user first comes to the site that shows what locales are available and user then selects the appropriate one.

Equivalent page navigation
In this scenario, navigation is displayed that allows the user to see the same page in a different locale on every page in the site.

Encoding

UTF-8 (Unicode) is recommended as the encoding for your page, since this encoding supports all the characters you need. If you cannot use UTF-8, then you should use escapes to represent characters that are not supported by the encoding of your page. As all content is stored in the JCR database with UTF-8 encoding, character data is preserved.

WebSphere Portal sites should use the encoding that is set up in the administration portlet. A servlet rendered site should use the encoding that is specified within WebSphere Portal. If a site needs to have more than one encoding, multiple servers are required each with the different encoding settings. A gateway is then used to convert the encoding.

Font face considerations

If implementing a multilingual site, it is worth considering what font to use. There are a few font faces in windows that are installed automatically and can show multilingual characters. One of the best font faces is Tahoma. It is easy to read and contains all the Unicode characters.

Related tasks:

"Adding a new language to render localized content" on page 1425
You can add new languages to the portlet to render localized content in different languages and reach a larger audience.

Creating more multilingual sites

In many cases, the number of languages or regions you require for a site is fixed, but there are times when you need to roll out new locales as your organization grows.

Procedure

1. Use the **ML Library Copy** portlet to copy either the base locale site or one of the translated sites that more closely matches the new site, assigning it a new name and locale during the copy.
2. Translate the copied items as required.
3. Update the relevant multilingual configuration file to reference your new library.

What to do next

When the initial translation is complete, you can run user acceptance tests before you deploy your new site and going into the day to day maintenance that is enabled through automated processes.

You can temporarily disable the automated syncing of this new site while it is going through its initial translation and testing phases.

Advanced options

There are more points of extensibility where you can enhance the multilingual solution to suit your specific requirements.

Mixed language site navigation

In a mixed language site, the navigation is structured to make it possible for users to easily discover what content is available for them in their preferred language, while they maintain the structure of the base language.

- For content that is available in complete translations, the localized navigation code can be used to show a link to the translated content.
- For content that is not available, a menu can be built to pull content from the translated site. For example, the menu might display a list of "Translated content in this area".
- The opposite can be done when you view translated content. For example, a menu might display a list of "Content in this area that is not translated".
- A site map can be built for each translation to provide a shortcut for users to find translated content. This action can be built by using a menu or navigator to explicitly pull content in from the translated site.
- The base site navigation is kept consistent on the page by building navigators that are set to the base site. When translated content is being displayed, the presentation template shows the base navigation always.

Side-by-side preview

Side by side previews might be built by using a special preview page with two web Content Viewer portlets.

- Custom rendering code would be required to pick up two contexts at the same time and display both pages, perhaps in a frame.
- A link to start this preview page might be added to the edit-time navigation or creation extension.

Side-by-side editing

Side by side editing might be built by using a special editing page with containing one authoring portlet and one web content viewer portlet.

- A link might be created that combines the URL parameters for displaying content within the web content viewer portlet with the URL parameters for opening another item in read or edit mode in the authoring portlet
- A link to start this preview page might be added to the edit-time navigation or creation extension.

Writing your own context processor

Provide your own processing whenever the web content viewer portlet renders items.

- Create a **ContextProcessor** class, which implements the com. com.ibm.workplace.wcm.api.ContextProcessor interface.
- Compile the **ContextProcessor** class into a new jar and place that jar in the *wp_profile_root*\wcm\shared\app directory.

Customizing the workflow synchronization email notifications

Modify the default email notifications to provide your own wording.

1. Go to *wp_profile_root* \paa\wcm_mls\components\wcm_mls\shared\app\ directory
2. Extract the contents of **wcm.ml.emailnotifications.jar** to *wp_profile_root*\ wcm\ prereq.wcm\wcm\shared\app .
3. Remove the *wp_profile_root*\wcm\prereq.wcm\wcm\shared\app\META-INF directory.
4. Customize the workflow synchronization email notifications:

Table 389. Localize / Regionalize email notifications (syncupdate.properties)

Message Key	Description
NEWDOCCREATED	The notification when a new translation is created.
LINKCREATED	The notification when a link is created within a localized or regionalized library.
DRAFTCREATED	The notification when a draft is created of any existing localized or regionalized item.
DRAFTEXISTS	The notification when a draft exists within a localized or regionalized library.
DRAFTRENAMED	The notification when a draft is renamed within a localized or regionalized library.
DRAFTMOVED	The notification when a draft is moved within a localized or regionalized library.
FAILED_NOPROCESSSETTING	The notification when no processing setting is found for a given localized or regionalized library
FAILED_CREATENEWDOC	The notification when a new translation fails to be created.
FAILED_CREATEDRAFT	The notification when a draft of an existing translation fails to be created.
FAILED_CREATELINK	The notification when a link fails to be created.
FAILED_CALBASELOCALEDOCPATH	The notification when the path to the current base locale path cannot be determined.
FAILED_RENAMEDRAFT	The notification when a draft fails to be renamed.
FAILED_MOVEDRAFT	The notification when a draft fails to be moved.

Table 390. Publishing Synchronization email notifications (syncpublish.properties)

Message Key	Description
ITEM_PUBLISHED	The notification when a translation is published.

Table 391. Expiry Synchronization email notifications (syncexpiry.properties)

Message Key	Description
ITEM_EXPIRED	The notification when a translation is expired.

Table 392. Delete Synchronization email notifications (syncdelete.properties)

Message Key	Description
ITEMDELETED	The notification when a translation is deleted.
SENTFORDELETION	The notification when a translation is sent for background deletion.
DELETEREQUEST	The notification when a localized translation is being requested to be deleted.
REGIONAL_DELETEREQUEST	The notification when a regionalized translation is being requested to be deleted.
DELETEFAILED	The notification when a translation deletion fails.

5. Modify the message by using any of the following multilingual tags:

Table 393. Multilingual tags

Message Key	Description
[LOCALE]	The locale of the current item.
[ITEM_ID]	The API ID of the current item.
[ITEM_LIBRARY]	The name of the library that the current item is in.
[ITEM_PATH]	The path of the current item.
[ITEM_PATH_LINK]	A link to open the current item within the authoring UI by using the path of the item as the link name.
[ITEM_NEW_NAME]	The new name of the current item, if the item was renamed.
[ITEM_OLD_NAME]	The old name of the current item, if the item was renamed.
[ITEM_NEW_PATH]	The new path of the current item, if the item was moved.
[ITEM_OLD_PATH]	The old path of the current item, if the item was moved.
[BASELOCALE_ITEM_ID]	The API ID of the current base locale item.
[BASELOCALE_ITEM_LIBRARY]	The name of the library that the current base locale item is in.
[BASELOCALE_ITEM_PATH]	The path of the current base locale item.

Table 393. Multilingual tags (continued)

Message Key	Description
[BASELOCALE_ITEM_PATH_ LINK]	A link to open the current base locale item within the authoring user interface by using the path of the item as the link name.
[ERRORS]	The current list of errors that are processing the current item.

6. Save your changes.
7. Translate your message and update the localized versions of the properties files.
8. Restart the server.

Extensions for multilingual sites

The multilingual solution uses a set of extensions that can be used with existing features to author, manage, and publish your multilingual websites.

"Edit-time navigation creation extension"
This extension provides a way to change between localizations of the same item and to create new items in localized libraries.

"Update notification extension" on page 2465
This extension uses the settings in the workflow synchronization section of the multilingual configuration file to automate the creation of localized items.

"Publishing synchronization extension" on page 2466
This extension uses projects to ensure that the base item and any draft localized items are published at the same time.

"Expiry synchronization extension" on page 2467
This extension ensures that the base item and any published localized items are expired at the same time.

"Deletion synchronization extension" on page 2467
This extension ensures that the base item and any draft localized items are deleted at the same time.

"Servlet render-time navigation extension" on page 2468
This extension provides navigation between equivalent published content in different locales from within a presentation template at rendering time.

"Portlet Render-time navigation extensions" on page 2468
This extension enables the Web Content Viewer to become locale aware and thus automatically switch to an equivalent object based on the current user's locale.

"Domain locale redirection extension" on page 2469
This extension provides a redirection from the main domain to another locale based on the locale settings of the current user.

"Library copy portlet extension" on page 2470
This extension is used as part of setting up a new locale by copying an existing library, and assigning it a new name and locale.

Edit-time navigation creation extension

This extension provides a way to change between localizations of the same item and to create new items in localized libraries.

The extension has two versions that differ only when the extension itself renders its results. The Auto Load version renders immediately, whereas the Manual Load version requires the user to click **Refresh** before any results are returned.

To use the edit-time navigation creation extension:

1. Add a text element that is named **ML Translations** to an authoring template.

   Note: CF07 This element is displayed in the content item form only if the Multilingual Solution is enabled. Otherwise, this element is hidden in the content item form, but still visible in the **Manage Elements** dialog.

2. Edit the element properties and add the following code to the custom JSP field:

   For the Auto load version before version 8.5 CF3

   ```
   readMode=/wps/wcmml;/jsp/html/MLAuthorTimeRead_AutoLoad.jsp,
   editMode=/wps/wcmml;/jsp/html/MLAuthorTimeEdit_AutoLoad.jsp
   ```

   CF03 **For the Auto load version for version 8.5 CF3 or higher**

   ```
   readMode=[wcm.mls.context.path];/jsp/html/MLAuthorTimeRead_AutoLoad.jsp,
   editMode=[wcm.mls.context.path];/jsp/html/MLAuthorTimeEdit_AutoLoad.jsp
   ```

   For the Manual load version before version 8.5 CF3

   ```
   readMode=/wps/wcmml;/jsp/html/MLAuthorTimeRead_ManualLoad.jsp,
   editMode=/wps/wcmml;/jsp/html/MLAuthorTimeEdit_ManualLoad.jsp
   ```

   CF03 **For the Manual load version for version 8.5 CF3 or higher**

   ```
   readMode=[wcm.mls.context.path];/jsp/html/MLAuthorTimeRead_ManualLoad.jsp,
   editMode=[wcm.mls.context.path];/jsp/html/MLAuthorTimeEdit_ManualLoad.jsp
   ```

3. Save the authoring template and reapply the template to all content that uses it ensuring the **Add new elements** option is selected.

How it works

Every time that you open a content item that uses this extension in the authoring portlet, all libraries that are configured in the current multilingual configuration file are searched for all matching content. To be matching, it must be in the same site area path and have the same name.

If a matching content item is found, then information about the content, item including a link to open that item in read mode, is displayed. If the existing content item is a link, then information about the originating content item is displayed instead.

If no matching content items are found, and you have edit access to the content item, you can either create a copy or link of the content item in each localized library:

Copy Clicking **Copy** creates a draft with the same name as the current item under the equivalent site area in the localized library.

Link Clicking **Link** creates a content link to the current content under the equivalent site area in the localized library.

Note: When performing copy or link operations against an item within the Portal Site library, any pages on the path to the item being copied will be copied as site areas in the destination library.

Update notification extension

This extension uses the settings in the workflow synchronization section of the multilingual configuration file to automate the creation of localized items.

To use the notification extension:

1. Add either the **MLConfiguration_v7 \ Localize** or **MLConfiguration_v7 \ Regionalize** workflow stage to your workflow after the approval stage, which is the first stage after the draft stage.
2. If any of your localized library sets do not have a base locale:
   a. Ensure that **storeOriginalWorkflow** is set to **true** within the **General Settings** of the associated multilingual configuration file.
   b. For each workflow used by translated content, create another workflow that is named ***WorkflowName*_Translation**, where *WorkflowName* is the name of the original workflow. This workflow must contain a draft stage, approval stage, and the **ML Configuration_v7 \ Switch Workflow** stage.
      - If legacy synchronized publishing is to be used, then use the **ML Configuration_v7 \ Pending Published – No Base Locale** stage instead of **ML Configuration_v7 \ Switch Workflow**.
      - If legacy synchronized publishing is not being used, then the **ML Configuration_v7 \ Switch Workflow** stage must also be added to the base workflow after the approval stage.
3. If any of your localized library sets are going to use localized authoring templates, ensure that the **supportLocalizedAuthoringTemplates** property in the **Localize** or **Regionalize** workflow synchronization settings of your multilingual configuration file is set to true.

How it works

When an item reaches the **Localize** or **Regionalize** workflow stage:

- If the base locale item is in the current stage or a later stage, then each configured library that does not already have a draft item is processed according to the **Localize** or **Regionalize** workflow synchronization settings of your multilingual configuration file and a notification is sent to the all relevant locale owners.
  - If the library that is being processed contains an existing published or expired item, then the workflow synchronization settings are ignored and a draft copy of the existing item is created.
  - If the library that is being processed does not contain an existing published or expired item, then it is either ignored, or a new draft or link is created depending on the configuration that is specified in the multilingual configuration file.
- If one or more localized copies of the current item are found that do not have a corresponding draft base locale item in any stage of the workflow, then all copies are moved to the next workflow stage and a notification is sent to the base locale owner to inform them that changes have been made to localized documents without an associated change to the base locale item.
- If the base locale item is in an earlier stage of the workflow, then nothing happens.

Note: If no base locale library is configured, then the current item is deemed to the base item for the current processing cycle only.

Note: If the base item is renamed or moved, then all localized drafts and regionalized drafts are moved or renamed

Publishing synchronization extension

This extension uses projects to ensure that the base item and any draft localized items are published at the same time.

To use the project synchronized publishing extension, set **syncPublish.useProjects=true** in the multilingual configuration file.

Note: Individual workflows can be excluded from using synchronized publishing by specifying their name in the **SyncPublish.workflowsToExcludeFromSyncProjectPublishing** setting in the multilingual configuration file.

Note: The **syncPublish.useProjects** and **SyncPublish.workflowsToExcludeFromSyncProjectPublishing** settings must be the same across all associated multilingual configuration files.

How it works

When an item enters a publish workflow stage the following things happen:

- If any draft localized copies of the current item are not in the pending publish state, then nothing happens.
- If all draft localized copies of the current item are in the pending publish state, then the current item and its copies are moved to the next workflow stage and a notification sent to relevant locale owners.

Expiry synchronization extension

This extension ensures that the base item and any published localized items are expired at the same time.

To use the extension, add the **MLConfiguration_v7 \ Pending Expire** workflow stage to your workflow before the expire stage.

How it works

When an item reaches the pending expire state the following things happen:

- If any published localized copies of the current item are not in the pending expire state, nothing happens.
- If all published localized copies of the current item are in the pending expire state, then the current item and its copies are moved to the next workflow stage and a notification are sent to the relevant locale owners.
- The expiry extension is only applied to individual workflows, not project workflows.

Deletion synchronization extension

This extension ensures that the base item and any draft localized items are deleted at the same time.

- To use the extension, add the **MLConfiguration_v7 \ Delete** workflow stage to the end of your workflow.
- If your workflow is not using expiry, then instead use the **MLConfiguration_v7 \ Delete – For non expiry workflows** workflow stage.

How it works

When an item reaches the delete workflow stage the following things happen:

- If there is a base locale copy of the current item and it is not in the delete workflow stage, then a deletion request email is sent to the base locale owner to inform them that one or more localized copies need to be removed.

- If the base locale copy of the current item exists in the delete workflow stage, then all available draft and published localized copies are removed and a notification sent to relevant localized library owners, regardless of the workflow stage the localized copies are in.
- The delete extension is only applied to individual workflows, not project workflows.

Servlet render-time navigation extension

This extension provides navigation between equivalent published content in different locales from within a presentation template at rendering time.

To add localized rendering links to content items, reference the JSP component that is named **MLConfiguration_v7/JSP - ML Locale Nav** to either the associated presentation template, or as a reference within a component or element that is also referenced in the presentation template.

How it works

- Every time that you render a page that includes the **MLConfiguration_v7/JSP - ML Locale Nav** component, all libraries that are configured in the multilingual configuration file are searched for all items that match the current content item.
- To be matching, the content must be located under an equivalent site path and have the same name.
- A link is created for each matching localized content item that is displayed in the language of the current user.
- While this extension does work for both servlet-rendering and portlet-rendering, in portlet-rendering it add links only for content displayed in the current portlet.

Portlet Render-time navigation extensions

This extension enables the Web Content Viewer to become locale aware and thus automatically switch to an equivalent object based on the current user's locale.

To enable this extension:

1. Update all Web Content Viewer Portlets to reference the multilingual context processor.

   Web Content Viewer
   Edit the portlet settings and select the `com.ibm.workspace.wcm.ml.contextProcessor.MLContextProcessor` from within the **Plugins** section of the **Advanced Options**.

   Legacy Web Content Viewer Portlet
   Add the following property to the configuration of the Web Content Viewer Portlet:

   ```
   ContextProcessorClass=com.ibm.workplace.wcm.contextprocessor.MLContextProcessor
   ```

2. Modify all of the pages in the site to add a web content associations for the localized sites:
   a. Go into edit mode.
   b. Open the toolbar.
   c. Select **Page**.
   d. Select **General**.
   e. Select **Details**.
   f. Select "Default site area".

g. Add the web content associations to the site areas in the localized sites that are equivalent to the existing associated site area in the base locale.
h. Click **OK**.

How it works

- When a Web Content Viewer Portlet is rendered, the multilingual context processor checks whether the locale has changed. If the locale has changed, then an equivalent item in the new locale is displayed.
- If a presentation template is selected in the Web Content Viewer configuration, then a localized presentation template is also searched for and used if available.
- The multilingual portal local switcher can be used to allow the user to select a locale in which to view the site. This selected locale overrides the locale that the user specifies in their browser or in their portal preferences. To enable this extension:
  1. Integrate the **MLPortalLocaleSwitcher.jsp** from the **wcm-multilocale.ear** into your theme.
  2. Place a Web Content Viewer portlet on the page that is configured to display the **JSP - ML Portal Locale Switcher** component from the **MLConfiguration_v7 library**.
     – Select a content item from one of the localized libraries as the content context.
     – Set "**Broadcast links to**" to "**This Page**".

Related concepts:

"Web content associations" on page 2023
Web content associations are used to combine portal pages and associated web content items that are managed by IBM Web Content Manager so they can be managed and rendered consistently. Web content associations map portal pages to the site structure in the IBM Web Content Manager system.

Domain locale redirection extension

This extension provides a redirection from the main domain to another locale based on the locale settings of the current user.

To use this extension:

1. Edit the `MLServletHomePageRedirection.jsp` in the `wcm-multilocale.ear`:
   - Comment out the lines in the **`getAvailableLocalesList`** method.
   - Change **`s_defaultLocale`** to match the default locale.
2. Update your HTTP Server. For example, if you are using IBM HTTP Server edit the following configuration settings:
   - Add the following text to the end of the IBM HTTP Server configuration file:

     Before version 8.5 CF03

     ```
     LoadModule rewrite_module modules/mod_rewrite.so
     RewriteEngine on
     RewriteRule ^/$ /wps/wcmml/jsp/html/MLServletHomePageRedirection.jsp [PT]
     ```

     CF03 Version 8.5 CF03 or higher

     ```
     LoadModule rewrite_module modules/mod_rewrite.so
     RewriteEngine on
     RewriteRule ^/$ /MLS_CONTEXT_ROOT/wcmml/jsp/html/MLServletHomePageRedirection.jsp [PT]
     ```

     Where *MLS_CONTEXT_ROOT* is the context root for the multilingual solution application.
   - For each locale in the **s_availableLocales** list, also add the following text to the end of the IBM HTTP Server configuration file:

Before version 8.5 CF03

```
RewriteRule ^/LOCALE$ /wps/wcm/connect/LIBRARY/SITE [PT]
```

- LOCALE: The string representation of the corresponding Java Locale object. For example,"fi" for Finnish and "pt_BR" for Brazilian Portuguese.
- LIBRARY: The web content library that is associated with the locale
- SITE: The top-level site area for the specified library

For example: `RewriteRule ^/en$ /wps/wcm/connect/english/ Internet [PT]`

Important: Never map any of the locale redirections to the main domain.

CF03 **Version 8.5 CF03 or higher**

```
RewriteRule ^/LOCALE$ /PORTAL_CONTEXT_ROOT/wcm/connect/LIBRARY/SITE [PT]
```

- LOCALE: The string representation of the corresponding Java Locale object. For example: "fi" for Finnish and "pt_BR" for Brazilian Portuguese.
- PORTAL_CONTEXT_ROOT: The context root for WebSphere Portal.
- LIBRARY: The web content library that is associated with the locale
- SITE: The top-level site area for the specified library

For example: `RewriteRule ^/en$ /wps/wcm/connect/english/ Internet [PT]`

Important: Never map any of the locale redirections to the main domain.

How it works

Every time that you request the top-level domain the locale of the current user, or the ordered list of preferred languages set in the browser, is checked against the list of available locales as in the JSP. The JSP then redirects back to the top-level domain plus the available locale, and the web server then redirects to the correct content item based on that locale.

While this extension is designed for servlet-rendering, it can be used with portlet-rendering to render different locales from different servers.

Library copy portlet extension

This extension is used as part of setting up a new locale by copying an existing library, and assigning it a new name and locale.

To use the extension:

1. Click the **Administration menu** icon. Then, click **Portal Content** > **ML Library Copy**.
2. Select a library to copy.
3. Select a new locale.
4. Type a new name for the library.
5. Click **Copy**.

Note: The **Portal Site** library cannot be cloned by using the **ML Library Copy** portlet, therefore if your base locale is the **Portal Site** library you must either:

- Use the "Edit-time navigation creation extension" on page 2464 to manually copy each of your content items in the **Portal Site** library to your localized libraries. This action automatically duplicates the page path as site areas in the destination library.
- Write some Web Content Manager API code to manually copy the **Portal Site** library by traversing the page hierarchy and creating site areas in the destination library with the same name as the page, then copying the content items by using the Web Content Manager API `Workspace.copy` method.

IBM Syndicated Feed Portlet for WebSphere Portal Express

The new IBM Syndicated Feed Portlet for IBM WebSphere Portal Express offers enhanced feed subscription and presentation capabilities.

Using IBM Syndicated Feed Portlet, a user can:

- Integrate, view, and manage RSS and ATOM feeds from portal pages
- Organize the feeds into new and existing feed categories
- Specify authentication options and credentials (private or shared) for password-protected feeds
- Extensively customize the presentation style for feeds
- Export the feeds of IBM Syndicated Feed Portlet to an Outline Processor Markup Language (OPML) compliant xml file and import multiple feeds by specifying an appropriate OPML URL.
- Enable and configure servlet and dynamic caching for caching the portlet user interface and feed data, respectively

Using IBM Syndicated Feed Portlet, an administrator can perform all the functions that a user can. Additionally, the administrator can:

- Regulate a user's ability to customize the portlet
  - Apply locks on a user's ability to perform feed subscription related functions, such as
    - Adding feeds to the portlet
    - Enabling or displaying the feeds in the view mode of the portlet
    - Configuring cache settings
  - Apply locks on a user's ability to customize feed presentation related options, such as
    - Choosing a presentation style for the feeds
    - Displaying or hiding the toolbar. A toolbar enables you to perform all the operations related to feeds and feed categories from within the view mode.
    - Displaying or hiding the channel bar. The channel bar displays the name of the feed category and the name of the feed subscription separated by a forward slash (/).
    - Customizing the portlet window title to be the same as the default portlet title, the title of the first selected feed, or a custom title
    - Displaying or hiding the author of feed items
    - Displaying or hiding the Published date of feed items

You can perform all the listed tasks from within the IBM Syndicated Feed Portlet. Steps for performing the previous tasks are documented in the WebSphere Portal Express help.

The following configuration tasks need to be performed by an Administrator from WebSphere Portal Express, IBM WebSphere Application Server, or the WebSphere Portal Express file system:

- Configure settings for a proxied environment
- Configure a user's ability to customize the portlet window title
- Enable display of SSL-secured feeds
- Configure global cookies for feeds.

"Additional information for using IBM Syndicated Feed Portlet"
When using IBM Syndicated Feed Portlet for WebSphere Portal it is important to consider additional information about authentication methods, feed details, and settings.

"Configuring proxy settings" on page 2473
IBM Syndicated Feed Portlet for IBM WebSphere Portal Express is supported in a proxied environment.

"Customizing the portlet title" on page 2473
With the appropriate configuration settings, you can customize the portlet window title to be the same as the default portlet title, the title of the first selected feed, or a custom title that you specify in the portlet configuration panel.

"Enabling display of SSL-secured feeds" on page 2475
When you use IBM Syndicated Feed Portlet, some feeds that use HTTP over SSL (HTTPS) might not function properly because of missing SSL certificates. You might encounter errors, such as `Missing certificate`, `Unknown certificate`, or `Untrusted certificate` because of missing root signer certificates or missing self-signed certificates in the certificate truststore. In both the cases, you must add the appropriate certificates to the certificate truststore.

"Configuring cookies and active content filtering" on page 2476
You can configure the IBM Syndicated Feed portlet to forward HTTP cookies with outbound requests to feed sources, or restrict the portlet to forward cookies only to specific domain names. You can also enable active content filtering for all feeds to remove active content such as Java script from the feed text.

"Client side aggregation (CSA) rendering in IBM Syndicated Feed Portlet" on page 2477
IBM Syndicated Feed Portlet can be configured to operate in the client side aggregation (CSA) rendering mode. Client side rendering ensures an improved user experience through faster response time. When you enable client side rendering in the portlet, the portal does not re-render the whole page but only the aspects of the portlet that change.

Additional information for using IBM Syndicated Feed Portlet

When using IBM Syndicated Feed Portlet for WebSphere Portal it is important to consider additional information about authentication methods, feed details, and settings.

Consider the following information when using the IBM Syndicated Feed Portlet:

- IBM Syndicated Feed Portlet supports the following authentication methods:
  - Basic authentication

  - Form-based authentication
  - Single Sign-On (SSO) via LTPA and Netegrity SiteMinder
- The xml file from which you export the feed details for one instance of IBM Syndicated Feed Portlet can not be used to import the feed details into another instance of the portlet.
- Initially, you can view portlet feeds in the View mode based on the settings configured by the Administrator. If you modify the settings, they take precedence over the subsequent settings that the Administrator configures.
- If you encounter problems viewing feeds that use international character sets, refer to the technote, "Viewing feeds using international character sets in the IBM Syndicated Feed Portlet for Websphere Portal", for more information.

Related information:

 Viewing feeds using international character sets from the IBM Syndicated Feed Portlet for WebSphere Portal

Configuring proxy settings

IBM Syndicated Feed Portlet for IBM WebSphere Portal Express is supported in a proxied environment.

About this task

To specify the details of proxy server, proceed as follows:

Procedure

1. Click the **Administration menu** icon in the toolbar. Then, click **Portlet Management** > **Portlets**.
2. Search for the Syndicated Feed portlet entry.
3. Click the **Configure portlet** (wrench) icon for the Syndicated Feed Portlet.
4. Add the following configuration parameters to the existing list of configuration parameters for the Syndicated Feed Portlet.

Table 394. Configuration parameters

Parameter	Value
`proxyHost`	Specify the proxy server hostname
`proxyPort`	Specify the proxy server port number

To add these configuration parameters:

a. Type the parameter name in the **New Preference** text field.
b. Type the value of the parameter in the **New value** text field.
c. Click **Add**.

Note: You do not need to restart the portal server after you specify the configuration parameters. The updated configuration becomes effective without restart.

Customizing the portlet title

With the appropriate configuration settings, you can customize the portlet window title to be the same as the default portlet title, the title of the first selected feed, or a custom title that you specify in the portlet configuration panel.

About this task

The IBM Syndicated Feed portlet provides dynamic window title support that implements the request attribute, `com.ibm.portal.portlet.Constants.DYNAMIC_TITLE`. WebSphere Portal Express defines the title bar for portlets in `control.jsp`. To add dynamic title support for the IBM Syndicated Feed portlet, you must deploy the dynamic window title support and update `control.jsp`.

Because the *PortalServer_root* directory is read only, you cannot modify the window title support libraries, `control.jsp`, or other theme files in this directory. You should include these files as part of a custom theme or deploy the files as a separate Web Archive (WAR). See the following topics for more information: *Location of theme resources* and *Creating a new theme*.

Procedure

1. Deploy the dynamic window title support with your custom Web Archive (WAR).
   a. Navigate to the following directory: *PortalServer_root*`/bp/wp.bp.feedspace/installableApps`

      Note: If you downloaded the IBM Syndicated Feed portlet from the IBM Business Solutions Catalog, the required files are available in the root directory of the archive.
   b. Locate and copy the following files to a work directory:
      - `dynamicWindowTitle.jar`
      - `dynamicWindowTitle.tld`
   c. Copy `dynamicWindowTitle.jar` to the following directory in your custom WAR: `WEB-INF/lib`
   d. Copy the file `dynamicWindowTitle.tld` to the following directory in your custom WAR: `WEB-INF/tld`
2. Customize the skin(s) that you plan to use with the IBM Syndicated Feed portlet. Perform the following steps for each skin you plan to use:
   a. Locate `control.jsp` in the `skin` directory of the WAR file.
   b. Locate the last `taglib` definition in the file. For example,
      ```
      <%@ taglib uri="/WEB-INF/tld/dnd.tld" prefix="dnd" %>
      ```
   c. Insert the following string after that tag:
      ```
      <%@ taglib uri="/WEB-INF/tld/dynamicWindowTitle.tld" prefix="dwt" %>
      ```
   d. Locate the following string:
      ```
      <portal-skin:portletTitle>
      ```
      If the file does not contain this string, insert it after the last `taglib` definition.
   e. Insert the following string before this tag:
      ```
      <dwt:renderTitle windowId="<%=myPortletID%>">
      ```
   f. Locate `</portal-skin:portletTitle>`.
   g. Insert the following string after this tag:
      ```
      </dwt:renderTitle>
      ```
   h. Add the following tag to the end:
      ```
      <dwt:setTitle windowId="<%=myPortletID%>"/>
      ```

i. Save and close `Control.jsp`. The changes you make to `Control.jsp` should be similar to the following:

```
<%@ taglib uri="/WEB-INF/tld/dnd.tld" prefix="dnd" %>
<%@ taglib uri="/WEB-INF/tld/dynamicWindowTitle.tld" prefix="dwt" %>
::::::::::::
::::::::::::
<dwt:renderTitle windowId="<%=myPortletID%>">
<portal-skin:portletTitle>
<portal-fmt:problem bundle="nls.problem"/>
</portal-skin:portletTitle>
</dwt:renderTitle>
<img alt="" style="border:0; text-align: <%= bidiAlignRight %>;"
width="1" height="22"
src='<portal-logic:urlFindInSkin file="title_minheight.gif"/>'>
</div></dnd:dragHandle></td>
<td class="wpsPortletIcons">
<%-- Do not include 'Skip to next portlet' link in --%>
<%-- standalone window or solo mode --%>

<%if(isJSAvail){%>
::::::::::::::::::::::
::::::::::::::::::::::

<dwt:setTitle windowId="<%=myPortletID%>" />
```

What to do next

You can now use the dynamic portlet window title customization option with all IBM Syndicated Feed portlet instances inside the skins you customized.

Note: The IBM Syndicated Feed portlet disables the portlet window title customization options in the Personalize/Edit Shared Settings/Configure modes when the portlet detects that it is displayed in a skin that does not support the dynamic window title features. The IBM Syndicated Feed portlet also disables these options if an administrator locks the window title features.

For more information, see *Tip: Changing a portlet title at run time in WebSphere Portal V6*.

Related information:

IBM WebSphere Portal Business Solutions Catalog

Enabling display of SSL-secured feeds

When you use IBM Syndicated Feed Portlet, some feeds that use HTTP over SSL (HTTPS) might not function properly because of missing SSL certificates. You might encounter errors, such as `Missing certificate`, `Unknown certificate`, or `Untrusted certificate` because of missing root signer certificates or missing self-signed certificates in the certificate truststore. In both the cases, you must add the appropriate certificates to the certificate truststore.

Before you begin

To add the appropriate certificates, you must have Administrator access to your IBM WebSphere Portal Express installation.

Procedure

1. Obtain the missing certificates.

There are various mechanisms for doing this depending on the type of certificate that is missing. In general, the quickest way to obtain the missing certificate is to access the feed you are trying to add directly from a web browser. All modern browsers provide the capability to view a certificate and its trust chain. However, the directions for viewing and exporting certificates from a web browser are browser vendor specific and therefore outside the scope of this document (except to say that they should be exported in Base-64 encoded format as a file with the extension .cer or .arm). If you need assistance with this task, contact your IT administrator. There are also many online resources that can help you complete this task.

Note: If you use self-signed certificates, you might be prompted with several warnings about the acceptance or trust of this certificate. In such cases, ensure that you trust the third-party server before you add the certificate to your certificate truststore.

2. Once you export the appropriate root signer or self-signed certificate, refer to the instructions in the Adding a signer certificate to a keystore topic to import the root signer certificate or to import the self-signed certificate as a new trusted signer certificate to the appropriate truststore.
3. Restart the WebSphere_Portal server.

Related information:

Adding a signer certificate to a keystore

Configuring cookies and active content filtering

You can configure the IBM Syndicated Feed portlet to forward HTTP cookies with outbound requests to feed sources, or restrict the portlet to forward cookies only to specific domain names. You can also enable active content filtering for all feeds to remove active content such as Java script from the feed text.

Procedure

1. Log in to WebSphere Portal Express as an administrator.
2. Click the **Administration menu** icon. Then, click **Portlet Management** > **Portlets**.
3. Locate the Syndicated Feed portlet.
4. Click the **Configure portlet** icon in the **Syndicated Feed Portlet** row.
5. Add the following preference and value pairs as appropriate to configure the portlet:

Table 395. Preference and value pairs for the Syndicated Feed portlet

Preference	Value
`cookiesToForward`	Specify one or more HTTP cookie names. Use a space to separate multiple cookie names. This preference forwards the specified cookies with the outbound portlet request when you connect to the feed source. **Note:** You can specify cookies to forward for a given feed when you add feeds to the portlet. The cookies that you specify as the value for `cookiesToForward` are appended to the list of cookies you specified when you added the feed.

Table 395. Preference and value pairs for the Syndicated Feed portlet (continued)

Preference	Value
`limitCookiesToForward`	Specify one or more domain names to which you want to forward cookies. This preference restricts cookie forwarding to only the domain names that you specify. You can specify a fully qualified domain name or a partially qualified domain name. For example, • Fully qualified domain name: `www.ibm.com` • Partially qualified domain name: `ibm.com`
`useACF`	Specify one of the following values: **`true`** Enables active content filtering for all feeds. This value removes any active content from the feed text; for example, embedded Java scripts. **`false`** Disables active content filtering for all feeds.

6. Click **OK** to save your changes.

Client side aggregation (CSA) rendering in IBM Syndicated Feed Portlet

IBM Syndicated Feed Portlet can be configured to operate in the client side aggregation (CSA) rendering mode. Client side rendering ensures an improved user experience through faster response time. When you enable client side rendering in the portlet, the portal does not re-render the whole page but only the aspects of the portlet that change.

The following feed management and feed presentation capabilities are available when client side rendering has been enabled for IBM Syndicated Feed Portlet:

Feed management capabilities

- Subscribing to feeds by specifying the feed url, feed title, and optionally feed credentials
- Editing a feed to change feed title
- Enabling or displaying the feeds in the view mode of the portlet
- Deleting a feed subscription

Feed presentation capabilities

- Customizing the number of articles to be displayed per feed
- Viewing the feed article content either expanded or collapsed
- Displaying or hiding the author of feed items
- Displaying or hiding the Published date of feed items

"Enabling client side rendering in IBM Syndicated Feed Portlet" on page 2478
To enable client side rendering in the IBM Syndicated Feed Portlet, create a `useClientSideRendering` configuration parameter and set its value to `true`.

"Configuring the profiling parameter on the page with the Syndicated Feed Portlet"
You must configure the full profiling parameter on the page on which you deployed the Syndicated Feed Portlet.

"Configuring the security role mapping for the Syndicated Feed Portlet" on page 2479
You must configure the security role mapping for the Syndicated Feed Portlet.

Enabling client side rendering in IBM Syndicated Feed Portlet

To enable client side rendering in the IBM Syndicated Feed Portlet, create a `useClientSideRendering` configuration parameter and set its value to `true`.

About this task

To enable client side rendering for IBM Syndicated Feed Portlet:

Procedure

1. From the WebSphere Integrated Solutions Console, click **Portlets** under **Portlet Management**.
2. Search for the Syndicated Feed portlet entry.
3. Click the **Configure portlet** (wrench) icon corresponding to the Syndicated Feed Portlet.
4. Add the `useClientSideRendering` configuration parameter to the existing list of configuration parameters for the Syndicated Feed Portlet. Set the value of this configuration parameter to `true`.

   Following are the steps to add a configuration parameter for a portlet:

   a. Enter the parameter name in the **New Preference** text field.
   b. Enter the value of the parameter in the **New value** text field.
   c. Click **Add**.

   Note: You do not need to restart the portal server after you specify the configuration parameters.

Configuring the profiling parameter on the page with the Syndicated Feed Portlet

You must configure the full profiling parameter on the page on which you deployed the Syndicated Feed Portlet.

Procedure

1. Create the page on which you want to deploy the Syndicated Feed Portlet. Make sure that the page inherits the portal default theme.
2. Deploy the Syndicated Feed Portlet on that page.
3. Locate the page in the **Manage pages** portlet.
4. Click **Edit Page Properties** for the page.
5. Click **Advanced options**.
6. Click **I want to set parameters**.
7. Add the new parameter and value as follows:
   - New parameter: `resourceaggregation.profile`
   - New value: Enter the profile override for the page, for example `profiles/profile_full.json`.
8. Click **Add**.

9. Click **OK** to save the new parameter.
10. Click **OK** to save your changes to the page properties.

Configuring the security role mapping for the Syndicated Feed Portlet

You must configure the security role mapping for the Syndicated Feed Portlet.

Procedure

1. Open the WebSphere Integrated Solutions Console.
2. Click **Application** > **Application Types**.
3. Click **WebSphere enterprise applications**.
4. Click the **Show filter function** icon.
5. In the **Search terms** field, type `PA_wp.feedspace`.
6. Click **Go**.
7. From the search result list, click **PA_wp.feedspace**.
8. Under Detail properties, click **Security role to user/group mapping** .
9. Check the check box for **All Role**.
10. From the drop-down list, select **All Authenticated in Application's Realm**.
11. Check and verify that the All Role has `All Authenticated in Application's Realm` listed for Special subjects.
12. Click **OK**.
13. Click **Apply** and **OK** to save your changes.

Chapter 15. Staging to production

During portal solution development, the solution is initially developed, tested, and refined on one server or a limited number of servers. The solution is deployed later on live systems, referred to as the production environment. The process of moving the solution from the development environment to the production environment is called staging.

Staging is required at multiple times during the development cycle of a solution. Each release of the solution is created on one system and then moved to the production system. During the development lifecycle, multiple solution releases are created and brought into production. Multiple solution releases are built on the previous product release.

Staging is only possible between the same product release. In contrast, upgrading from one release to a newer release is called migration.

IBM WebSphere Portal Express and IBM Web Content Manager solutions can consist of many artifacts. These artifacts include portlets, themes and skins, portlet services, page layouts, wires, portlet configurations, portlet data, web content, and personalization rules. Staging helps you move those artifacts to the production environment in a controlled way.

security manager for authorization decisions requires that the same external security manager is used to manage authentication to the portal.

Overview of staging to production

Review the topics in the overview section to understand all the requirements to successfully stage your production environment.

"Staging to production process"
This overview provides information necessary to understand how to stage your portal to production from managing your configuration to team roles and portal solution release configurations.

"Tools for staging to production" on page 2485
You can use several tools to stage your server to a production environment.

"Staging a virtual portal overview" on page 2486
A virtual portal shares several resources with the main portal installation, especially all the code artifacts and the JVM. When you deploy a portal solution release to a virtual portal, these types of artifacts must not be deployed again.

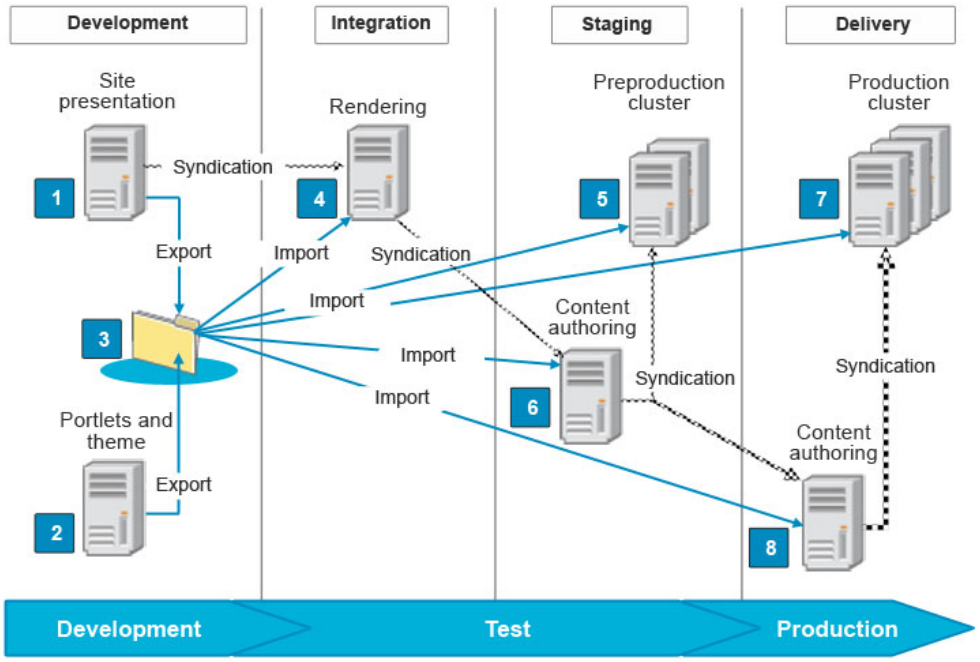
"Staging to production list" on page 2487
The items in the production list can be included in the staging to production Portal Application Archive (PAA) file. However, if you are not using the staging to production PAA file, use this list to determine the tools that are required to move your artifacts from the staging server to the production server. You can also use this list to determine what tool to use for the items not included in the staging to production PAA file.

Staging to production process

This overview provides information necessary to understand how to stage your portal to production from managing your configuration to team roles and portal solution release configurations.

Initial and differential release

The content of a portal solution release is different for the initial staging process and the staging processes that modify the existing content. The following high-level image illustrates the staging to delivery process:



The following information describes the diagram annotation:

1. The web content development environment is where the following information is developed:
   - Presentations
   - Authoring templates
   - Taxonomies
   - Content libraries
   - Content components such as personalization components
   - Personalization rules
2. The Portal development environment is where the following information is developed:
   - Portlets
   - Extensions
   - Themes
   - Applications
3. The export directory is where you store all source artifacts, including Web Content Manager artifacts. You can either compress this directory and import it to your other staging environments or build a Portal Application Archive (PAA) file.
4. Use the rendering environment to test the rendering of the web content and portal development artifacts together. Developers might have access to this environment to test their code.
5. Use the preproduction environment to test everything on a cluster configuration that mirrors the live cluster. Only the administrators have access to this environment.
6. You can use the Content Authoring environment to test the syndication process before you move to the production environment. This environment is the first place where you can test the full content of your site. The full content includes the page structure and the web content. The authoring (6) and rendering (5)

environments are also a good place to run more performance measurements. These measurements ensure that your production environment performs to your expectations. Some teams might also want to use the staging environment for training or demonstration purposes.

7. The production cluster environment is the live website cluster.
8. The authoring environment is where content developers and editors develop new content and content updates for the site.

The initial release

The initial solution release must move all artifacts from the source portal to the staging target, which is assumed to be empty. An empty portal does not hold any application payload. There are procedures in place to allow a clean portal installation and to prepare it as target for an initial staging step. Depending on the set of artifacts that is deployed on the source, multiple tools are required. Use these tools to export the artifacts from the source and import them to the target.

When the initial release is deployed to the production system, users access the system. They can create more user data such as customization and other data. Typically an initial release contains the following data:

- Custom code libraries to the application server that are needed by your applications and extensions
- WAR or EAR files to the application server that might contain custom extensions or themes and skins
- Portlet WAR files
- IBM Web Content Manager libraries
- The initial portal content in an XML file format that is created with the XMLAccess tool
- Personalization rules

The differential release (making updates)

Updates to the portal solution can include creating, modifying, or deleting various artifacts. Artifacts can be pages, portlets, rules, and web content. While the new content can be deployed as a full release, this option would have the following drawbacks:

- Removing of all release data and deploying the new full release is time-consuming. Instead, it would be beneficial to deploy the differences, especially since the differences are only a small part of the solution.
- Customizations can get lost.

To avoid the drawbacks, a differential release can be defined, holding only the changes and not the complete release. A differential release is created by comparing two release exports. Then, you figure out the changes that are required to convert one system into the other one. Release Builder is the tool that helps with this task.

WARNING: If you redeploy your site daily, your JCR size increases because of page versions. Periodically clean up your versions to reduce the JCR size. Go to "Clearing version history" on page 1194 for information.

Use the ReleaseBuilder to manage release configurations independent of user configurations. Release configuration data can be exported into an XMLAccess

configuration file. During staging of follow-on releases, it is possible to stage differences between two releases. Use the XML configuration interface for these differences. Difference means differences between release configurations, including configuration entities that were removed, added, or changed in comparison to the previous release. Go to "Updates using ReleaseBuilder" on page 3624 for information.

Web content updates are done with the Web Content Manager syndication feature. If managed pages are enabled, syndication also ensures that all required page artifacts are transferred along with the content.

Go to "Updates with syndication" on page 2503 for information about managing updates across multiple lines of production.

If your staging server has a different LDAP than the production, go to *Member fixer with syndication* and *Maintaining web content* for information.

Related concepts:

"Web content administration tools" on page 1176
IBM Web Content Manager includes tasks and tools to help maintain your content management system. For example, use the member fixer task to resolve renamed or deleted users and user groups. Use the workflow checker and updater tools to modify workflow security settings, reschedule pending actions, and add workflow to items. Web Content Manager also includes tools to assist with library and item management, and item and version history management.

Related reference:

"Member fixer with syndication" on page 1183
You can configure your system to automatically run the member fixer tool when syndicating. The member fixer is run on the subscriber during syndication. It is run against items that have just been syndicated. Details of the member fixer operations are included in the syndication report.

Tools for staging to production

You can use several tools to stage your server to a production environment.

Portal application archive (PAA)

Each type of artifact can be exported, moved to the target, and imported independently. However, a common archive format does the same tasks. Place all artifacts into the correct location inside the archive file. Copy the PAA file to the target and then use the Solution Installer to deploy it.

For an initial release, the archive file contains all the artifacts that are defined previously. Additional custom components require special handling. The PAA file can be augmented by custom actions that modify the behavior of the Solution Installer as required.

Syndication

Syndication allows your business users to define the content of your Portal Site through an Authoring system. They can then synchronize the content structure, navigation, and web content documents between the authoring system and the production system. Thus syndication is another way to stage changes from the authoring system to the production system. It defines the source and target of the syndication. The artifacts are moved in the direction from source to target only.

Because the business users do the management, the artifacts that can be syndicated are limited to the less critical ones not having the potential to crash the server. Contrary to the PAA process where the release data can be placed on a media that can be moved between the systems, syndication uses an online approach to move changes from one system to another. Thus syndication requires a connection between the systems.

Syndication can be set up between main or virtual portals on different systems or on the same portal system. Because the syndicated pages refer to portlets with the ID of the portlets, an initial staging needs to take place before syndication can be used.

Staging a virtual portal overview

A virtual portal shares several resources with the main portal installation, especially all the code artifacts and the JVM. When you deploy a portal solution release to a virtual portal, these types of artifacts must not be deployed again.

Redeploying can break the references in other virtual portals. Shared resources must be deployed only to the base portal. It is important that the shared resources are deployed to the base portal before they are referenced from the Virtual Portal. Because of these requirements, staging virtual portals needs to be done in the following sequence:

Note: Follow this sequence if the base portal assembly and the Virtual Portal are the same on the source and target server. For example, the staging server has the base portal, VP1, and VP2. Then, you would stage this information to a production server that also contains the base portal, VP1, and VP2.

1. Stage the base portal. This step includes the shared and unique resources for the base portal. You can split the Portal Application Archive (PAA) file for the base portal into two components. One is for the unique components and the other for the shared components. When you build the PAA, use the **exportUniquePortalPAA=false** to export shared artifacts. Or, use **exportSharedPortalPAA=false** to export the unique artifacts. For more information, see *Parameters to customize the release*.
2. Stage the unique artifacts for the Virtual Portal. This step is repeated for all Virtual Portals.

If the source and the target server are not structured the same, then modify the sequence. For example, if the original production server is not capable of carrying the entire base portal load, VP1, and VP2, then you create a new production server that contains only VP2.

1. Stage the shared components of the base portal only.
2. Stage the unique content (for example VP2) of the required Virtual Portal into the base portal of the new server.

This sequence results in a server with only VP2 content.

Note: Before you run the **build-initial-release-paa**, and you have created realms to define the user populations, you must create a super realm. In addition to the realms that you create to define the user populations of the individual virtual portals, you must create a super realm. This super realm spans all other realms and contains all the users of those other realms. It is also known as the default realm. To log in to a virtual portal, the virtual portal administrator and all users must be a member of the realm for that virtual portal. To allow a user access

to more than one virtual portal, that user, and the Virtual Member Manager node to which the user belongs in the hierarchy of the user directory, must be a member of all the realms associated with these virtual portals. This applies to a super administrator who is responsible for all virtual portals within an entire Portal installation. To administer the virtual portals, the master administrator must be a member of the realms of these virtual portals.

Related concepts:

"Parameters to customize the release" on page 2502
You can use these parameters to customize the **build-initial-release-paa** task when you are creating the initial or differential release.

Staging to production list

The items in the production list can be included in the staging to production Portal Application Archive (PAA) file. However, if you are not using the staging to production PAA file, use this list to determine the tools that are required to move your artifacts from the staging server to the production server. You can also use this list to determine what tool to use for the items not included in the staging to production PAA file.

Within the Staging to production documentation, the following directories are the root directories where you place your subdirectories:

- For the base portal: `WebSpherePortal`
- For the Virtual Portals: *`object_id`*

Themes and skins

Choose one of the following static resource options:

Static resources in the WebDAV file store
: Follow the staging to production documentation to export and import your themes and skins. For more information, see "Developing themes for a production portal" on page 2813. To include these resources in your initial or differential release PAA, use the **-DexportWebDavTheme=true** parameter.

  Directory structure
  : Use this method if you are building a custom PAA. Compress the files that contain static theme content and place them in the `content/webdav` directory. This placement allows the themes to be automatically uploaded to the `dav:fs-type1/themes/` directory in the WebDAV file store.

Static resources in an `.ear` file
: Follow the staging to production documentation to export and import your themes and skins. Read "Developing themes for a production portal" on page 2813. To include these resources in your initial or differential release PAA, use the **-DappsToExtract** parameter.

  Directory structure
  : Use this method if you are building a custom PAA. Place the theme `.ear` files in the `InstallableApps/ear` directory.

Portal Server artifacts

The following information is a list of portal server artifacts:

- action
- client

- component
- content-node (pages)
- credential-segment
- device-class
- global-settings
- language
- markup
- resource-type
- skin (definition)
- task
- theme (definition)
- services-settings
- servlet (definition)
- portal
- portlet-app
- portlet
- portletinstance
- proxy-config
- transformation
- transformation-app
- transformationinstance
- url-mapping-context
- virtual-resource
- web-app
- wsrp-producer
- cross-page-wire
- wire

These artifacts are included by default in your initial or differential release PAA. If you are building a custom PAA, use an `XMLAccess.xml` file for installation or uninstallation.

XMLAccess `.xml` files (for installation)
Place these files in the following directory: `content/xmlaccess/install`.

Note: These files might contain the statements to install portlets and transformations out of the `war` directory and to create pages.

XMLAccess `.xml` files (for uninstallation)
Place these files in the following directory: `content/xmlaccess/uninstall`.

Note: These files might contain the statements to remove portlets, transformations, and pages.

References
For information about the portal XML configuration interface and how to work with it, read *Working with the XML configuration interface* and *XML configuration reference*.

WebDAV files

Directory

Use this directory if you are building a custom PAA file. Put these files in the `webdav` directory. This directory and its subtree structure contain artifacts that must be uploaded to the WebDAV file store. There are the following possible subdirectories:

- `themes`
- `skins`
- `layout-templates`
- `common-resources`

References

See "Exporting content from the filestore" on page 2820.

Personalization rules and campaigns

These artifacts are included by default in your initial or differential release PAA. Use these tasks if you are building a custom PAA.

Task Use the **`pznload`** task to export and import the rules and campaigns.

Directory

Place the files in the `content/pzn` directory. Personalization-related artifacts, such as JAR files that contain business rules and personalization `.nodes` files, are in this directory.

References

See "Staging Personalization rules to production" on page 3629.

Web Content Manager data

To include Web Content Manager data in your initial release PAA, use the **`-DexportWcmData=true`** parameter. If you are using a custom PAA, then use the following tasks.

Attention: Run the export task if the data set is less than 10,000 content items and behind a firewall. Syndicate the content if the data set is less than 10,000 content items and not behind a firewall. If the data set is large, clone your database and run the **`sycn-vanityurl-data`** task to synchronize your vanity URL data. Either clone your database or run the export task. Do not do both.

Export task

Use the **`export-wcm-data`** task to export Web Content Manager data.

Import task

Use the **`import-wcm-data`** task to import Web Content Manager data.

Directory

Place the files in the `content/wcm` directory. Each subdirectory represents a separate web content library. These libraries are a specialized form of JCR artifact. Web content libraries are separated into their own directory.

References

Read "Exporting and importing web content libraries" on page 1200.

Portlet WAR files

These artifacts are included by default in your initial or differential release PAA.

If you are building a custom PAA, choose one or more of the following portlet options:

JSR 168 and JSR 286 Portlets (without an `XMLAccess` script to deploy)
Place these portlets in the following directory: `InstallableApps/portlets`.

JSR 168 and JSR 286 Portlets (with an `XMLAccess` script to deploy)
Place these portlets in the following directory: `InstallableApps/war`.

IBM API Portlets
Place these portlets in the following directory: `InstallableApps/war`.

Note: Existing IBM portlets are not handled automatically. These portlets are copied to the archive and must be installed by an accompanied **`XMLAccess`** script.

Transformation `.war` files
Place these portlets in the following directory: `InstallableApps/war`.

Note: These `.war` files are not handled automatically. These files are copied to the archive and must be installed by an accompanied **`XMLAccess`** script.

Shared libraries

To create shared libraries in production, use the **`sharedAppResourcesRootDir`** and **`sharedExtResourcesRootDir`** parameters when you create the initial or differential release PAA. If you are deploying shared libraries manually, be sure that they are included on the production server and that their contents are up to date. Or, if you are manually creating a PAA, put the JAR files into either the `shared/app` or `shared/ext` directories of your custom PAA.

EAR files

To include EAR files in your initial or differential release PAA, use the **`-DappsToExtract`** parameter.

If you are using a custom PAA, place files in the `InstallableApps/ear` directory. This directory contains any EAR files that must be deployed to the application server. It also contains WAR files that do not contain any portlets and must be installed directly to the application server.

More components to stage

The following list contains all portal components that are not handled by the initial import. Use the information in the following list to install the component on each system that is involved in the staging to production process.

Portal Server Performance Tuning parameters
Follow the performance tuning guide: Performance

Active Site Analytics Statistics
There are no applicable tools for exporting or importing Active Site Analytics Statistics. Statistics from staging server are not typically exported or imported. Read “Analyzing portal usage data” on page 1687.

WebSphere Application Server Resource Environment Providers (Portal service settings are stored as Resource Environment Providers)
Run the **`wsadmin`** task to export. Run the following **`ConfigEngine`** task to import: **`update-properties`**.

Read the following documentation:

- http://www.ibm.com/developerworks/websphere/techjournal/0904_chang/0904_chang.html
- "Setting service configuration properties" on page 283

External Security Manager

Follow the staging to production documentation. Read "Staging and external security managers" on page 2506.

Custom components

The following items are custom components:

- Custom login commands
- Custom credential vault adapters
- Custom credentials
- Custom component property files

Custom components are not part of WebSphere Portal Express. Custom components that are developed by the customer must be manually re-created on each system that is involved in the staging to production process.

Vanity URLs

These artifacts are included by default in your initial or differential release PAA. A vanity URL is a portal artifact that is stored with the portal page that it targets. Vanity URLs are managed and stored in the Web Content Manager Portal Site library. Therefore, if you want to stage your portal to production, it is not enough to stage the portal pages by using the portal XML configuration interface (XMLAccess). You must transfer data on both the portal and the Web Content Manager side.

Related concepts:

Staging-server topology for Web Content Manager
If you need to deliver large complex websites with many site users and content creator, implement a staging server configuration. Implementing a staging server provides an environment for checking for accuracy, issues with the design, and performance. With a staging server configuration authoring, staging and delivery are separated onto different servers. The staging environment can be used for testing or to allow changes from the authoring environment to accumulate before you syndicate the changes to the delivery environment in a single batch.

"Web content testing environments" on page 114
Testing environments can be simple or complex. A simple example is a stand-alone server where content and non-content items are tested before they are sent to the live site. A complex environment is a complete replica of your delivery environment. A more complex environment is used to test content, theme, and application changes and the performance of your delivery environment.

"Publishing personalization rules overview" on page 2298
WebSphere Portal Express Personalization sends published rules across HTTP to a servlet which resides on each personalization server. This servlet can receive publishing data or initiate new publishing jobs. When a user begins a publishing job from the personalization authoring environment, the local servlet is provided with the set of information necessary to complete the job. The local servlet contacts the destination endpoint servlet (which could be the same servlet) and sends its data to it. The destination servlet reports success or failure.

Related tasks:

"Exporting and importing a web content library" on page 1203
You can export the contents of a web content library to disk and import this data into another web content server. Use this feature to make a backup copy of a web content library, and to move data between servers. This function cannot be used to send updates, deletes, and moves. It is only suitable for populating new items.

Deleting a web content library
When a web content library is no longer required, you can delete the library.

"Using the XML configuration interface to work with virtual portals" on page 1414
You can export and import the contents of individual virtual portals by using the XML configuration interface. For example, you can use the XML configuration interface to fill a newly created virtual portal with content. As each virtual portal has its own globally unique portal ID, you can determine all resources that are associated with that virtual portal clearly and individually.

Related reference:

"XML configuration reference" on page 1082
Learn more about XML input structure, XML tags for portal resources, attributes for special purposes such as locale data, object IDs, and more. Also find information about action attributes that determine the type of processing that the XML configuration reference applies to the portal resource. There are syntax restrictions that you need to consider as well as information about how to determine the object IDs for portal resources.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

 Configuring a web content staging environment
Configure the staging environment to emulate the web content delivery environment and allow for testing before deployment.

Creating and deploying the initial release

Build the initial Portal Application Archive (PAA) file, prepare the servers for the deployment, and then deploy the initial release to your production servers.

1. "Creating the initial release" on page 2493
   When you want to stage your initial release to another server, you must build the initial release Portal Application Archive (PAA) file from your source server.
2. "Preparing the servers for initial staging" on page 2495
   You must set up your production servers before you move content from the staging server to the production server. The steps remove any release content from these servers to prepare them for the following import of the release data. It is assumed that these servers are fresh installed servers. The servers must not contain any customization data. Complete all database and security configurations.
3. "Deploying the initial release" on page 2496
   After you create and prepare your initial staging and production servers, you must install and deploy the contents of the staging server to the production server. If you have a cluster, install and deploy on the primary and secondary nodes of the cluster. After you stage the initial release to production servers, you can install and deploy your differential release to the production servers.

4. "Deploying the initial release in a multiple cluster" on page 2498
   If you are using a multiple cluster in single cell topology, you must run the **empty-portal** task before you deploy the initial release and before you deploy the PAA file. Use the **-DemptyPortalDuringDeployPAA=false** parameter during the deployment.

Creating the initial release

When you want to stage your initial release to another server, you must build the initial release Portal Application Archive (PAA) file from your source server.

About this task

The initial and differential release PAA file contains all the pertinent items that are related to your release, including the following items:

- Portal Server artifacts
- Web Content Manager Libraries (optional: use the **exportWcmData** parameter)
- Shared Libraries (optional: use the **sharedAppResourcesRootDir** and or the **sharedExtResourcesRootDir** parameters)
- WebDAV files for themes/skins (optional: use the **exportWebDavTheme** parameter)
- Portlet WAR filesEAR files (optional: use the **appsToExtract** parameter)
- WebSphere Application Server Resource Environment Providers (optional: use the **exportREP** parameter)
- Vanity URLs

Procedure

1. Install, configure, and upgrade WebSphere Portal Express on the staging server. This step includes database and security configurations.
2. Develop a release on the staging server. As part of this step, you might have pages, portlets, themes, skins, personalization rules, web content, and other items.
3. Open a command line and change to the *wp_profile_root*/ConfigEngine directory.
4. Run the following command to create a Portal Application Archive (PAA) file for your initial release, including PAA files for all of your virtual portals:
   - Linux :

     ```
     ./ConfigEngine.sh build-initial-release-paa -DdestPAADir=directory_to_store_PAA -DWasPasswor
     ```

   - IBM i:

     ```
     ConfigEngine.sh build-initial-release-paa -DdestPAADir=directory_to_store_PAA -DWasPassword=
     ```

   - Windows:

     ```
     ConfigEngine.bat build-initial-release-paa -DdestPAADir=directory_to_store_PAA -DWasPassword
     ```

   Note: The directory that you enter for the **destPAADir** parameter must be an existing directory.

   You can add the following parameters to customize the **build-initial-release-paa** task:

   sharedAppResourcesRootDir
   The directory for .jar files, classes, and compressed files that are then stored in the shared/app directory.

   sharedExtResourcesRootDir
   The directory for .jar files, classes, and compressed files that are then stored in the shared/ext directory.

appsToExtract
A comma-separated list of applications to extract from the source environment. For example, you might extract the `YourTheme.ear,MyApp.ear,CustWorkflow.ear` files. They are then packaged in the PAA file.

exportWebDavTheme
Enter a value of `true` to download and include the `themes.zip` file from WebDAV.

exportWcmData
Enter a value of `true` to export all Web Content Manager libraries. The **export-wcm-data** task is run when you use **exportWcmData**.

Restriction: Do not use this parameter in place of syndicating the libraries from the target environment.

Related concepts:

"Staging Personalization rules to production" on page 3629
Use the steps in this file to move Personalization rules from a staging system to a production system. There are a number of methods for moving rules between servers, each suitable for different situations.

Related tasks:

"Synchronizing the vanity URL database" on page 2104
Vanity URLs are stored as part of the page in the JCR database in the portal page site area of Web Content Manager. For performance reasons, the data is also stored in the WebSphere Portal Express database. When the data is modified, the portal synchronizes the data between both sides. However, under certain circumstances it can happen that the data is not synchronized. For such cases, the portal provides a configuration task that synchronizes the data.

"Using the XML configuration interface to work with virtual portals" on page 1414
You can export and import the contents of individual virtual portals by using the XML configuration interface. For example, you can use the XML configuration interface to fill a newly created virtual portal with content. As each virtual portal has its own globally unique portal ID, you can determine all resources that are associated with that virtual portal clearly and individually.

"Exporting and importing a web content library" on page 1203
You can export the contents of a web content library to disk and import this data into another web content server. Use this feature to make a backup copy of a web content library, and to move data between servers. This function cannot be used to send updates, deletes, and moves. It is only suitable for populating new items.

"Transferring portal configuration data by using the XML configuration interface" on page 1072
When you use the XML configuration interface to transfer WebSphere Portal Express configuration data, you export or import an XML script file. In most cases, you can use the result file from an XML export for an XML import. Sometimes you can use the export result file directly, sometimes you must modify it.

Related reference:

"XML configuration reference" on page 1082
Learn more about XML input structure, XML tags for portal resources, attributes for special purposes such as locale data, object IDs, and more. Also find information about action attributes that determine the type of processing that the XML configuration reference applies to the portal resource. There are syntax restrictions that you need to consider as well as information about how to determine the object IDs for portal resources.

Related information:

"Working with the XML configuration interface" on page 1061
You can use the XML configuration interface to export and import IBM WebSphere Portal Express configurations. This way you can configure selected features or areas of your portal. You can also transfer a complete configuration from one portal to another.

Preparing the servers for initial staging

You must set up your production servers before you move content from the staging server to the production server. The steps remove any release content from these servers to prepare them for the following import of the release data. It is assumed that these servers are fresh installed servers. The servers must not contain any customization data. Complete all database and security configurations.

Procedure

1. Install and configure WebSphere Portal Express on the production server. This step includes database and security configurations.

   Attention: Ensure that your security configurations and context root URLs match on the staging and production servers. Also, ensure that the production server is configured with the wanted external database.
2. Log on to the production server and delete all existing Web Content Manager Libraries.

   Click the **Administration menu** icon. Then, click **Portal Content** > **Web Content Libraries**. Click **Delete library** for each library on the system.

   Attention: Edit any libraries that have the option **Prohibit library from being deleted**. Clear that option before you delete them.
3. If you use the web application bridge feature, set the context root for the **wp.vwat.servlet.ear** application:
   a. Log on to the WebSphere Integrated Solutions Console.
   b. Go to **Applications** > **Application Types** > **WebSphere enterprise applications**.
   c. Find and click the **wp.vwat.servlet.ear** application link.
   d. Under the **Web Module Properties** heading, click **Context Root For Web Modules**.
   e. Change the context root to /. This step can create name conflicts. Add a rewrite rule to avoid these conflicts. For more information read *Apache Module mod_rewrite* and *Providing short vanity URLs*.
   f. Click **OK**.
   g. Click **Save** to save your changes to the master configuration.
   h. Stop and restart the **wp.vwat.servlet.ear** application.
4. If you have a clustered environment, turn off automatic node synchronization on all secondary nodes.

Related information:

Exporting and importing web content libraries

IBM Web Content Manager provides two methods for exporting and importing web content libraries: an export or import that operates on one library, and an export or import that creates a separate copy of a library. With either method, you can export the contents of a web content library to disk and import this data into another web content server. If you're working with a copy of a library, you can also import that library into the same web content server multiple times, resulting in a new library after each import without affecting previous copies. Exporting and importing libraries can be used to make a backup copy of a web content library and can also be used to move data between servers. However, this function cannot be used to send updates, deletes, and moves. It is only suitable for populating new items.

Deploying the initial release

After you create and prepare your initial staging and production servers, you must install and deploy the contents of the staging server to the production server. If you have a cluster, install and deploy on the primary and secondary nodes of the cluster. After you stage the initial release to production servers, you can install and deploy your differential release to the production servers.

Procedure

1. Run the following command from the *wp_profile_root*/ConfigEngine directory of the production server to install and deploy the initial PAA file on the production server:
   - Linux :

     ```
     ./ConfigEngine.sh install-paa -DPAALocation=/WebSpherePortal.paa -DWasPassword=password -DPortalAdminPwd=password -Dwp.si.offlineMode=true
     ./ConfigEngine.sh deploy-paa -DappName=WebSpherePortal -DforceDeploy=true -DWasPassword=password -DPortalAdminPwd=password
     ```

   - IBM i:

     ```
     ConfigEngine.sh install-paa -DPAALocation=/WebSpherePortal.paa -DWasPassword=password -DPortalAdminPwd=password -Dwp.si.offlineMode=true
     ConfigEngine.sh deploy-paa -DappName=WebSpherePortal -DforceDeploy=true -DWasPassword=password -DPortalAdminPwd=password
     ```

   - Windows:

     ```
     ConfigEngine.bat install-paa -DPAALocation=/WebSpherePortal.paa -DWasPassword=password -DPortalAdminPwd=password -Dwp.si.offlineMode=true
     ConfigEngine.bat deploy-paa -DappName=WebSpherePortal -DforceDeploy=true -DWasPassword=password -DPortalAdminPwd=password
     ```

2. Run the following commands to install and deploy the virtual portal PAA file on the production server:
   - Linux :

     ```
     ./ConfigEngine.sh install-paa -DPAALocation=/object_id.paa -DWasPassword=password -DPortalAdminPwd=password -Dwp.si.offlineMode=true
     ./ConfigEngine.sh deploy-paa -DappName=object_id -DforceDeploy=true -DWasPassword=password -DPortalAdminPwd=password -DVirtualPortalParame
     ```

   - IBM i:

     ```
     ConfigEngine.sh install-paa -DPAALocation=/object_id.paa -DWasPassword=password -DPortalAdminPwd=password -Dwp.si.offlineMode=true
     ConfigEngine.sh deploy-paa -DappName=object_id -DforceDeploy=true -DWasPassword=password -DPortalAdminPwd=password -DVirtualPortalParamete
     ```

   - Windows:

     ```
     ConfigEngine.bat install-paa -DPAALocation=/object_id.paa -DWasPassword=password -DPortalAdminPwd=password -Dwp.si.offlineMode=true
     ConfigEngine.bat deploy-paa -DappName=object_id -DforceDeploy=true -DWasPassword=password -DPortalAdminPwd=password -DVirtualPortalParamet
     ```

   Tip: Where **VirtualPortalParameter** is one of the following options:

   -DVirtualPortalHostName=*myvirtualportal.mycompany.com*
   Use this parameter if you know the host name of the virtual portal.

   -DVirtualPortalContext=*VirtualPortal1*
   Use this parameter if you know the context root of the virtual portal.

If you are deploying multiple Virtual Portals, run **list-all-virtual-portals** on the source environment, to correctly associate the **VirtualPortalParameter** value with the same Virtual Portal. For more information, see *Portal configuration tasks for administering virtual portals*.

Note: Anything included in the PAA file is included when you run this command, including Web Content Manager and EAR files. If you created the PAA by setting the `exportWcmData` parameter to `true`, then the Web Content Manager libraries are imported during the deployment step.

3. If you cloned your JCR database, extra steps are necessary. You must attach the clone JCR after you do the deployment for the base and virtual portals. The JCR database has objectIDs for the virtual portal that must match the source environment. The **deploy-paa** for the virtual portal does that task, but the **deploy-paa** for the virtual portals must be done before you attach to the cloned database. Run the following task on the target server to synchronize your vanity URL data. The release database and JCR databases share some vanity URLs. If you clone the JCR database, the sharing is broken. This task fixes those errors.
   - Linux :

     ```
     ./ConfigEngine.sh sync-vanityurl-data -DWasPassword=password -DPortalAdminPwd=password
     ```
   - IBM i:

     ```
     ConfigEngine.sh sync-vanityurl-data -DWasPassword=password -DPortalAdminPwd=password
     ```
   - Windows:

     ```
     ConfigEngine.bat sync-vanityurl-data -DWasPassword=password -DPortalAdminPwd=password
     ```
4. Review the checklist. Export or manually re-create any items that cannot be packaged with the PAA file such as Resource Environment Providers and custom content.
5. If you use WSRP, review the configuration of the WSRP web services.
   - If you use your portal as a WSRP producer, manually re-create the web service security configuration of the WSRP service providers. For more information, see the *WSRP services* section.
   - If you use your portal as a WSRP Consumer, verify the web service security configuration and manually re-create the web service security configuration if needed.
     a. If you configured WSRP Producer ports for web service security, for example, by using the Web Services admin portlet or XMLAccess, review the web service security configuration.
     b. If you configured WSRP web service clients, for example, by configuring WSRP service clients through policy sets in the administrative console or by adding service references, you must manually re-create the configuration of the WSRP web service clients. For more information, see the *WSRP services* section.
     c. If you configured web service security by using an LTPA version 1 token, you can enable those tokens. For more information, see the *WSRP services* section.
     d. If you configured HTTP-cookie-based single sign-on, for example, by configuring corresponding client cookie forwarding rules, you must review and re-create the resource environment properties for the cookie forwarding rules. For more information, see the *WSRP services* section.
6. Optional: Export search collections from the staging server and import them into the production server. For more information, see *Exporting and importing search collections*. If you are using alternative context root URLs, go to **Manage Search** portlet. To open the **Manage Search** portlet, click the **Administration**

menu icon. Then, click **Search Administration** > **Manage Search**. Update the search collection URL links on the staging server to point to the alternative root. For example, `http://yourdomain.com:port/your_root/seedlist/myserver?SeedlistId=&Source=com.ibm.workplace.wcm.plugins.seedlist.retriever.WCMRetrieverFactory&Action=Get`

Note: If you installed any additional PAA files on the staging server, they are not included in the initial release PAA. Each additional PAA that was installed and deployed on the staging server must be manually installed and deployed on the production server after the initial release PAA is installed.

7. If you have a clustered environment, enable automatic synchronization for the nodes and synchronize the nodes.
8. Restart the server.

Related concepts:

"WSRP services" on page 1433
IBM WebSphere Portal Express supports the Web Services for Remote Portlets (WSRP) standard. By using this standard, portals can provide portlets, applications, and content as WSRP services. Other portals can then integrate the WSRP services as remote portlets for their users.

Related tasks:

"Exporting and importing search collections" on page 707
View the steps to export search collections from a source portal and import them into a target portal.

Related reference:

"Portal configuration tasks for administering virtual portals" on page 1406
You can use configuration tasks for administering virtual portals.

Deploying the initial release in a multiple cluster

If you are using a multiple cluster in single cell topology, you must run the **empty-portal** task before you deploy the initial release and before you deploy the PAA file. Use the **-DemptyPortalDuringDeployPAA=false** parameter during the deployment.

Before you begin

If you have a single cell with a cluster A and a cluster B, use the following steps after you installed the PAA file on both clusters.

Procedure

1. On cluster B, run the following commands.
   - Linux :

     ./ConfigEngine.sh empty-portal -DWasPassword=*password* -DPortalAdminPwd=*password*
   - IBM i:

     ConfigEngine.sh empty-portal -DWasPassword=*password* -DPortalAdminPwd=*password*
   - Windows:

     ConfigEngine.bat empty-portal -DWasPassword=*password* -DPortalAdminPwd=*password*
2. On cluster A, do a full resynchronization with the WebSphere Integrated Solutions Console.
3. On cluster A, run the following commands:
   - Linux :

     ./ConfigEngine.sh empty-portal -DWasPassword=*password* -DPortalAdminPwd=*password*
   - IBM i:

ConfigEngine.sh empty-portal -DWasPassword=*password* -DPortalAdminPwd=*password*

- Windows:

ConfigEngine.bat empty-portal -DWasPassword=*password* -DPortalAdminPwd=*password*

4. On cluster B, do a full resynchronization with the WebSphere Integrated Solutions Console.
5. On cluster A, run the following commands:
   - Linux :

     ./ConfigEngine.sh deploy-paa -DappName=WebSpherePortal -DforceDeploy=true -DWasPassword=*password* -DPortalAdminPwd=*password* -DemptyPort

   - IBM i:

     ConfigEngine.sh deploy-paa -DappName=WebSpherePortal -DforceDeploy=true -DWasPassword=*password* -DPortalAdminPwd=*password* -DemptyPortal

   - Windows:

     ConfigEngine.bat deploy-paa -DappName=WebSpherePortal -DforceDeploy=true -DWasPassword=*password* -DPortalAdminPwd=*password* -DemptyPorta

6. On cluster B, run the following commands:
   - Linux :

     ./ConfigEngine.sh deploy-paa -DappName=WebSpherePortal -DforceDeploy=true -DWasPassword=*password* -DPortalAdminPwd=*password* -DemptyPort

   - IBM i:

     ConfigEngine.sh deploy-paa -DappName=WebSpherePortal -DforceDeploy=true -DWasPassword=*password* -DPortalAdminPwd=*password* -DemptyPortal

   - Windows:

     ./ConfigEngine.bat deploy-paa -DappName=WebSpherePortal -DforceDeploy=true -DWasPassword=*password* -DPortalAdminPwd=*password* -DemptyPor

Creating and deploying a differential release

Create a differential release of your existing Portal Application Archive (PAA) file. Then, deploy your changes to your production servers. Use Syndication to move IBM Web Content Manager content to your production servers.

1. "Creating the differential release"
   To update your Portal Application Archive (PAA) file, you must create a differential release from your source server.
2. "Deploying the differential release" on page 2500
   After you create the differential release PAA file, you can install and deploy your differential release to the production servers. If you have a cluster, install and deploy on the primary and secondary nodes of the cluster.

Creating the differential release

To update your Portal Application Archive (PAA) file, you must create a differential release from your source server.

Procedure

1. Develop a release on the staging server. As part of this step, you might have non-managed pages, portlets, themes, skins, and other items.

   Tip: Use Syndication to move IBM Web Content Manager content to your production servers.
2. Open a command line and change to the *wp_profile_root*/ConfigEngine directory.
3. Run the following command to create a Portal Application Archive (PAA) file for your differential release, including PAA files for all of your virtual portals. Go to *Parameters to customize the release* for parameters to customize your **build-initial-release-paa** task.
   - Linux :

     ./ConfigEngine.sh build-initial-release-paa -DdestPAADir=*directory_to_store_PAA* -DpreviousPA

   - IBM i:

     ConfigEngine.sh build-initial-release-paa -DdestPAADir=*directory_to_store_PAA* -DpreviousPAA=

- Windows:

```
ConfigEngine.bat build-initial-release-paa -DdestPAADir=directory_to_store_PAA -DpreviousPAA=i
```

Important: The directory that you enter for the **destPAADir** parameter must be an existing directory.

You can add the following parameters to customize the **build-initial-release-paa** task.

javaoption
Add this parameter if you are staging a large set of data. For example, add this string to your command, `-javaoption -Xmx2048M`. Depending on your system and the amount of data, the **-Xmx** value might be higher. Use `2048M` as a starting value.

sharedAppResourcesRootDir
The directory for `.jar` files, classes, and compressed files that are then stored in the `shared/app` directory.

sharedExtResourcesRootDir
The directory for `.jar` files, classes, and compressed files that are then stored in the `shared/ext` directory.

appsToExtract
A comma-separated list of applications to extract from the source environment. For example, you might extract the "`wps_theme,iehs,dojo_resources`" files. They would then be packaged in the PAA file.

exportWebDavTheme
Enter a value of `true` to download and include the `themes.zip` file from WebDAV.

What to do next

Deploy your differential release on the production servers.

Related tasks:

"Parameters to customize the release" on page 2502
You can use these parameters to customize the **build-initial-release-paa** task when you are creating the initial or differential release.

Deploying the differential release

After you create the differential release PAA file, you can install and deploy your differential release to the production servers. If you have a cluster, install and deploy on the primary and secondary nodes of the cluster.

Procedure

1. Run the following command from the *wp_profile_root*`/ConfigEngine` directory of the production server to uninstall and delete any existing differential PAA files:
   - Linux :

```
./ConfigEngine.sh uninstall-paa -DappName=WebSpherePortalUpdate -DWasPassword=password -DPortalAdminPwd=password

./ConfigEngine.sh delete-paa -DdeleteAll=true
```

   - IBM i:

```
ConfigEngine.sh uninstall-paa -DappName=WebSpherePortalUpdate -DWasPassword=password -DPortalAdminPwd=password

ConfigEngine.sh delete-paa -DdeleteAll=true
```

   - Windows:

```
ConfigEngine.bat uninstall-paa -DappName=WebSpherePortalUpdate -DWasPassword=password -DPortalAdminPwd=password

ConfigEngine.bat delete-paa -DdeleteAll=true
```

2. Run the following command to install and deploy the differential PAA file:
   - Linux :

     ```
     ./ConfigEngine.sh install-paa -DPAALocation=/WebSpherePortalUpdate.paa -DWasPassword=password -DPortalAdminPwd=password -Dwp.si.offlin

     ./ConfigEngine.sh deploy-paa -DappName=WebSpherePortalUpdate -DforceDeploy=true -DWasPassword=password -DPortalAdminPwd=password
     ```

   - IBM i:

     ```
     ConfigEngine.sh install-paa -DPAALocation=/WebSpherePortalUpdate.paa -DWasPassword=password -DPortalAdminPwd=password -Dwp.si.offlineM

     ConfigEngine.sh deploy-paa -DappName=WebSpherePortalUpdate -DforceDeploy=true -DWasPassword=password -DPortalAdminPwd=password
     ```

   - Windows:

     ```
     ConfigEngine.bat install-paa -DPAALocation=/WebSpherePortalUpdate.paa -DWasPassword=password -DPortalAdminPwd=password -Dwp.si.offline

     ConfigEngine.bat deploy-paa -DappName=WebSpherePortalUpdate -DforceDeploy=true -DWasPassword=password -DPortalAdminPwd=password
     ```

3. Run the following command from the *wp_profile_root*/ConfigEngine directory of the production server to uninstall virtual portal updated PAA files:
   - Linux :

     ```
     ./ConfigEngine.sh uninstall-paa -DappName=object_idUpdated.paa -DWasPassword=password -DPortalAdminPwd=password

     ./ConfigEngine.sh delete-paa -DdeleteAll=true
     ```

   - IBM i:

     ```
     ConfigEngine.sh uninstall-paa -DappName=object_idUpdated.paa -DWasPassword=password -DPortalAdminPwd=password

     ConfigEngine.sh delete-paa -DdeleteAll=true
     ```

   - Windows:

     ```
     ConfigEngine.bat uninstall-paa -DappName=object_idUpdated.paa -DWasPassword=password -DPortalAdminPwd=password

     ConfigEngine.bat delete-paa -DdeleteAll=true
     ```

4. Run the following commands to install and deploy the virtual portal PAA file:
   - Linux :

     ```
     ./ConfigEngine.sh install-paa -DPAALocation=/object_idUpdated.paa -DWasPassword=password -DPortalAdminPwd=password -Dwp.si.offlineMode

     ./ConfigEngine.sh deploy-paa -DappName=object_idUpdated -DforceDeploy=true -DWasPassword=password -DPortalAdminPwd=password -DVirtualP
     ```

   - IBM i:

     ```
     ConfigEngine.sh install-paa -DPAALocation=/object_idUpdated.paa -DWasPassword=password -DPortalAdminPwd=password -Dwp.si.offlineMode=t

     ConfigEngine.sh deploy-paa -DappName=object_idUpdated -DforceDeploy=true -DWasPassword=password -DPortalAdminPwd=password -DVirtualPor
     ```

   - Windows:

     ```
     ConfigEngine.bat install-paa -DPAALocation=/object_idUpdated.paa -DWasPassword=password -DPortalAdminPwd=password -Dwp.si.offlineMode=

     ConfigEngine.bat deploy-paa -DappName=object_idUpdated -DforceDeploy=true -DWasPassword=password -DPortalAdminPwd=password -DVirtualPo
     ```

   Tip: Where **VirtualPortalParameter** is one of the following options:

   -DVirtualPortalHostName=*myvirtualportal.mycompany.com*
   Use this parameter if you know the host name of the virtual portal.

   -DVirtualPortalContext=*VirtualPortal1*
   Use this parameter if you know the context root of the virtual portal.

   If you are deploying multiple Virtual Portals, run **list-all-virtual-portals** on the source environment, to correctly associate the **VirtualPortalParameter** value with the same Virtual Portal. For more information, see *Portal configuration tasks for administering virtual portals*.

   Note: Run syndication for Web Content Manager differential content.

5. If you have a clustered environment, enable automatic synchronization for the nodes and synchronize the nodes.
6. Optional: Complete this step if PAA files include shared libraries. If you included **-DsharedAppResourcesRootDir** and **-DsharedExtResourcesRootDir** when the differential and initial release was created, remove the shared library

that was created during deployment of the initial release. For example, if you included the `sample.jar` file in the initial release PAA, and updated the `sample.jar` file in the differential release PAA, there is a class path conflict with this JAR file that is included in two locations. Find and remove the following JAR files from initial deployment:

- *wp_profile_root*`/paa/WebSpherePortal/components/WebSpherePortal.shared/shared/ext/sample.jar`
- *wp_profile_root*`/paa/WebSpherePortal/components/WebSpherePortal.shared/shared/app/sample.jar`

7. Restart the server.

Related concepts:

"WSRP services" on page 1433
IBM WebSphere Portal Express supports the Web Services for Remote Portlets (WSRP) standard. By using this standard, portals can provide portlets, applications, and content as WSRP services. Other portals can then integrate the WSRP services as remote portlets for their users.

Related tasks:

"Exporting and importing search collections" on page 707
View the steps to export search collections from a source portal and import them into a target portal.

Related reference:

"Portal configuration tasks for administering virtual portals" on page 1406
You can use configuration tasks for administering virtual portals.

Parameters to customize the release

You can use these parameters to customize the **`build-initial-release-paa`** task when you are creating the initial or differential release.

`previousPAA`
Points to a previous PAA file, which is used to create an updated PAA file.

`sharedAppResourcesRootDir`
The directory for shared application resources that are added to the PAA file.

`sharedExtResourcesRootDir`
The directory for the `shared/ext` resources that are added to the PAA file.

`appsToExtract`
A comma-separated list of applications to extract from the IBM WebSphere Application Server and add to the PAA file.

`exportWebDavTheme`
When set to true, this parameter adds WebDAV theme resources to the PAA file.

`exportREP`
When set to true, this parameter adds Resource Environment Provider custom properties to the PAA file. Its default is true.

`versionPAA`
Set to the version number of the PAA file that you created. The default is 1.

`minorVersionPAA`
Set to the minor version number of the PAA file that you created. The default is 0.

destPAADir
Writes PAA files to the directory you specify. The default is the system temporary directory.

exportUniquePortalPAA
Set to true to export the unique portal resource to the base of the virtual portal. The default is true.

exportSharedPortalPAA
Set to true to export the shared resource from the base portal. The default is true.

exportPortalSiteLib
Set to true to export the Portal site library that is used by the managed pages feature. The default is true if managed pages is turned on.

exportManagedPages
Set to true to export managed pages with XMLAccess. The default is false if managed pages is turned on and true if managed pages is turned off.

exportPAAUrlMappings
Set to true to include URL mappings in the PAA file. The default is true.

exportWcmData
Set to true to include IBM Web Content Manager content in the PAA file. The default is false.

Important: Use the **exportWcmData** parameter only for the initial release. Use syndication for your Web Content Manager differential content.

javaoption
Add this parameter if you are staging a large set of data. For example, add this string to your command: **-javaoption -Xmx2048M**. Depending on your system and the amount of data, the **-Xmx** value might be higher. Use 2048 Mb as a starting value.

Updates with syndication

After setting up your initial staging and production servers and deploying the initial release, you can use the syndication feature of IBM Web Content Manager to update content in web content libraries. If managed pages are enabled, syndication also ensures that all required page artifacts are transferred along with the content.

"Syndication and staging"
You can use syndication to update content that was originally created by deploying portal solution releases with either the XML configuration interface or through a Portal Application Archive (PAA) file. You can also set up syndication between virtual portals or primary portals on the same system or between virtual portals on different systems.

"Staging artifacts that are not transferred by syndication" on page 2505
The `ExportManagedPagesRelease.xml` file provides an example that you can use with the XML configuration interface to export all portal artifacts except pages and wires. You can use this file when staging to production with managed pages.

Syndication and staging

You can use syndication to update content that was originally created by deploying portal solution releases with either the XML configuration interface or

through a Portal Application Archive (PAA) file. You can also set up syndication between virtual portals or primary portals on the same system or between virtual portals on different systems.

Important: Successful syndication requires that the object IDs for portlets, themes, and other artifacts are the same on both the syndicator and subscriber. Because the syndication process itself does not manage these artifacts, you must synchronize the two servers by an initial staging process before you syndicate. In addition, each time that you deploy an artifact that is not managed by syndication to the source portal, you must stage the artifact with the appropriate staging tool. However, if the source and target are different virtual portals on the same portal server, this step is unnecessary, because these artifacts are shared between virtual portals.

Limitations:

- Syndication can be set up only between systems that use the same user repository.
- Syndication for managed pages between multiple servers requires that you run an initial staging to all the servers.
- The syndication process runs a prerequisite check on the subscriber to ensure that any required themes, skins, portlets, or iWidgets on a page are present on the target system. If any required objects are missing on the subscriber, syndication is not run for the page. This behavior can result in missing pages and syndication errors for affected pages. Use the XML configuration interface to transfer the missing resources.
- iWidgets and portlets can store data in the WebDAV file store. The syndication process does not verify or transfer WebDAV data. If a portlet on a managed page requires data from the WebDAV file store, you must manually copy the required objects to the target system.

For information about using syndication, see *Syndication*.

Managed pages and syndication

If managed pages are enabled and you are using syndication as part of the staging process, the following considerations apply:

- In addition to web content, syndication includes artifacts like pages and wires.
- When you populate the production server for the first time, you must run a full export with either the XML configuration interface or through a Portal Application Archive (PAA) file. This full export can include pages and wires.
- When you update the production server after the initial staging, do not continue to export pages and wires, but instead use syndication to transfer the updates. If you export pages and wires after the initial staging, you might inadvertently overwrite changes that are already published through syndication.

The `ExportManagedPagesRelease.xml` file or a Portal Application Archive (PAA) file is available for exporting all artifacts except pages and wires. For details on using this file, see *Staging artifacts that are not transferred by syndication* or *Creating a differential release*.

Related concepts:

"Syndication" on page 448
Use syndication to replicate web content library data from one server to another server. Syndication is based on a syndicator and subscriber relationship. The syndicator has the current data. The subscriber received the current data from the syndicator.

"Syndication relationships" on page 449
Syndication is the method that is used by IBM Web Content Manager to replicate data from a web content library on one server to a web content library on another server.

"The XML configuration interface" on page 1054
Use the XML configuration interface (XML Access) for exchanging portal configurations.

Related tasks:

"Staging artifacts that are not transferred by syndication"
The ExportManagedPagesRelease.xml file provides an example that you can use with the XML configuration interface to export all portal artifacts except pages and wires. You can use this file when staging to production with managed pages.

"Creating the differential release" on page 2499
To update your Portal Application Archive (PAA) file, you must create a differential release from your source server.

Staging artifacts that are not transferred by syndication

The ExportManagedPagesRelease.xml file provides an example that you can use with the XML configuration interface to export all portal artifacts except pages and wires. You can use this file when staging to production with managed pages.

Before you begin

Stage artifacts that are not part of the syndication, such as portlet definitions, themes, and personalization rules with the release builder Portal staging tool. It ensures that the artifacts have the same object IDs across the staging systems because they are referenced by this object ID.

About this task

The file is in the *PortalServer_root*/doc/xml-samples directory.

Procedure

1. To export these artifacts with the ExportManagedPagesRelease.xml file, use the **xmlaccess** command.

   Linux
   ```
   ./xmlaccess.sh -in /opt/IBM/WebSphere/PortalServer/doc/xml-
   samples/ExportManagedPagesRelease.xml -out /Sample1/content/
   xmlaccess/install/staging-version1.xml -url "http://
   stagingserver.example.com:port/wps/config" -user wpsadmin_user_ID
   -password wpsadmin_pwd
   ```

   IBM i
   ```
   xmlaccess.sh -in /QIBM/ProdData/WebSphere/PortalServer/V8/
   product_offering/doc/xml-samples/ExportManagedPagesRelease.xml
   -out /Sample1/content/xmlaccess/install/staging-version1.xml -url
   "http://stagingserver.example.com:port/wps/config" -user
   wpsadmin_user_ID -password wpsadmin_pwd
   ```

   Windows
   ```
   xmlaccess.bat -in C:\IBM\WebSphere\PortalServer\doc\xml-samples\
   ExportManagedPagesRelease.xml -out \Sample1\content\xmlaccess\
   install\staging-version1.xml -url "http://
   stagingserver.example.com:port/wps/config" -user wpsadmin_user_ID
   -password wpsadmin_pwd
   ```

   The exported configuration is stored in the staging-version1.xml file.

2. You can use the exported file to import these artifacts. Use the /Sample1/content/xmlaccess/install/staging-version1.xml file and use the **xmlaccess** command.

   Linux `./xmlaccess.sh -in /Sample1/content/xmlaccess/install/staging-version1.xml -out /Sample1/content/xmlaccess/install/staging-version1-Output.xml -url "http://productionserver.example.com:port/wps/config" -user wpsadmin_user_ID -password wpsadmin_pwd`

   IBM i `xmlaccess.sh -in /Sample1/content/xmlaccess/install/staging-version1.xml -out /Sample1/content/xmlaccess/install/staging-version1-Output.xml -url "http://productionserver.example.com:port/wps/config" -user wpsadmin_user_ID -password wpsadmin_pwd`

   Windows

   ```
   xmlaccess.bat -in C:\Sample1\content\xmlaccess\install\staging-version1.xml -out C:\Sample1\content\xmlaccess\install\staging-version1-Output.xml -url "http://stagingserver.example.com:port/wps/config" -user wpsadmin_user_ID -password wpsadmin_pwd
   ```

Staging and external security managers

Staging to production where external security managers (ESM) are used is complex and needs special consideration. The following section discusses considerations of the impact of external security managers on the staging process. External security managers can be used to externalize authentication and authorization decisions from the portal. Externalization of authentication decisions to an external security manager has no impact on the staging to production functionality. Management of authorizations by an external security manager has an impact on the staging to production scenario. Using an external security manager for authorization decisions requires that the same external security manager is used to manage authentication to the portal.

Use the security manager to manage entities that the centralized security manager controls. Managing those entities elsewhere violates the principles of centralized security control. This principle does not change in staging-to-production support. For staging-to-production, this principle leads to the inability to modify security definitions of a resource that is externalized. The following sections describe the direct implications in external security manager (ESM) scenarios for staging-to-production.

The XML configuration interface is used for configuration management in the staging-to-production support. The XML configuration interface covers configurations that are managed by the portal. The XML configuration interface configuration files contain a list of all role mappings for all internally managed resources. Using external security managers allows management of entitlements for selected resources outside of the portal.

For IBM WebSphere Portal Express, the XML configuration interface covers security definitions of externalized resources. You can provide a request parameter within an export request through this functionality for the XML configuration interface. This parameter causes the XML configuration interface to include role mappings of all resources in the output file. The exported file contains role

mappings for internally managed resources and externally managed resources. This feature allows for the staging of role mapping of all resources that are not already externalized in the target system.

The following support matrix illustrates staging support of entitlements in scenarios that involve external security managers (ESM). The source system refers to the system from which the configuration is exported (typically the integration system). The target system is the system into which the configuration is going to be imported (for example, staging or production system).

	no-ESM Target	**ESM Target**
no-ESM Source	All role mappings can be fully synchronized.	Resources on the target system are initially managed internally after release staging. Resources can be externalized on the target system. Role mappings of externally managed resources in the target system are reinternalized.
ESM Source	Role mappings of internalized resources can be synchronized. Synchronization of role mappings of externalized resources results in errors during import. (Manual configuration file updates ("change external to internal state") are required to prevent errors.)	Role mappings of internalized resources are fully synchronized. Role mappings of externalized resources can be synchronized only on time of externalization. Role mappings for resources that are externalized on the target system cannot be synchronized.

In summary, role mappings can be synchronized when the resource is managed internally by the WebSphere Portal Express server. When the resource is externalized, role mappings are owned by the external security manager system and cannot be synchronized.

However, it is not possible to modify role mappings for existing externalized resources and synchronize their updated role mappings to a target system.

Staging from system without external security manager to a system with external security manager

This combination is used where the integration system is reduced as much as possible and is configured like a development environment rather than a production environment.

All entitlements for all resources that are staged in this scenario are fully synchronized. All resources in the target system are managed internally. The external security manager is not used for managing authorization decisions.

It is possible to manually externalize resources. You can also manually edit XML configuration interface configuration files. In this step, all role mappings that were defined within the portal before are moved to the external security manager. Managing these resources externally might include updating the role mappings of these resources outside the portal.

During staging of subsequent releases, resources with updated entitlements appear in the XML configuration interface file as internally managed resources. These resources are internalized when this file is imported to the target system. You can manually update the configurations for externally managed resources to prevent this action. However, there is no build-in support for managing these kinds of configurations. Special care must be taken to manually replicate all changes of role mappings of externalized resources on the target system.

Staging from system with external security manager to system without external security manager

In this scenario, entitlements for internalized resources can be staged without any limitations. These configurations appear as if the source system was not configured to use an external security manager.

If entitlements for resources that are externalized on the source system are staged, errors occur on the target system. During import of this configuration, the target system is asked to externalize a resource. The system is not configured to support resource externalization. This option results in an error condition.

These errors can be resolved by manually updating the configuration file to have all resources that are managed internally. If role mappings for externalized resources are contained in the configuration file, entitlements are synchronized between the two systems.

Chapter 16. Developing

Developing for IBM WebSphere Portal Express includes developing portlets, extending the applications that are provided with the portal, and more. WebSphere Portal Express includes IBM Web Experience Factory to help you start developing quickly. You can also use IBM WebSphere Studio Application Developer.

"REST service for Web Content Manager" on page 3306
Application developers can use Representational State Transfer (REST) services to work with Web Content Manager. The REST service for Web Content Manager provides authoring access to content items and elements. The service follows the Atom Publication Protocol, and Atom feeds, and entries are accessible in XML (application/atom+xml) and JSON (application/json) format.

"How to display data from external sources" on page 3399
You display data from external sources by using the same methods as you would when you create a website.

"Instrumenting web content for Active Site Analytics" on page 3399
You can collect information from web content for Active Site Analytics.

"Java messaging services for web content" on page 3401
Web Content Manager supports for the notification of events such as item state changes, or services starting and stopping. These notifications can be delivered as messages to the Java messaging service.

"Developing basic PAA file applications" on page 3402
Solution developers can create their own Portal Application Archive (PAA) files. The developers can then use the configuration wizard to add on their applications to their IBM WebSphere Portal Express environment.

"Developing advanced PAA file applications" on page 3431
Developers can create their own advanced Portal Application Archive (PAA) file. The advanced PAA file contains custom content. The developers can then use the configuration wizard to add on their applications to their IBM WebSphere Portal Express environment.

"IBM UX Screen Flow Manager" on page 3475
The IBM UX Screen Flow Manager helps operators, developers, and dialog modelers develop fine-granular, small split portlets. Portlets provide the functions for the steps that are presented as screens. You can configure the sequence, transitions, and workflow of a set of screens. Screen flows work like wizards and process the screens to complete the task for the user. Screen flows are completed by a single user and have a short lifetime.

Changing to developer mode

Change to the developer mode to develop portals and portlets. Startup performance is improved within a developer mode environment.

Before you begin

Install IBM WebSphere Portal Express.

About this task

Choose one of the following options to change to developer mode immediately after you install WebSphere Portal Express:

"IBM i: Configuring developer mode" on page 2512
Use the developer mode to improve startup performance and to configure IBM WebSphere Portal Express for development. Developer mode is for a development environment only. Do not use the developer mode in a production environment. You can also run the **optimize-derby-database** task to improve the Derby database performance.

"Linux: Configuring developer mode" on page 2513
Use the developer mode to improve startup performance and to configure IBM WebSphere Portal Express for development. Developer mode is for a

development environment only. Do not use the developer mode in a production environment. You can also run the **optimize-derby-database** task to improve the Derby database performance.

"Windows: Configuring developer mode" on page 2514
Use the developer mode to improve startup performance and to configure IBM WebSphere Portal Express for development. Developer mode is for a development environment only. Do not use the developer mode in a production environment. You can also run the **optimize-derby-database** task to improve the Derby database performance.

IBM i: Configuring developer mode

Use the developer mode to improve startup performance and to configure IBM WebSphere Portal Express for development. Developer mode is for a development environment only. Do not use the developer mode in a production environment. You can also run the **optimize-derby-database** task to improve the Derby database performance.

Before you begin

Install WebSphere Portal Express.

About this task

This task modifies the following components:

JVM The JVM is switched to development mode. This setting is a WebSphere Application Server specific setting independent of WebSphere Portal Express. See the *Application server setting topic* in the related information section for exact changes that are made with this setting. The initial heap size is set to 768 MB to reduce the amount of garbage collection during startup.

Portlets
Portlets and web applications are activated on first access and not at the startup. However, some of the portlets and applications are required at startup. Create a white list, which contains the list of applications, that are required at startup.

Note: To add applications to the white list, modify the *wp_profile_root*/PortalServer/config/StartupPerformance/wp.base_TargetMapExclList.properties file. Add a line such as *App_name*, where *App_name* is the name of the application. Log on to the WebSphere Integrated Solutions Console and go to **Applications** > **Application Types** > **WebSphere enterprise applications** to get a list of available applications.

Procedure

1. Open a command prompt.
2. Stop the WebSphere_Portal server. Go to "Starting and stopping servers, deployment managers, and node agents" on page 1216 for information.
3. Change to the *wp_profile_root*/ConfigEngine directory.
4. Run the ConfigEngine.sh optimize-derby-database task to improve the performance of your Derby database.

Important: This task is appropriate only in a demonstration or development environment that is not configured to use Web Content Manager. You can also run the **optimize-derby-database** task after large data changes in the database.

5. Start the WebSphere_Portal server.
6. Run the `ConfigEngine.sh enable-develop-mode-startup-performance -DWasPassword=`*`password`* tasks. Then, stop and restart the WebSphere_Portal server to propagate your change.
7. Prepare the remote web server for your developer mode.

What to do next

Run the `ConfigEngine.sh disable-develop-mode-startup-performance -DWasPassword=`*`password`* task to revert to a production server. Then, stop and restart the WebSphere_Portal server to propagate your change.

Note: You can run the **disable-develop-mode-startup-performance** task for the following scenarios:

- When you are done developing your portal and portlets.
- If the development settings are not adequate for a special development situation.
- When you cannot re-create a problem on the development server.

Related tasks:

Application server settings

Linux: Configuring developer mode

Use the developer mode to improve startup performance and to configure IBM WebSphere Portal Express for development. Developer mode is for a development environment only. Do not use the developer mode in a production environment. You can also run the **optimize-derby-database** task to improve the Derby database performance.

Before you begin

Install WebSphere Portal Express.

About this task

This task modifies the following components:

JVM The JVM is switched to development mode. This setting is a WebSphere Application Server specific setting independent of WebSphere Portal Express. See the *Application server setting topic* in the related information section for exact changes that are made with this setting. The initial heap size is set to 768 MB to reduce the amount of garbage collection during startup.

Portlets

Portlets and web applications are activated on first access and not at the startup. However, some of the portlets and applications are required at startup. Create a white list, which contains the list of applications, that are required at startup.

Note: To add applications to the white list, modify the *wp_profile_root*`/PortalServer/config/StartupPerformance/wp.base_TargetMapExclList.properties` file. Add a line such as *App_name*,

where *App_name* is the name of the application. Log on to the WebSphere Integrated Solutions Console and go to **Applications** > **Application Types** > **WebSphere enterprise applications** to get a list of available applications.

Procedure

1. Open a command prompt.
2. Stop the WebSphere_Portal server. Go to "Starting and stopping servers, deployment managers, and node agents" on page 1216 for information.
3. Change to the *wp_profile_root*`/ConfigEngine` directory.
4. Run the `./ConfigEngine.sh optimize-derby-database` task to improve the performance of your Derby database.

   Important: This task is appropriate only in a demonstration or development environment that is not configured to use Web Content Manager. You can also run the **`optimize-derby-database`** task after large data changes in the database.
5. Start the WebSphere_Portal server.
6. Run the `./ConfigEngine.sh enable-develop-mode-startup-performance -DWasPassword=`*`password`* tasks. Then, stop and restart the WebSphere_Portal server to propagate your change.
7. Prepare the remote web server for your developer mode.

What to do next

Run the `./ConfigEngine.sh disable-develop-mode-startup-performance -DWasPassword=`*`password`* task to revert to a production server. Then, stop and restart the WebSphere_Portal server to propagate your change.

Note: You can run the **`disable-develop-mode-startup-performance`** task for the following scenarios:

- When you are done developing your portal and portlets.
- If the development settings are not adequate for a special development situation.
- When you cannot re-create a problem on the development server.

Related tasks:

Application server settings

Windows: Configuring developer mode

Use the developer mode to improve startup performance and to configure IBM WebSphere Portal Express for development. Developer mode is for a development environment only. Do not use the developer mode in a production environment. You can also run the **`optimize-derby-database`** task to improve the Derby database performance.

Before you begin

Install WebSphere Portal Express.

About this task

This task modifies the following components:

JVM The JVM is switched to development mode. This setting is a WebSphere Application Server specific setting independent of WebSphere Portal

Express. See the *Application server setting topic* in the related information section for exact changes that are made with this setting. The initial heap size is set to 768 MB to reduce the amount of garbage collection during startup.

Portlets

Portlets and web applications are activated on first access and not at the startup. However, some of the portlets and applications are required at startup. Create a white list, which contains the list of applications, that are required at startup.

Note: To add applications to the white list, modify the *wp_profile_root*`\PortalServer\config\StartupPerformance\wp.base_TargetMapExclList.properties` file. Add a line such as *App_name*, where *App_name* is the name of the application. Log on to the WebSphere Integrated Solutions Console and go to **Applications** > **Application Types** > **WebSphere enterprise applications** to get a list of available applications.

Procedure

1. Open a command prompt.
2. Stop the WebSphere_Portal server. Go to "Starting and stopping servers, deployment managers, and node agents" on page 1216 for information.
3. Change to the *wp_profile_root*`\ConfigEngine` directory.
4. Run the `ConfigEngine.bat optimize-derby-database` task to improve the performance of your Derby database.

   Important: This task is appropriate only in a demonstration or development environment that is not configured to use Web Content Manager. You can also run the **`optimize-derby-database`** task after large data changes in the database.
5. Start the WebSphere_Portal server.
6. Run the `ConfigEngine.bat enable-develop-mode-startup-performance -DWasPassword=`*`password`* tasks. Then, stop and restart the WebSphere_Portal server to propagate your change.
7. Prepare the remote web server for your developer mode.

What to do next

Run the `ConfigEngine.bat disable-develop-mode-startup-performance -DWasPassword=`*`password`* task to revert to a production server. Then, stop and restart the WebSphere_Portal server to propagate your change.

Note: You can run the **`disable-develop-mode-startup-performance`** task for the following scenarios:

- When you are done developing your portal and portlets.
- If the development settings are not adequate for a special development situation.
- When you cannot re-create a problem on the development server.

Related tasks:

Application server settings

Dojo and WebSphere Portal Express

WebSphere Portal Express contains an instance of the IBM Dojo Toolkit, a JavaScript library that is based on the Dojo Toolkit. When you develop components that use Dojo, you must be aware of how the portal uses Dojo, and the tips and restrictions when you use Dojo.

WebSphere Portal Express Version 8.5 comes with six versions of Dojo:

- **Dojo V1.9.x**. This version is packaged in its own web application named Dojo_Resources. You can manage it in the WebSphere Integrated Solutions Console. By default it is deployed at the context root `wps/portal_dojo`. The path for the Dojo V1.9.x files is *PortalServer_root*`/theme/wp.theme.dojo/installedApps/dojo.ear/dojo.war/V1.9`.
- **Dojo V1.7.x**. This version is packaged in its own web application named `Dojo_Resources`. You can manage it in the WebSphere Integrated Solutions Console. By default it is deployed at the context root `wps/portal_dojo`. The path for the Dojo V1.7.x files is *PortalServer_root*`/theme/wp.theme.dojo/installedApps/dojo.ear/dojo.war/v1.7` .
- **Dojo V1.6.x**. This version is packaged in its own web application named `Dojo_Resources`. You can manage it in the WebSphere Integrated Solutions Console. By default it is deployed at the context root `wps/portal_dojo`. The path for the Dojo V1.6.x files is *PortalServer_root*`/theme/wp.theme.dojo/installedApps/dojo.ear/dojo.war/v1.6` .
- **Dojo V1.4.x**. This version is packaged in its own web application named `Dojo_Resources`. You can manage it in the WebSphere Integrated Solutions Console. By default it is deployed at the context root `/portal_dojo`. The path for the Dojo V1.4.x files is *PortalServer_root*`/theme/wp.theme.dojo/installedApps/dojo.ear/dojo.war/v1.4.3` .
- **Dojo V1.3.x**. This version is packaged in its own web application named `Dojo_Resources`. You can manage it in the WebSphere Integrated Solutions Console. By default it is deployed at the context root `/portal_dojo`. The path for the Dojo V1.3.x files is *PortalServer_root*`/theme/wp.theme.dojo/installedApps/dojo.ear/dojo.war` .
- **Dojo V1.1.1**. This version is packaged in the directory *PortalServer_root*`/installer/wp.ear/installableApps/wps_theme.ear/wps_theme.war/themes/dojo/portal_dojo` .

All Dojo packages can be used by IBM and non-IBM components.

Notes:

1. The Portal 8.5 theme uses Dojo V1.9.x by default in WebSphere Portal Express v8.5. This version is the only supported version of Dojo with the Portal 8.5 theme.
2. The Portal 8.0 theme uses Dojo V1.7.x by default in WebSphere Portal Express v8.0. This version is the only supported version of Dojo with the Portal 8.0 theme.
3. The Portal 7.0.0.2 theme uses Dojo V1.6.x by default in WebSphere Portal Express v7.0.0.2. This version is the only supported version of Dojo with the Portal 7.0.0.2 theme.
4. The Page Builder theme uses Dojo V1.4.x by default in WebSphere Portal Express v7.0. The Page Builder theme is deprecated in Portal 8.5, and will no longer be supported in the next release.

5. The Dojo toolkit that is provided with the portal will be updated as needed over time. This toolkit might include entire new Dojo versions, and specific defect fixes. Compatibility of future Dojo versions is defined by the Dojo project.

Where WebSphere Portal Express uses Dojo

The Portal 8.5 theme uses Dojo V1.9.x to support various portlets and edit-mode user interface components. The portlets and components that use Dojo vary over time. The current set of portlets that uses Dojo includes Managed Pages edit-mode user interface components, Search Center, External Search Results, Tagging and Rating, Tag Cloud, Active Site Analytics, Unified Task List, WidgetWrapper Portlet, WCM Rendering Portlet, Web Content Libraries, Web Content Subscribers, Web Content Syndicators, Feed Schedules, Web2Bookmarks Portlet, and CMIS Picker Dialog.

Note: The portal components that use Dojo are only supported for use with the bundled Dojo.

Dojo is packaged as a module as part of the modular architecture for the Portal 8.5 theme. For performance optimization, the minimal profile does not load Dojo at all. The default deferred profile does not load Dojo initially for view mode, but, rather, only when a page is put into edit mode. When you do not load the Dojo module or any module that is a prerequisite for the Dojo module in the non-deferred part of your profile, then you need not pay any performance price of loading Dojo in view mode and at runtime. If you elect, you are now free to use a different JavaScript library in the view mode for your custom theme and runtime. Create and load your own modules for the library.

Using Dojo in your custom portal themes

For information about how to use Dojo in custom portal themes that you create see the topic about Creating a new theme.

Upgrading your custom portal themes to a later Dojo version supported by the portal: To upgrade your existing custom themes to a later supported version of Dojo, see the following sections:

Portal 7.0.0.2 theme, Dojo V1.6.x -> Portal 8.0 theme, Dojo V1.7.x: see Updating a Portal 7.0.0.2 theme to use Dojo 1.7.

Page Builder theme, Dojo V1.4.x -> Portal 7.0.0.2 theme, Dojo V1.6.x: see Updating a Page Builder theme to use Dojo 1.6.

Dojo V1.3.x -> Dojo v1.4.x: see Dojo and WebSphere Portal Express.wp7

Dojo V1.1.x -> Dojo v1.3.x: see Dojo and WebSphere Portal Express.wp7

Dojo usage best practices

When you are working with Dojo and portal components, be aware of the following best practices:

- Only one instance of Dojo can be loaded in a page, and the current Dojo policy is that the first Dojo included in the page takes precedence. Therefore:
  - Dojo can be loaded only once per namespace. You can load more than one version of Dojo in the page by using the Dojo native support for scopes. For details, see the Dojo documentation. For performance reasons, it is best to load only one version of Dojo in the page. However, applications or the

theme can load more Dojo bundles if they are scoped to different namespaces than the Dojo defaults. You need to make sure that applications and portlets that reference Dojo at a particular namespace scope work correctly with the version of Dojo that is mapped to that scope.

– Different themes in the same portal can use different versions of Dojo. You need to make sure that the portlets that run in these themes can work with both Dojo versions.

Support for multiple versions of Dojo at run time creates a more robust application, which is more stable across migrations and upgrades. But the support can be slightly more complex depending on the added code complexity.

- All portlets that use Dojo widgets must manually call the Dojo parser when they are loading. Setting djConfig.parseOnLoad has no effect. Otherwise, no widgets are parsed in the portlet.

  All portlets must pass a markup element to the parser. The markup element can exist only inside the DOM of the portlet. Otherwise, Dojo parses the entire document body every time the parser is called without a markup element. Other portlets might get parsed two or three times, which causes the Dojo parser to fail every time it hits a widget that is already parsed.

  Here is an example of correct usage:

```
<script> dojo.addOnLoad( function () { dojo.parser.parse( "<%=namespace%>portletWidgetContainer" ); } ); </script>
  <div id="<%=namespace%>portletWidgetContainer">
   <div dojoType="some.Widget"></div>
  </div>
```

- The `lotusui30dojo` class is set on the body element in the modularized themes, and its corresponding CSS files are linked in as well. To use a different theme within a particular portlet, do not change the CSS classes of the body element from within the portlet. Because this action has consequences on all other portlets and theme components that use Dijits on the page. Instead, use a separate node within the portlet to contain all the widgets that are used by that portlet. Then, assign the different theme class name on the container node inside the portlet.
- It is important to note that the placement of the theme class (for example, `lotusui30dojo`, `soria`) is vital for the theme to display correctly in Dijit components. If a Dijit component creates any elements as direct children of the body element, instead of or in addition to the same place in the DOM where the Dijit component was attached, then it is important to explicitly assign the class name. The class name is assigned for the secondary theme to the DOM node that is a direct child of the body, in addition to the containing DOM node of the portlet's widgets. For example:

```
<body class="tundra">
 ...
 <!-- Portlet A -->
 <div class="wpsPortletBody">
  <!-- Contents of this portlet -->
  <div class="soria">
   <!-- Any Dijits here will use the soria theme instead of tundra -->
   <button class="dijit dijitReset dijitLeft dijitInline dijitDropDownButton">
   ...
   </button>
  </div>
 </div>
 ...
 <!-- Portlet B -->
 <div class="wpsPortletBody">
  <!-- Any Dijits created here will use the tundra theme -->
  ...
 </div>

 <!-- This table node was created for the dijit.Menu component as part of the
     dijit.form.DropDownButton from Portlet A -->
 <table class="dijit dijitMenu dijitReset dijitMenuTable soria">
  <!-- This menu will use the soria theme instead of tundra, but needs to have it explicitly
```

```
      assigned since none of its ancestors have the soria class applied -->
  ...
 </table>
</body>
```

In this example, when Portlet A creates dijit.form.DropDownButton, the code to create that widget also creates a menu component and a new DOM node under the body. But the code does not assign a CSS class to that DOM node. If nothing else is done after the portlet creates the DropDownButton, then the DropDownButton uses the `soria` class. But the menu that pops up when the button is clicked uses the `tundra` class. In cases like this, it is important to explicitly set the `soria` class on the DOM node that contains the menu.

- If possible, load JavaScript code at the end of the theme markup after the rest of the page is rendered, including the page contents themselves. The initial page can then render quickly without blocking requests to serve large JavaScript files, including custom layer files. However, provide core Dojo functions in the head section of the page so that user interface components such as portlets and widgets can have access to these core functions.
- In relation to loading resources at the end of the theme rather than all inside the head section, the best practice for `dojo.require` statements in portlet applications, widgets, or other user interface components is to include them immediately before the objects or code within the modules that they load are used. This means that if the code is not used until after the page is loaded, for example through an onload handler that is registered by using `dojo.addOnLoad`, then the `dojo.require` statement is best served within the onload handler itself. The theme can then load more resources at the end of the page in layer files after all page components are included. As a result, if the theme already provided those modules in layer files at the end of the markup, the `dojo.require` statements can return immediately, when the onload handlers are fired. This also avoids unnecessary requests for modules that the theme loads but that is not loaded at the point in time when the user interface component is included in the markup on the page.

Dojo usage restrictions

Observe the following restrictions when you use Dojo with the portal:

- Only one instance of Dojo can be loaded in a page.
- Do not rely on making your own updates to the bundled Dojo package. WebSphere Portal Express support is likely to update individual files over time, and might even replace the entire package.

Related tasks:

"Updating custom theme Dojo references" on page 893
The default Dojo context root in WebSphere Portal Express is `/WpsContextRoot/portal_dojo`. You can find the value of `WpsContextRoot` in *wp_profile_root*`/ConfigEngine/properties/wkplc.properties`.

Extending WebSphere Portal class path

Custom code must either be added by a PAA or as APP through the administrative console. Adding custom code in shared application or `wp_profile/classes` is critical because it can be overwritten, deleted by updates or migration.

About this task

There are several options to add the general code that is not part of an EAR or WAR files.

Procedure

1. Add the custom code to a shared library.
   a. Use the administrative console to define a shared library, by creating and associating the shared library with an application, module, or server. For more information, see *Managing shared libraries*.

      Documentation resource: Managing shared libraries.
   b. Add the shared library to the custom applications or to the entire WebSphere Portal Express Server.
      - Adding the shared library to the custom application.

        You can associate a shared library with an application or module. Classes that are represented by the shared library are then loaded in the application's class loader, making the classes available to the application. For more information, see *Associating shared libraries with applications or modules*

        Documentation Resource: Adding the shared library to the custom application
      - Adding the shared library to the entire WebSphere Portal Express.

        You can associate shared libraries with the class loader of a server. Classes that are represented by the shared library are then loaded in a server-wide class loader, making the classes available to all applications deployed on the server. For more information, see *Associating shared libraries with servers.*

        Documentation Resource: Add the shared library to the entire WebSphere Portal
2. If you do not want to create a shared library, then you can drop the code into the profile/classes directory `/opt/IBM/WebSphere/wp_profile/classes/MyCustom.jar`. The code is then available to all applications in the server.
3. If you are developing a PAA, include the compressed files in the PAA `components/COMPONENTNAME/shared/app` or `components/COMPONENTNAME/shared/ext` directories. The solution installer handles creating the shared libraries for these files.

Developing themes and skins

You can create themes using modules to contribute to separate areas of pages to provide flexibility, enhance the user experience, and maximize performance. To optimize themes on your website, use the theme optimization module framework. The framework separates feature-specific logic and capabilities from the theme code.

"Understanding the Portal Version 8.5 modularized theme" on page 2521
Modern websites and browsers enable incredible new capabilities that can greatly enhance your user's web experiences. However, these capabilities are not without cost in terms of large page sizes and more processing in the browser when each page is rendered. These capabilities are worth it when you need them, but removing them for an entire site or including them only on pages that take advantage of these capabilities provides for more flexibility.

"The module framework" on page 2526
The module framework allows extensions to contribute to different areas of a page to provide flexibility, enhance the user experience, and maximize performance.

"Customizing the theme" on page 2658
The module framework allows themes to be customized in order to provide flexibility, enhance the user experience, and maximize performance.

"Developing themes for a production portal" on page 2813
Use the theme artifacts to package a theme for staging to production.

"Device classes" on page 2821
Device classes are used in IBM WebSphere Portal Express as an abstraction for common properties for the device of a client. For instance, tablet computers can be grouped into a device class `tablets`, since they share a form factor and possibly other traits such as touch interface, or additional hardware sensors.

"Responsive Web Design" on page 2826
Responsive Web Design provides content parity between mobile devices and desktop channels, which enhances user experience and brand consistency. Seamless changes in screen size, from small to large, are now possible while the order of the content is maintained. Content maintenance is simplified by having one site that is represented by one set of assets.

Understanding the Portal Version 8.5 modularized theme

Modern websites and browsers enable incredible new capabilities that can greatly enhance your user's web experiences. However, these capabilities are not without cost in terms of large page sizes and more processing in the browser when each page is rendered. These capabilities are worth it when you need them, but removing them for an entire site or including them only on pages that take advantage of these capabilities provides for more flexibility.

The new modularized theme provides a flexible framework that:

- Minimizes download size by giving you the control to specify just the capabilities that are needed for a certain scenario or use case.
- Minimizes the number of requests by combining necessary resources.

Previous themes required a monolithic design and that the same content was downloaded for every page. Theme optimization allows the theme to be highly adaptive to the content you are displaying on certain pages. For example, on pages where only simple content is displayed you can define a lightweight profile. A lightweight profile causes the system to download few static resources such as JavaScript and CSS files. However, on pages where more advanced scenarios are required you can choose to switch to a more powerful profile that causes more resources to download than on the other pages. This way you have only the capabilities you need on certain pages, but all other pages do not pay the penalty. As a result the overall system performance increases significantly.

Theme optimization uses modules and profiles to achieve the flexibility that allows you to achieve better performance. Modules are the components of the new theme that define capabilities. Examples are Tagging&Rating, Dojo, or jQuery. Profiles define sets of modules which can be assigned per page. A default profile is used if no page-specific profile is defined.

By applying these concepts it is possible to turn on and off an arbitrary number of features for certain pages, develop modules independent of each other for greater development speed and flexibility, easily add new capabilities later on into an existing theme and build an altogether new theme with the existing one. This building block concept allows the new theme to work side by side through self contained modules without impacting the existing theme.

The Portal Version 8.5 theme contains three types of files: JavaScript, dynamic content (JSP files) and static resources.

"Static resources"
Static resources include the markup that is defined by `.html`, `.css`, and `.js` files that are used by the theme. Some `.json` files are used to define menu options, module definitions, and module profiles.

"Dynamic content (jsp) resources" on page 2525
Dynamic content includes resources that are defined by `jsp` files that are used by the theme.

"Style resources" on page 2526
JavaScript content includes module resources defined by `js` files used by the theme.

Static resources

Static resources include the markup that is defined by `.html`, `.css`, and `.js` files that are used by the theme. Some `.json` files are used to define menu options, module definitions, and module profiles.

For more information about modularization and modules that are provided by the IBM WebSphere Portal Express 8.5 Theme, see *Modules*. Each WebDAV folder has a readme file with additional information about each section.

On WebDAV: `http://host:port/wps/mycontenthandler/dav/fs-type1`.

Theme module definitions

`fs-type1:\themes\Portal8.5\contributions`

`theme.json`: Module definitions for **`wp_theme_portal_85`**.

- **`wp_theme_menus`**
- **`wp_theme_portal_85`**
- **`wp_theme_skin_region`**
- **`wp_theme_high_contrast`**
- CF06 Removed in combined cumulative fix 06
  - **`wp_theme_edit`**

CF06 `theme_edit.json`: Contribution that is loaded when edit mode is turned on. Separates view mode from edit mode. Module definitions for **`wp_theme_portal_edit_85`**

- **`wp_theme_portal_edit_85`**

`dojo19.json`: Module definitions for Dojo, dijit, and dojox from version 1.9.1.

`oneui303.json`: Module definitions for OneUI version 3.0.3.

- **`wp_one_ui`**
- **`wp_one_ui_dijit`**

`contextmenu.json`: Contains component action menu presentation templates.

`simple_contextmenu.json`: Contains simple menu presentation templates.

`worklight61.json`: Contains Worklight mappings.

Style templates

`fs-type1:\themes\Portal8.5\css`

Style templates contain the `css` definitions for the styles that are provided by the theme. The style options are shown in the **Style** tab on the site toolbar when in **Edit Mode**.

Each of the listed directories has a css definitions file for that style and an icon that contains the image that represents the style. For the black style, there is a `black.css` and an `icon.gif` file under the black directory.

The `css` directory contains the following directories and files:

- Directories: `black`, `blue`, `gold`, `green`, `orange`, `purple`, `red`, `white`, `images`, and `default`.
- Files:
  - `master.css`: Includes the `sideNav.css`, and `default.css` files.
    - CF06 Removed in combined cumulative fix 06
      - `contextmenu.css`
      - `contextmenuCommon.css`
  - `masterRTL.css`: Includes the `sideNavRTL.css`, `defaultRTL.css`, and `default.css` files.
    - CF06 Removed in combined cumulative fix 06
      - `contextmenuRTL.css`
      - `contextmenuCommon.css`

  These files contain the compressed (minified) versions of the css definitions.

  Note: Compressing the files makes the files harder to read, so uncompressed versions are provided. It is best to use uncompressed versions when you are editing and debugging your theme CSS. For more information about using uncompressed versions of css files and special instructions on handling the `master.css` file, see *Enabling the uncompressed CSS files*.

The `default` directory includes the base styles for the theme. The `default_view.css` file contains styles for view mode. The `default_edit.css` file contains styles for edit mode. The `default_search.css` file contains styles for search. These styles are applied to the default theme and optimize it for mobile devices. These styles apply to menus, navigation, layout, and footer.

The `images` directory contains the following common images:

- `blank.gif`: A blank gif file that is used for spacing.
- `loading.gif`: The image that is shown when the page or other resource is loading.
- `loadingDark.gif`: A darker version of the `loading.gif` image.
- `master.png`: A sprite file with common images that are used in the theme. Some of the images in the sprite include the IBM and WebSphere Portal Express logos, search icon, the pencil icon for the edit/view mode toggle, and various sized arrow images.

Images

`fs-type1:\themes\Portal8.5\images`

- Directories: `addContent` and `taggingAndRating`
- Files: `favicon.ico`

js

`fs-type1:\themes\Portal8.5\js`

- `head.js`: This file contains JavaScript processing included in the head section.
- `skinRegion.js`: This file contains JavaScript that applies accessible areas to the skins.
- `mobilenav.js`: This file contains JabaScript processing for the mobile areas of the theme.
- CF06 Removed in combined cumulative fix 06
  - `contextmenu.js`: This file contains the JavaScript implementation of the menu.
  - `theme.js`: This file contains JavaScript processing for the edit capabilities in the theme.
  - `highContrast.js`: This file contains JavaScript processing for high contrast enabled systems. This file was moved to a different module in combined cumulative fix 06.

Layout templates

`fs-type1:\themes\Portal8.5\layout-templates`

Each directory contains a layout template. These templates are shown on the **Change Layout** tab of the site toolbar when in edit mode. Each directory contains a `layout.html` file that defines the layout and an `icon.gif` file that is a thumbnail image of the layout.

The `layout.html` file defines the decorations of the content area that are different between pages.

The layout templates are scoped to the Portal 8.5 theme.

Directories: `1Column`, `1Row2ColumnUnequal`, `1Row3ColumnEqual`, `2ColumnEqual`, `2ColumnLeft`, `2ColumnRight`, `2Row`, `3ColumnCenter`, `3ColumnEqual`, `TopColumn2ColumnUnequal`, and `TopColumn3ColumnCenter`.

Menu definitions

`fs-type1:\themes\Portal8.5\menuDefinitions`

Link to menu framework

- `pageAction.json`: Defines the menu items for the **Page Action** menu. Items include the definitions for the Tagging and Rating, Impersonation, and other page-related menu items.
- `skinAction.json`: Defines menu items for the portlet menu. It includes the definitions for the actions that are available for the portlet. Included items are Wiring, Delete, Edit Settings, Minimize, and Maximize.

nls

`fs-type1:\themes\Portal8.5\nls`

This directory contains translated HTML files for the theme.

Profiles

`fs-type1:\themes\Portal8.5\profiles`

These are the several module profiles that define which modules are loaded when a page is loaded. For more information about deferred modules, see *Modules, and Deferred and non-deferred modules*. For more information about profiles, see *Included Profiles*.

Skins

`fs-type1:\themes\Portal8.5\skins`

Directories: `Hidden`, `NoSkin`, and `Standard`.

These directories contain the ready-to-use skins that are available with the WebSphere Portal Express 8.5 theme: Hidden, NoSkin, and Standard. Each directory has a `skin.html` file, which defines the markup for the skin.

Localized files:

- `fs-type1:\themes\Portal8.5\skins\Standard\nls`
- `fs-type1:\themes\Portal8.5\skins\Hidden\nls`
- `fs-type1:\themes\Portal8.5\skins\NoSkin\nls`

Theme markup

`fs-type1:\themes\Portal8.5\`

Localized versions: `fs-type1:\themes\Portal8.5\nls`

- `Plain.html` Defines a markup that has no decorations. This file is used for some basic dialogs in the theme.
- `theme.html`: Defines the markup that is identical for all the pages to which this theme is applied.
- `theme_sidenav.html`: Defines a theme layout that uses a side navigation.

Dynamic content (jsp) resources

Dynamic content includes resources that are defined by `jsp` files that are used by the theme.

The entry point for a theme is either `Default.jsp` or `Plain.jsp`, both of these files are located here:

PortalServer_root`\theme\wp.theme.modules\webapp\installedApps\ThemeModules.ear\ThemeModules.war\themes\html\`

- `Default.jsp`: Is a main entry point for the theme and bootstraps any required infrastructure, and then delegates all markup rendering to the static template file, which is usually `theme.html`.

- `Plain.jsp`: Is an alternative entry point. This jsp is commonly used for helps or rendering a portlet with an iframe skin.
- `includePortalTaglibs.jspf`: Includes the IBM WebSphere Portal Express taglibs.

dynamicSpots

PortalServer_root`\theme\wp.theme.themes\default85\installedApps\DefaultTheme85.ear\DefaultTheme85.war\themes\html\dynamicSpots`

- `commonActions.jsp`: Provides the common actions in the banner sections and the **Actions** menu. The **Actions** menu is not available to anonymous users.
- `crumbTrial.jsp`: Provides the crumb trail information for the location.
- `footer.jsp`: Provides the footer information for the page.
- `head.jsp`: Provides the header information for the page.
- `navigation.jsp`: Provides the navigation controls for the page.
- `mobileNavigation.jsp`: Provides the mobile navigation controls for the page.
- `sideNavigation.jsp`: Provides the side navigation controls for the page.
- `status.jsp`: Template for error, warning, and information messages.

Style resources

JavaScript content includes module resources defined by `js` files used by the theme.

OneUI CSS

The common CSS elements used across Lotus products. For the IBM WebSphere Portal Express version 8.5 theme, these styles are only for content. The WebSphere Portal Express version 8.5 theme markup is independent of OneUI.

For more information, see the *ICS UI Developer Guide*.

Related information:

ICS UI Developer Guide

The module framework

The module framework allows extensions to contribute to different areas of a page to provide flexibility, enhance the user experience, and maximize performance.

The modularization framework decouples feature enablement from the theme code itself. Themes can be developed more easily with little knowledge about the details of how underlying code for a feature works. Logical points are provided where modules can contribute data into a theme at run time and then optimize those contributions by combining them where possible. Multiple disparate remote sources can be combined into one request for greater performance.

The features of a theme can be enabled and disabled by using a profile to configure them. You can then focus your time on the interface design of the theme without being concerned about the details of getting the features to work correctly within their theme. It is easy to turn off features that are not needed in one environment that might be used in another environment. For example, you can disable editing capabilities in a production portal environment, while they are enabled in a development environment. The same theme code can be used across such environments where the only variable is the module inclusion profile.

CF03 You can set dependencies on features in portlets and profiles. The features are automatically loaded onto the page in an aggregated way. Your profile does not need to contain more modules that are required by portlets. Your profile can be focused on theme capabilities. Modules that are only required by portlets can be loaded by the portlets. Portlet dependencies are loaded independently from the profile. If many portlets use the same features across many pages, you can add this feature to the profile for better cache performance. If you created your theme based on WebSphere Portal Express version 8.5 before CF03, then you must enable this feature to use it. Themes that are based on WebSphere Portal Express version 8.5 created in CF03 or after have this feature enabled by default.

Modules are registered extensions that are then used by a module profile. Each module is enumerated by the modules unique identifiers. A module might require other modules to allow the automatic inclusion of necessary code that is required to make a particular feature work. For example, you can use of the Dojo Toolkit within a module. A module can use the Dojo Toolkit to build custom widgets. To separate the code for the module from the Dojo code, the module requires certain Dojo modules to ensure that the code is loaded in the correct sequence. This separation allows greater serviceability by decoupling the packaging of the code for each module.

"Basic artifacts and their relation" on page 2528
The theme modularization framework foresees the following major artifacts and relations to one another.

"Contribution types" on page 2530
Modules can contribute different types of data to the extension points within the page.

"Deferred and nondeferred modules" on page 2533
The module framework allows a profile to specify whether to defer a particular module. By default, a module is loaded when the initial page loads, but modules that are specified as deferred modules are loaded after the page loads.

CF03 "Resource Aggregator overview" on page 2534
When the page renders, the Resource Aggregator processes all of the modules coming from the profile.

"Module dependencies in portlets" on page 2535
When a portlet requires existing client-side capabilities, such as Dojo, loaded on the page, it can define a set of portlet preferences that declare the capabilities it requires.

CF03 "Change the auto-loading of portlet capabilities" on page 2538
You can change if portlets automatically load their dependent capabilities for a theme.

"Response rendering for themes" on page 2540
To decrease the response time of your portal, a template parser resolves which modules are needed and collects all of the modules that are enabled by the current profile. Parts of the content are parsed and displayed as soon as possible.

"Writing modules" on page 2542
You can define global or theme-specific contributions that contain a theme module, which applies to different scopes through theme profiles. A module defines its contributions by using a `plugin.xml` or JSON file.

"Theme Optimization Analyzer" on page 2566
Use the Theme Optimization Analyzer to view, but not edit, all parts of the theme optimization framework inside of WebSphere Portal. With the portlet,

you can easily see which pages have specific profiles that are set or inherited. You can also see which profiles are available and belong to which theme.

"Troubleshooting modular themes" on page 2600
You can debug your modules to improve performance.

"Specify profiles" on page 2605
You can specify the profiles that you want to use two different ways depending on whether you are an administrator or a user. Profiles help define which modules are used to render a page.

"Modules that are provided with the modularized theme" on page 2608
WebSphere Portal Express provides a set of ready-to-use modules.

"Modules that add features to a theme" on page 2652
Theme modules contribute resources, such as JavaScript, CSS and HTML, to a theme or page. The following modules come with the modularized theme and specifically contribute a feature rather than working in the background. The modules can be added to a theme by listing them in the theme's default profile, or to a single page or set of pages.

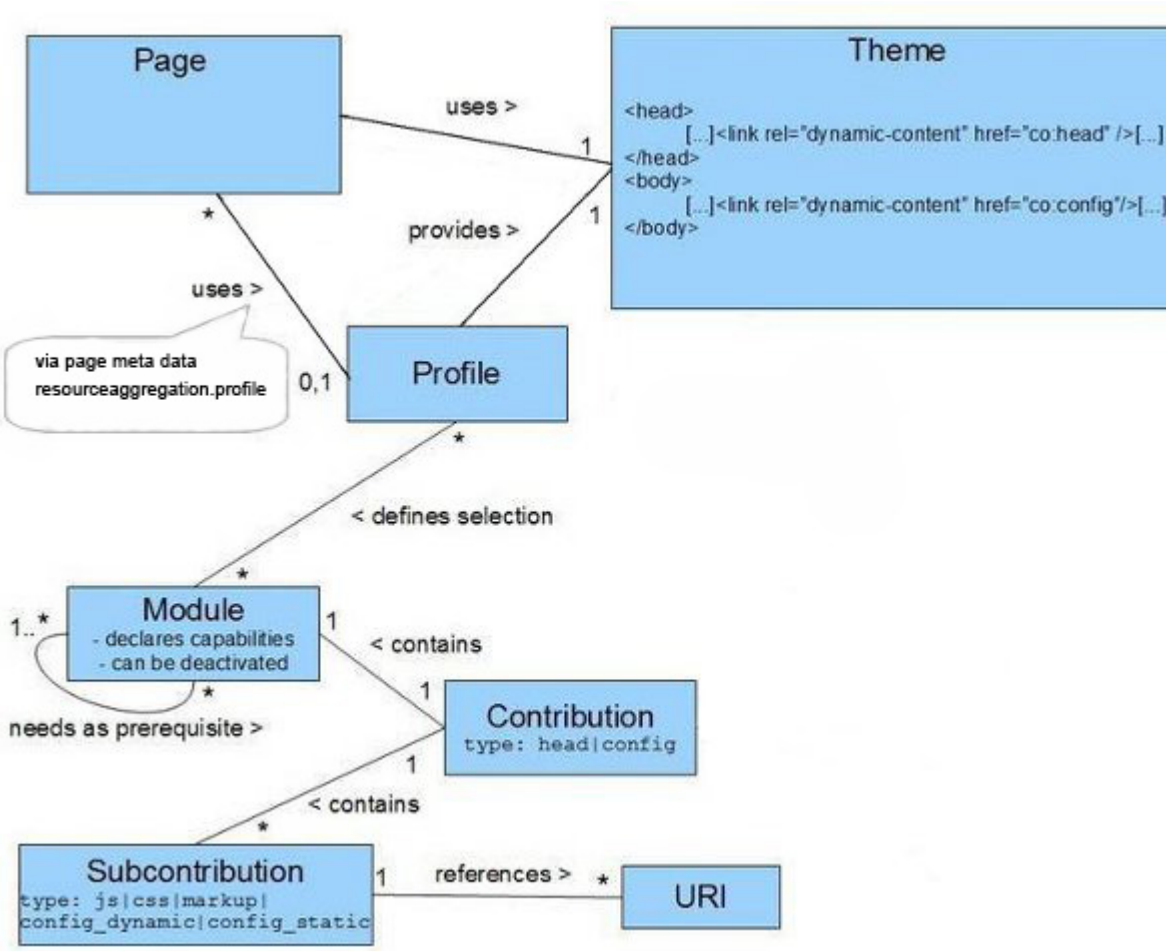
"Adding or removing a module from a profile" on page 2656
To add or remove a module, update the profile that is used to render a page for the theme.

"ConfigEngine tasks for creating a new profile based on a template and an existing profile in the system" on page 2657
You can extend the theme module framework with a new ConfigEngine task.

Basic artifacts and their relation

The theme modularization framework foresees the following major artifacts and relations to one another.

The relations of the major artifacts are depicted in the following figure:



A theme is assigned to a portal page. Dynamic content spots are used to write all of the data for modules that have contributions for a certain type. The module framework provides the dynamic content spots **co:config** and **co:head**. The theme must provide one or more profiles, which declares the set of modules that are used for that profile. One of these profiles is used by each page to which the theme is applied. The profile that is used for the page is the default profile that the theme developer or administrator defines for the theme by setting the theme metadata **resourceaggregation.profile**.

The Page administrator can explicitly overwrite the default by setting the page metadata **resourceaggregation.profile**.

The modules available in the system can either be registered through the extension point **com.ibm.portal.resourceaggregator.module** in a `plugin.xml` file, or using a JSON file that is located within a theme in the folder `<theme-root>/contributions`. This path is not customizable.

For example, the head spot, `<link rel="dynamic-content" href="co:head"></link>`, and aggregates all contributions of type head.

You choose where to place dynamic content spots if the following constraints are fulfilled:

- The head spot must be located inside the <head> element in the page markup. The <head> element must be placed before the <body> element in the page markup.
- The config spot must be located inside the <body> element in the page markup. Make the config spot the last element before the closing tag.

Contributions to the head extension point are best used for data that must be loaded at the beginning of the page. These contributions include producing markup that is only valid inside the <head> element or providing core functions that must be available to other inline code within the page body. For example, <link> elements can show only inside the head element, therefore all CSS must be added to the head contribution. Also, any inline JavaScript that uses a utility function to attach an event handler to the onload event, such as **dojo.addOnLoad**, depends on defining that function earlier in the page markup.

Place the config spot at the end of the page body element. This placement allows the bulk of the JavaScript resources and other enablement code to be written to the markup after the page layout and content area. The perceived responsiveness of the server increases by showing the page content while the browser is still downloading resources that are defined after the content area. However, there is no guarantee that the config spot is placed there, and it is valid for a theme designer to place the config spot inside the <head> element when it fulfills the first constraint.

Because the location of the config spot can vary, any code that displays inside the page content does not assume that the config spot was already generated. The JavaScript code in the content area that depends on other JavaScript modules that are loaded by the config spot are deferred until the onload event is triggered, which guarantees that the code inside a portlet, for example, always works regardless of the position of the config spot. An example is portlet JavaScript that creates client-side widgets on page load. Widget modules that themselves can be loaded by the config spot while the code that creates and uses them is run on load.

Modules can declare dependencies on prerequisite modules. Additionally, they can declare their capabilities, which enable portlets and other code on the page to query the availability of a certain capability.

CF03 Portlets can declare a dependency on capabilities in a way that automatically downloads all resources of a module. You do not need to add this module to the profile, but you must set this behavior in your theme.

Contribution types

Modules can contribute different types of data to the extension points within the page.

By specifying the type of data, a module can optimize how that data is loaded on the page. For instance, all of the JavaScript contributions to the co:config spot from multiple modules can be loaded by a single request to the server. That request generates a single response that contains the data for each contribution.

Contribution type: head

CSS

Contributions of this type must be valid Cascading Style Sheet (CSS) syntax and are always placed inside the `<head>` element. They are only valid in the `co:head` extension point.

Use generated CSS that uses relative URLs when you reference other resources like images or other CSS documents. By using relative URLs, resources can be resolved regardless of the context root in which they are placed.

JavaScript code

Contributions that add static JavaScript code are added it to this type so that it is properly cached and potentially optimized by the framework.

Contributions of this type cannot vary based on the client to further optimize the code. For example, do not use a data source that generates different data for different browsers to eliminate code paths that branch based on the browser type or version. Instead, the data source must generate the same data that works across all supported browsers. Users can then download and save the combined output as a static resource in a separate domain.

Important: It is the responsibility of the contributor to ensure that the content of the contribution does not include malicious code.

Static JavaScript configuration

Contributions to this type must be valid JavaScript code, publicly cacheable, and have no dependencies on the current user, navigation state, or runtime context.

It is common for static JavaScript code to use JavaScript variables that are defined in the global scope to have access to context information that must be provided outside the static script code.

Some of this configuration might be static across pages and can change only when the server is restarted, such as values that are retrieved from Resource Environment Provider custom properties. This data can be loaded as an external script so that it can be cached across users and pages.

Because it is possible to change through configuration changes, this configuration typically has relatively short cache expiration times so that it is still publicly cacheable but can still expire to pick up potential configuration changes.

The link to the URL that creates the aggregated content is part of a script tag.

Important: It is the responsibility of the contributor to ensure that the content of the contribution does not include malicious code. For example, configuration values that are read by a data source must be escaped before serializing the data.

Dynamic JavaScript configuration

Some configuration data that is used by JavaScript code is dynamic and susceptible to changing across pages or users. This configuration includes data, such as the current locale or URLs that are generated dynamically by the server.

Contributions to this type must be valid JavaScript code and are written to the markup inside an inline script element so that they are never cached.

Important: It is the responsibility of the contributor to ensure that the content of the contribution does not include malicious code. For example, configuration values that are read by a data source must be escaped before serializing the data.

Markup

Contributions to this type must be valid HTML in context of the extension point where they contribute to. For instance, contributions to the `co:head` extension point can include markup that is only valid for the <head> section of the page. Alternatively, contributions to the `co:config` extension point can include markup that is only valid inside the <body> tag.

This type is best used for contributions whose markup is not primarily visual but rather semantic, such as resource loading.

Modules that have visual markup must be documented and provided through a **`POC URI`** dynamic content spot. You can place this contribution in a specific place within the theme template relative to the design of the theme. If these unique location-important markup contributions are intended to be configurable or modified by a user, the dynamic content spot mapping data source might be used to map the **`POC URI`** to an alternative **`URI`**. Also, you can associate a dynamic content spot with a module ID so that the rendering of the data can be influenced by the profile that is used for the page.

It is best not to use portal render request-dependent attributes because those attributes sometimes are not available in all cases. For example, when used in deferred mode, the render context is not available.

Contribution type: `config`

- JavaScript code
- Static JavaScript configuration
- Dynamic JavaScript configuration
- Markup

See the head section for details about each of these subcontributions.

Contribution type: `dyn-cs`

Contributions of this type allow you to define dynamic content spots through modules instead of defining them through Resource Environment Providers. This also allows you to overwrite dynamic content spots for different pages by using different modules on the profiles. The only allowed subcontribution is markup. For more information, see *Dynamic content spots*.

Contribution type: `menu`

The only allowed subcontribution is JSON. For more information, see *Menu framework*.

Contribution type: `xslt`

The only allowed subcontribution is `xslt`. You can use `xslt` to gather `xslt` resources along the module hierarchy.

Deferred and nondeferred modules

The module framework allows a profile to specify whether to defer a particular module. By default, a module is loaded when the initial page loads, but modules that are specified as deferred modules are loaded after the page loads.

Nondeferred modules are enabled whenever a request for a page is made to the portal servlet that results in a full page refresh. Use a client-side JavaScript module to load resources that are associated with deferred modules on demand. For example, load deferred modules when entering edit mode for a page. Resources that are not required in view mode can be lazy loaded when the page mode is changed to edit. For more information, see *i$ JavaScript API specification*.

Note: If you enter edit mode while using the deferred profile available out of box, the following error displays when using a JavaScript console: `dojo.back.init() must be called before the DOM has loaded. If using xdomain loading or djConfig.debugAtAllCosts, include dojo.back in a build layer.` This error is thrown by Dojo because the `dojo.back` package is loaded in a deferred way. This code path is used only by older browsers, which are no longer supported. This JavaScript error has no affect on function.

If a nondeferred module requires a deferred module, the server-side combiner framework promotes the deferred module to be nondeferred. The promoted module is then loaded during the initial page rendering process. The module is not deferred, and all of its contributions to each extension point are displayed in view mode. Also, the contributions are not included when any remaining deferred modules are loaded later.

Because deferred modules are loaded distinctly after a page load, the types of resources that can be deferred are necessarily a subset of what can be loaded. CSS, JavaScript code, and markup can be deferred. Therefore, the following rules define when contributions to various places are loaded for deferred modules.

- CSS contributions to the head are deferred and then inserted into the `<head>` element on demand by using the `<link>` element.
- JavaScript configuration, both static and dynamic for both head and config spots, is deferred and loaded as JavaScript.
- Static JavaScript code contributions to the head and config spots are deferred and loaded as JavaScript.
- Markup contributions are lazy loaded by requesting the markup data for all deferred modules that contribute to the config or head markup section. This data is inserted into the page at the appropriate location that is based on where the spot is defined by the theme template.

Note: Because markup contributions can be lazy loaded when a module is deferred, certain limitations apply to the markup inserted dynamically using JavaScript. Script elements, for example, do not run when inserted into the markup as an HTML string. Modules that can be deferred must not generate script elements in their markup contribution to the config spot, unless they are used for another semantic purpose. For example, setting the *type* attribute to some value unknown to the client browser. The framework does not check for or handle any markup that results in side effects or unintended behavior. It is up to the module developer to handle any unintended behavior.

Do not use portal render request-dependent attributes because those attributes are not available in all cases. For example, when used in deferred mode, the render context is not available.

Resource Aggregator overview

When the page renders, the Resource Aggregator processes all of the modules coming from the profile.

The module framework is based on modules that are specified in a profile, and a profile is assigned to a page. It takes all of the resources coming from the modules and aggregates, or combines, them into as few URIs to link on the page as possible to achieve optimal performance. Non-deferred modules are kept separate from deferred modules, so there ends up being two sets of URIs.

Note: You can turn Remote Debugging on to break apart each resource as its own individual, uncompressed request for debugging purposes. For details on how to turn Remote Debugging on and off, see *Theme Optimization Analyzer Utilities*.

For more information on how to create modules and profiles, see *Writing modules*. For more information on how to assign a profile, see *Specifying profiles*.

CF03

Resource aggregation for portlets

In addition to processing the modules coming from the profile, the Resource Aggregator can also process all of the modules coming from the capability dependencies of all of the portlets on a page. The Resource Aggregator processes portlet modules for any theme that has its `metadata resourceaggregation.autoLoadPortletCapabilities` property set to true. For more information about how to set this metadata flag, see *Changing the auto-loading of portlet capabilities*.

For more information about how to specify portlet capability dependencies, see *Module dependencies in portlets*.

With auto-loading of portlet capabilities on, the Resource Aggregator also processes all of the modules coming from all of the capability dependencies of all of the portlets on the page. It processes them while also processing the modules coming from the profile. If the same module or subcontribution is encountered more than once, priority is given in the following order, first-to-last.

1. non-deferred from profile
2. non-deferred from portlets
3. deferred from profile
4. deferred from portlets

For example, if a profile uses Dojo deferred but a portlet uses Dojo non-deferred, Dojo loads non-deferred as part of the combined ResourceAggregator request for the portlets.

When everything is sorted into one of the four buckets, the Resource Aggregator takes all of the resources coming from the modules of each bucket and aggregates, or combines, them into as few URIs to link on the page as possible. There are four sets of URIs, or up to double the number of total URIs, or requests, for the page as compared to when auto-loading of portlet capabilities is off.

CF03

Performance

Auto-loading enhances the ease of use of your system. Users can choose any portlet on a page, if the portlets declare their capabailites, without worrying about the theme or profile. Fewer profiles are needed and the profiles can be smaller.

Auto-loading uses caching on both the client and server level, so the performance of your portal is nearly the same as when it is turned off. You can also mitigate the slight performance drop by adding modules to the theme. The more common or shared a module is, the more it is beneficial to include it in a profile. For example, if a module is used often by portlets on many pages, list that module in the profile so it can be cached as part of the profile URIs. The profile URIs change infrequently and remain cached.

CF03

ResourceCombinerService API

When a portlet must link to its own application-specific resources, it can do that in CF03 with the ResourceCombinerService public API. Previously, you had to link those resources individually. The ResourceCombinerService API can be called to combine resources to into a single optimized URI for optimum performance. For more information, see ResourceCombinerService public API in the *API Javadoc*.

The API has the same flexibility as the Portal module definition syntax to support all the URI possibilities. It has flexibility to support IBM Web Content Manager content URIs, deviceClass classes and equations for mobile, language locales for natural language system, type of debug for the uncompressed version of resources, and type of rtl for bidirectional languages.

The portlet can call the API in the portlet processing, such as in `doHead` for head contributions, or in `doRender` for body contributions. The portlet must manage the dynamic linking of the URIs into the head or body so it can manage the order and grouping of resources. The portlet must manage the cache headers within their datasources. The default cache headers are defined as part of global IBM WebSphere Application Server REP settings. They are `Cache-Control: public, max-age=86400`.

In most case, the returned list from getCombinedURI contains one URI. In a few cases, for example, with CSS size limits in Internet Explorer, it may return multiple URIs. The portlet must code to handle 1 to n.

Note: You must use this API with application-specific resources only. Shared, non-application-specific resources must be defined as modules or capabilities and specified as the portlet module dependencies, using the capability portlet preferences. It optimizes performance. It ensures multiple portlets on a page do not load the same resources. And it controls the order of resources that are loaded, because shared resources must load before application-specific resources.

Related information:

IBM Digital Experience V8.5 API and Developer Reference Documentation

Module dependencies in portlets

When a portlet requires existing client-side capabilities, such as Dojo, loaded on the page, it can define a set of portlet preferences that declare the capabilities it requires.

Define two preferences for each capability the portlet depends upon. One preference that describes the ID of the capability and one that defines the minimal value or version in the following format, `major.minor.revision`:

ID:

```
Preference Name: capability.<sequence-number>.id
Preference Value : <the capability name>
```

Value:

```
Preference Name: capability.<sequence-number>.minValue
Preference Value: <the capability value>
```

Sample

```
<portlet-preferences>
    <preference>
        <name>capability.1.id</name>
        <value>dojo</value>
        <read-only>true</read-only>
    </preference>
    <preference>
        <name>capability.1.minValue</name>
        <value>1.6</value>
        <read-only>true</read-only>
    </preference>

    <preference>
        <name>capability.2.id</name>
        <value>oneUI</value>
        <read-only>true</read-only>
    </preference>
    <preference>
        <name>capability.2.minValue</name>
        <value>3.0.1</value>
        <read-only>true</read-only>
    </preference>

    <preference>
        <name>capabilities.selfManaged</name>
        <value>false</value>
        <read-only>true</read-only>
    </preference>
</portlet-preferences>
```

CF03 In this version, `capability.`*`sequence-number`*`.minValue` is optional. If you do not know the version numbers or the portlet works with any version of the capability, then do not specify the minValue preference, and the system loads the highest available version of the capability.

CF03 There is not always a capability that is defined for every module. Many modules are not part of any capability. Every module automatically surfaces itself as an implicit capability, or a capability with the same name and version number as the module. If there is no capability or you only know the name of the module, you can specify the module name for the value of `capability.`*`sequence-number`*`.id`.

For the list of available capabilities, see *Modules provided with the WebSphere Portal theme*.

Self managed

In addition to the definition of the capabilities, the portlet must define a `capabilities.selfManaged` preference that describes whether the portlet handles the error case of missing capabilities itself or not. To delegate the handling of the error case to WebSphere Portal Express, the preference `capabilities.selfManaged` must be set to `false`.

Note: A portlet can either handle all capability dependencies or none.

If this preference is not set, the framework expects that the portlet itself manages the handling of the error case of a missing capability dependency.

Non-self managed error handling

When `capabilities.selfManaged` is set to `false`, WebSphere Portal Express handles the error case of missing capability dependencies. It displays error messages. One set of error messages, error codes: `EJPNK0022E`, `EJPNK0023E`, and `EJPNK0024E`, can display at the beginning of the page when you try to add a portlet to a page. The other set of error messages, error codes: `EJPNK0026E`, `EJPNK0027E`, and`EJPNK0028E` can display within the portlet when a portlet is already on a page and the page is rendered. The messages contain detailed information for each portlet about which capabilities are missing entirely or do not meet the minimum version required.

The non-self managed error handling can also be turned off for certain situations, such as for maximum performance on a production system or in the remote rendering use case (WSRP) where the consumer is not of type WebSphere Portal Express. For more information, see *Configuration settings for capability filters*.

Loading the capabilities

While each portlet declares the capabilities the portlet depends on, those capabilities and their modules do not load automatically. The loading of the modules is done through the profile that is assigned to the page, so you must assign a profile that is adequate for the portlets that are to be used on a page.

If the non-self managed error handling messages appear, they mean that the required capabilities are missing from the profile. The errors can be resolved by assigning a different profile to the page.

CF03

Auto-loading the capabilities

All portlets can automatically load their dependent capabilities, independent of which profile is assigned to the page. This feature can be turned on per theme in the theme metadata by setting `resourceaggregation.autoLoadPortletCapabilities` to `true`. For details of how to set this metadata, see *Changing the auto-loading of portlet capabilities*.

The Portal 8.5 theme in CF03 and newer has the auto-loading turned on by default. So, if you create your custom theme from the Portal 8.5 theme on a system at CF03 or beyond, you do not need to manually enable this feature. If your custom theme was created from the Portal 8.5 theme on a system before CF03, you need to manually enable this feature if you want it.

Usage of the auto-loading is recommended, because it makes the system overall simpler and easier to use. Users can use whichever portlets they want on a page (provided the portlets declare their needed capabilities) without having to think about the theme and which profile is assigned to the page. Fewer profiles are needed, and the size of each profile can be smaller.

If auto-loading is turned on, the non-self managed error handling messages appear less frequently, and if they do appear, they mean that the required capabilities are not installed or active anywhere in the system. The errors can be resolved by installing the capabilities and modules or making sure that they are active.

When auto-loading is turned on for a theme, it is not common to turn it off later. If you do, many pages can report missing capability error messages until appropriate profiles are assigned to the pages.

CF03

Deferred Capabilities

A portlet can now also declare a deferred capability dependency.

ID:

```
Preference Name: deferredCapability.<sequence-number>.id
Preference Value : the capability name
```

Value:

```
Preference Name: deferredCapability.<sequence-number>.minValue
Preference Value: the capability minimum version
```

Sample:

```
<portlet-preferences>
    <preference>
        <name>deferredCapability.1.id</name>
        <value>wcm_inplacEdit</value>
        <read-only>true</read-only>
    </preference>
</portlet-preferences>
```

With auto-loading on, a deferred capability does not load until the page is in edit mode or when the `i$.loadDeferred` API is used. With auto-loading off, a deferred capability declaration behaves like a capability declaration in that the loading of the capability's modules is done by the profile and not the portlet.

The `deferredCapability` sequence-number is independent of the `capability` sequence-number. For example, both can start at 1. And, like `capability`, `deferredCapability.`*`sequence-number`*`.minValue` is optional.

Change the auto-loading of portlet capabilities

You can change if portlets automatically load their dependent capabilities for a theme.

The Portal 8.5 theme in combined cumulative fix 03 and newer has the auto-loading turned on by default. So, if you create your custom theme from the Portal 8.5 theme on a system at CF03 or beyond, you do not need to manually enable this feature. If your custom theme was created from the Portal 8.5 theme on a system before CF03, you need to manually enable this feature if you want it.

Usage of the auto-loading is recommended, because it makes the system simpler and easier to use. Users can use whichever portlets they want on a page, provided the portlets declare their needed capabilities, without having to think about the theme and which profile is assigned to the page. Fewer profiles are needed, and the size of each profile can be smaller.

When auto-loading is turned on for a theme, it is not common to turn it off later. If you do, many pages can report missing capability error messages until appropriate profiles are assigned to the pages.

CF03 "Changing the auto-loading of portlet capabilities with WebDAV"
You can change the auto-loading of portlet capabilities with WebDAV

CF03 "Changing the auto-loading of portlet capabilities with XMLAccess"
You can change the auto-loading of portlet capabilities with XMLAccess

Changing the auto-loading of portlet capabilities with WebDAV:

You can change the auto-loading of portlet capabilities with WebDAV

Procedure

1. Connect your WebDAV client to `http://server:port/wps/mycontenthandler/dav/themelist/`.
2. Find the folder for your theme and copy the `metadata.properties` file to your local drive.
3. Edit the local copy of the file and set the auto-loading property to true or false.
   For example:
   ```
   resourceaggregation.autoLoadPortletCapabilities=true
   ```
4. Copy the local copy of the `metadata.properties` file back into the folder for your theme in the `themelist` folder.

Changing the auto-loading of portlet capabilities with XMLAccess:

You can change the auto-loading of portlet capabilities with XMLAccess

Procedure

1. Export the theme. You can export all themes that are defined for WebSphere Portal Express with the following script, or insert the specific theme object ID you want to export.
   ```
   <?xml version="1.0" encoding="UTF-8"?>
   <request
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd" type="export"
   create-oids="true">
       <portal action="locate">
          <theme action="export" objectid="*" />
       </portal>
   </request>
   ```
2. Modify the value of the **`resourceaggregation.autoLoadPortletCapabilities`** parameter to true or false, or add the parameter if it does not exist. For example:
   ```
   <parameter name="resourceaggregation.autoLoadPortletCapabilities" type="string" update="set"><
   ```
3. Import the XML file with the command line or Import XML.
4. Restart WebSphere Portal Express.

Response rendering for themes

To decrease the response time of your portal, a template parser resolves which modules are needed and collects all of the modules that are enabled by the current profile. Parts of the content are parsed and displayed as soon as possible.

When the template parser encounters a dynamic content spot that starts with the prefix `dyn-cs:`, it resolves it by locating and starting the Dynamic Content Spot Mapping DataSource. If the mapping definition from a logical name to a URI includes module information, the spot is rendered only if the module is defined for the current profile.

When the template parser encounters a dynamic content spot for a particular combiner, the combiner service decodes the URI and collects all of the modules that are enabled by the current profile. After all of the defined modules are collected, any modules are added that the defined modules require and that are not already included in the list.

The modules are then processed in order, and contributions to each type are aggregated into sets. All contributions of a particular type at the current extension point are collected into groups by their type.

Important: It is the responsibility of the contributor to ensure that the content of the contribution does not include malicious code. For example, config values that are read by a data source must be escaped before serializing it.

Responses are rendered in this order:

1. CSS
   - CSS is only valid in the head extension point.
   - The framework generates a link element with an `href` attribute whose value is URL that resolves to the combined results of all contributions to CSS at the head extension point.
   - When debug is enabled, separate link elements are generated for each contribution.
2. Static JavaScript configuration
   - The framework writes an external `<script>` tag with a `src` attribute whose value is a URL that resolves to the combined results of all contributions to static JavaScript configuration at the current extension point.
   - When debug is enabled, separate script elements are generated for each contribution.
3. Dynamic JavaScript configuration. The framework generates an inline script tag that loads the combined results of all contributions to dynamic JavaScript configuration at the current extension point.
4. JavaScript code
   - The framework generates an external script tag that loads the combined results of all contributions to JavaScript code at the current extension point.
   - When debug is enabled, separate script elements are generated for each contribution.
5. Markup
   - The framework directly writes the output from each markup contribution at the current extension point to the output stream.

- This framework is intended to be used for markup that always shows in the page when this module is enabled. Use markup for content that is not visual but is semantic.
- Do not use portal render request-dependent attributes, because there is no guarantee that those attributes are available in all cases. For example, when used in deferred mode, the render context is not available.

CF03

Updates for Fix Pack 03

If portlets are on the page and have dependencies that are defined through capabilities, the framework gathers all modules for the provided capabilities from all portlets and downloads those resources. Each is gathered with separate requests for better caching performance. You must set this in your theme.

If a module is already downloaded as part of the profile, the system does not download it again as part of the portlet requests. The order of the responses is identical as depicted previously, with the exception that there are potentially two requests per type.

Profile CSS

Contains all resources from modules within the profile that contribute CSS. This request is skipped if there is no CSS contribution from any module.

Portlet CSS

Contains all resources from modules that are gathered from portlet capabilities that contribute CSS. This request is skipped if there is no CSS contribution from any module.

Profile `Static JavaScript configuration`

Contains all resources from modules within the profile that contribute `Static JavaScript configuration`. This request is skipped if there is no `Static JavaScript configuration` contribution from any module.

Portlet `Static JavaScript configuration`

Contains all resources from modules that are gathered from portlet capabilities that contribute `Static JavaScript configuration`. This request is skipped if there is no `Static JavaScript configurations` contribution from any module.

Profile `Dynamic JavaScript configuration`

Contains all resources from modules within the profile that contribute `Dynamic JavaScript configuration`. This type is written inline into the page and doesn't generate an extra request.

Portlet `Dynamic JavaScript configuration`

Contains all resources from modules that are gathered from portlet capabilities that contribute `Dynamic JavaScript configuration`. This type is written inline into the page and doesn't generate an extra request.

Profile JavaScript

Contains all resources from modules within the profile that contribute JavaScript. This request is skipped if there is no JavaScript contribution from any module.

Portlet JavaScript

Contains all resources from modules that are gathered from portlet capabilities that contribute JavaScript. This request is skipped if there is no JavaScript contribution from any module.

Profile Markup

Contains all resources from modules within the profile that contribute Markup. This request is skipped if there is no Markup contribution from any module.

Portlet Markup

Contains all resources from modules that are gathered from portlet capabilities that contribute Markup. This request is skipped if there is no Markup contribution from any module.

Writing modules

You can define global or theme-specific contributions that contain a theme module, which applies to different scopes through theme profiles. A module defines its contributions by using a `plugin.xml` or JSON file.

System modules

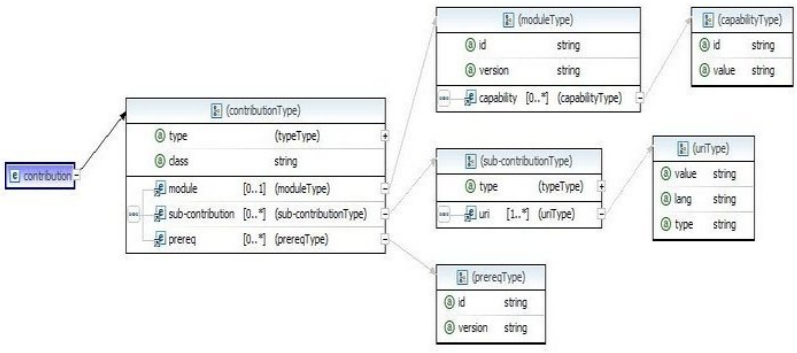
Available to all themes in the system by using a `plugin.xml` file as part of a compressed file in the portal class path, web module, or as a single file in the `WEB-INF` directory of a web module. The theme module must be declared within its extension point: `com.ibm.portal.resourceaggregator.module`.

Theme-specific modules

Can be referenced only by the theme that defines them in a JSON file within its contributions folder. The contributions folder is the default folder, but the location can be changed by setting the `resourceaggregation.contributions.folder` metadata on the theme, as in the case of profiles. These files are automatically scanned by the system and the contributions that are defined are registered for the theme.

XML schema

The following figure depicts the XML schema for this extension point:



The following code is an example of a `plugin.xml` file where the module is deployed within a web application with the context root `res:{war:context-root}/`.

```
<extension point="com.ibm.portal.resourceaggregator.module" id="testModuleExtension1">
       <module id="testModule1" version="1.0">

           <capability id="capabilityA" value="1.0.0"/>
           <capability id="capabilityB" value="1.5"/>

           <prereq id="testModuleA"/>
           <prereq id="testModuleB" minVersion="1.2.3.4"/>
           <prereq id="testModuleC" type="optional"/>
```

```
        <title lang="en" value="en Module"/>
        <title lang="de" value="de Module"/>
        <title lang="es" value="es Module"/>

        <description lang="en" value="one two three"/>
        <description lang="de" value="ein zwei drei"/>
        <description lang="es" value="uno dos tres"/>

        <contribution type="head">
            <sub-contribution type="css">
                <uri value="res:{war:context-root}/css/helloWorld.css" />
                <!-- define alternate styles for right to left -->
                <uri type="rtl" value="res:{war:context-root}/css/helloWorld_rtl.css" />
                <!-- define alternate styles for an iPad -->
                <uri deviceClass="tablet+iOS" value="res:{war:context-root}/css/helloWorld_iPad.css" />
            </sub-contribution>
            <sub-contribution type="js">
                <uri value="res:{war:context-root}/js/helloWorldHead.js" />
            </sub-contribution>
            <sub-contribution type="markup">
                <uri value="res:{war:context-root}/markup/helloWorldHead.html" />
            </sub-contribution>
        </contribution>

        <contribution type="config">
            <sub-contribution type="js">
                <uri value="res:{war:context-root}/js/helloWorldBody_root.js" />
                <!-- define alternate js for when the Portal is using different languages -->
                <uri lang="en" value="res:{war:context-root}/js/helloWorldBody_en.js" />
                <uri lang="de" value="res:{war:context-root}/js/helloWorldBody_de.js" />
                <uri lang="es" value="res:{war:context-root}/js/helloWorldBody_es.js" />
                <!-- define alternate js for debugging purposes in LTR and RTL environments -->
                <uri type="debug" value="res:{war:context-root}/js/helloWorldBody_debug.js" />
                <uri type="debug,rtl" value="res:{war:context-root}/js/helloWorldBody_debug_rtl.js" />
            </sub-contribution>
            <sub-contribution type="config_dynamic">
                <uri value="res:{war:context-root}/jsp/helloWorldBodyConfig.jsp" />
            </sub-contribution>
            <sub-contribution type="config_static">
                <uri value="res:{war:context-root}/jsp/helloWorldBodyStatic.jsp" />
            </sub-contribution>
        </contribution>

        <contribution type="menu">
            <sub-contribution type="json">
                <uri value="res:{war:context-root}/js/helloWorld.json" />
            </sub-contribution>
        </contribution>

        <contribution type="dyn-cs">
            <sub-contribution type="markup" ref-id="some_dynamic_spot_id">
                <uri value="res:{war:context-root}/jsp/helloWorldDynamicSpot.jsp" />
            </sub-contribution>
        </contribution>

        <moduleActivation extensionID="com.ibm.portal.resourceaggregator.util.ResourceEnvironmentProviderModuleActivationHandler"
            <parameter name="rep" value="RESOURCE_ENV_PROVIDER_NAME" />
            <parameter name="key" value="KEY_IN_RESOURCE_ENV_PROVIDER"/>
        </moduleActivation>

        <runtimeActivation>
            <condition deviceClass="tablet"/>
        </runtimeActivation>

    </module>
</extension>

<extension point="com.ibm.portal.resourceaggregator.module" id="testModuleExtension2">
    <module id="testModule2" version="1.0">
        <!-- Some other module... -->
    </module>
</extension>

<extension point="com.ibm.portal.resourceaggregator.module" id="testModuleExtension3">
    <module id="testModule3" version="1.0">
        <!-- One last module... -->
    </module>
</extension>
```

The following code is an example of a `myModules.json` file that is stored in the contributions folder of a theme. The structure, keys, and values that are used in the `plugin.xml` file are adapted to a JSON format with these rules:

- Keys that allow for multiple child entries on the same level are used in the plural form. For example, prereq becomes prereqs and capability becomes capabilities.

- File paths must start with a forward slash and are resolved relative to the theme root folder.
- JSP files cannot be served out of WebDAV. All resources relative to the theme root must be of a static type, for example, js, CSS, or HTML.

```
{
        "modules":[{
            "id":"testModule1",
            "version":"1.0",

            "capabilities":[{
                    "id":"capabilityA",
                    "value":"1.0.0"
                },
                {
                    "id":"capabilityB",
                    "value":"1.5"
                }
            ],

            "prereqs":[{
                    "id":"testModuleA"
                },
                {
                    "id":"testModuleB",
                    "minVersion":"1.2.3.4"
                },
                {
                    "id":"testModuleC",
                    "type":"optional"
                }
            ],

            "titles":[{
                    "lang":"en",
                    "value":"en Module"
                },
                {
                    "lang":"de",
                    "value":"de Module"
                },
                {
                    "lang":"es",
                    "value":"es Module"
                }
            ],

            "descriptions":[{
                    "lang":"en",
                    "value":"one two three"
                },
                {
                    "lang":"de",
                    "value":"ein zwei drei"
                },
                {
                    "lang":"es",
                    "value":"uno dos tres"
                }
            ],

            "contributions":[{
                "type":"head",
                "sub-contributions":[{
                    "type":"css",
                    "uris":[{
                        "value":"/css/helloWorld.css"
```

```
            },
            // define alternate styles for right to left
            {
                "value":"/css/helloWorld_rtl.css",
                "type":"rtl"
            },
            // define alternate styles for an iPad
            {
                "value":"/css/helloWorld_iPad.css",
                "deviceClass":"tablet+iOS"
            }]
        },
        {
            "type":"js",
            "uris":[{
                "value":"/js/helloWorldHead.js"
            }]
        },
        {
            "type":"markup",
            "uris":[{
                "value":"/markup/helloWorldHead.html"
            }]
        }]
    },{
        "type":"config",
        "sub-contributions":[{
            "type":"js",
            "uris":[{
                "value":"/js/helloWorldBody_root.js"
            },
            // define alternate js for when the Portal is using different languages
            {
                "value":"/js/helloWorldBody_en.js",
                "lang":"en"
            },
            {
                "value":"/js/helloWorldBody_de.js",
                "lang":"de"
            },
            {
                "value":"/js/helloWorldBody_es.js",
                "lang":"es"
            },
            // define alternate js for debugging purposes in LTR and RTL environments
            {
                "value":"/js/helloWorldBody_debug.js",
                "type":"debug"
            },
            {
                "value":"/js/helloWorldBody_debug_rtl.js",
                "type":"debug,rtl"
            }]
        },
        {
            "type":"config_dynamic",
            "uris":[{
                "value":"/config/helloWorldBodyConfig.js"
            }]
        },
        {
            "type":"config_static",
            "uris":[{
                "value":"/config/helloWorldBodyStatic.js"
            }]
        }]
    },{
```

```
            "type":"menu",
            "sub-contribution":[{
                "type":"json",
                "uris":[{
                    "value":"/js/helloWorld.json"
                }]
            }]
        },{
            "type":"dyn-cs",
             "sub-contribution":[{
                "type":"markup",
                "ref-id":"some_dynamic_spot_id",
                "uris":[{
                    "value":"/html/helloWorldDynamicSpot.html"
                }]
            }]
        }],

        "moduleActivation":{
            "extensionID":"com.ibm.portal.resourceaggregator.util.ResourceEnvironmentProviderModu
            "parameters":[{
                "name":"rep",
                "value":"RESOURCE_ENV_PROVIDER_NAME"
            },
            {
                "name":"key",
                "value":"KEY_IN_RESOURCE_ENV_PROVIDER"
            }]
        },

        "runtimeActivation":[{
            "condition":{
                "deviceClass":"tablet"
            }
        }]

    },{
        "id":"testModule2",
        "version":"1.0"
        // Some other module...
    },{
        "id":"testModule3",
        "version":"1.0"
        // One last module...
    }]
}
```

Contribution URIs for Web Modules

{war:context-root}

This variable fetches the context root of the containing WAR file and inserts it in place. It only works for modules defined as part of a WAR file. This does not work for modules defined in WebDAV or somewhere else in the classloading hierarchy, for example, a shared application.

"Defining theme modules" on page 2547
You can define theme modules in XML or JSON.

"Module schema definition" on page 2554
You can use these elements to define theme modules.

"Profile schema definition" on page 2558
You can write a profile schema with valid JSON.

"Simple modules" on page 2559
Simple modules for the resource aggregator framework are provided in the WebDAV folder. You can define modules quickly with a limited set of features with these simple modules.

"Dynamically extending an existing menu item from a module" on page 2565
You can use a module to add menu items to a menu where the menu item displays only on certain pages,

Defining theme modules:

You can define theme modules in XML or JSON.

About this task

The following samples are a breakdown of the previous examples and cover all available syntax for the `plugin.xml` extension point and JSON module definitions.

Procedure

1. Define the module. An ID is required and version is an optional decimal value.

```
<module id="testModule1" version="1.0" >
```

```
       "modules":[
        // ...  The ellipses indicate place holders for syntax explained in other steps
        {
            "id":"testModule1",
            "version":"1.0",
            // ...
        },
        // ...
        ]
```

2. Optional: Define any number of capabilities in the module with a required ID string and value in dot notation, for example, `1.2.3.4`. This information is added to the overall theme client capabilities map that is carried in the `com.ibm.portal.theme.client.capabilities` request attribute. Portlets can query from the server-side, which client-side capabilities are present, such as JavaScript libraries and CSS style catalogs. Client-side JavaScript code can query the global JSON object `com_ibm_theme_capabilities` for all available capabilities within the scope of the currently rendered page.

```
<capability id="capabilityA" value="1.0.0"/>
       <capability id="capabilityB" value="1.5"/>
```

```
"capabilities":[{
                "id":"capabilityA",
                "value":"1.0.0"
            },
            {
                "id":"capabilityB",
                "value":"1.5"
            }
        ],
```

CF03 In addition to the explicit capabilities defined here, an implicitly defined capability per module was also introduced. This implicitly defined capability has the name and version of the module. If the same name is defined explicitly, the explicit one is used.

3. Optional: Define any number of prereqs inside the module, with a required ID string and optional type string or minVersion in dot notation. If type `optional` is used, no errors are output if the prereq cannot be found on the system.

```
<prereq id="testModuleA"/>
        <prereq id="testModuleB" minVersion="1.2.3.4"/>
        <prereq id="testModuleC" type="optional"/>
```

```
"prereqs":[{
                "id":"testModuleA"
            },
            {
                "id":"testModuleB",
                "minVersion":"1.2.3.4"
            },
            {
                "id":"testModuleC",
                "type":"optional"
            }
        ],
```

4. Optional: Add a title or titles for the module in different languages.

```
<title lang="en" value="en Module"/>
        <title lang="de" value="de Module"/>
        <title lang="es" value="es Module"/>
```

```
 "titles":[{
                "lang":"en",
                "value":"en Module"
            },
            {
                "lang":"de",
                "value":"de Module"
            },
            {
                "lang":"es",
                "value":"es Module"
            }
        ],
```

5. Optional: Add a description or descriptions in different languages.

```
<description lang="en" value="one two three"/>
        <description lang="de" value="ein zwei drei"/>
        <description lang="es" value="uno dos tres"/>
```

```
    "descriptions":[{
            "lang":"en",
            "value":"one two three"
        },
        {
            "lang":"de",
            "value":"ein zwei drei"
        },
        {
            "lang":"es",
            "value":"uno dos tres"
        }
    ],
```

6. Require at least one: Add contributions each with at least one child subcontribution to the module. A contribution must be one of four types, which determines where its subcontributions are returned to the HTML page:

   head These contributions are linked into the head section of the theme at the `co:head` dynamic content spot.

   config These contributions are added to the end of the body section in the theme at the co:config dynamic content spot.

   menu These contributions are available to be called by the theme menu framework.

dyn-cs The outputs of these contributions are added to the theme at the specified dynamic content spot.

See *Working with dynamic content spots*. Each contribution has at least one subcontribution. Each subcontribution has at least one URI to a resource, and exactly one of the URIs is returned from each subcontribution. Subcontributions each has a required type:

css Cascading style sheet files can be added to head contributions only.

js JavaScript files can be linked to head or config contributions.

json JavaScript Object Notation output is used by menu contributions only.

markup
HTML can be served by head, config, or dyn-cs dynamic spots.

config_static
This type is a JavaScript configuration object that is static, global and publicly cacheable, made available in head or config contributions.

config_dynamic
This type is a JavaScript configuration object that can change based on the current page or user, made available in head or config contributions.

See Contribution types. Given these contribution and subcontribution types, one can implement the following use cases in a module. See the sample code at the beginning of this topic to see all the following snippets put together.

a. Optional: Add a CSS file. If you add more than one CSS file of the same type, use a separate subcontribution for each one.

```
<contribution type="head">
                <sub-contribution type="css">
                    <uri value="res:/HelloWorld/css/helloWorld.css" />
                </sub-contribution>
            </contribution>
```

```
"contributions":[{
                "type":"head",
                "sub-contributions":[{
                    "type":"css",
                    "uris":[{
                        "value":"/css/helloWorld.css"
                    }
                }
            }],
```

b. Optional: Define an alternative CSS file for bidirectional styles. These styles are only served up when using a `rtl` language.

```
<contribution type="head">
                <sub-contribution type="css">
                    <uri value="res:/HelloWorld/css/helloWorld.css" />
                    <!-- define alternate styles for right to left -->
                    <uri type="rtl" value="res:/HelloWorld/css/helloWorld_rtl.css" />
                </sub-contribution>
            </contribution>
```

```
"contributions":[{
                "type":"head",
                "sub-contributions":[{
                    "type":"css",
                    "uris":[{
                        "value":"/css/helloWorld.css"
                    },
                    // define alternate styles for right to left
                    {
                        "value":"/css/helloWorld_rtl.css",
```

```
                    "type":"rtl"
                }]
            }
        }],
```

c. Optional: Define a resource for a specific device class. Within the same subcontribution, add URIs with the appropriate device class string or equation identifier. See *Device class equations*.

```
<contribution type="head">
            <sub-contribution type="css">
                <uri value="res:/HelloWorld/css/helloWorld.css" />
                <!-- define alternate styles for an iPad -->
                <uri deviceClass="tablet+iOS" value="res:/HelloWorld/css/helloWorld_iPad.c
            </sub-contribution>
        </contribution>
```

```
"contributions":[{
            "type":"head",
            "sub-contributions":[{
                "type":"css",
                "uris":[{
                    "value":"/css/helloWorld.css"
                },
                // define alternate styles for an iPad
                {
                    "value":"/css/helloWorld_iPad.css",
                    "deviceClass":"tablet+iOS"
                }]
            }
        }],
```

d. Optional: Add a piece of markup to the head section, ensuring that the HTML validates within the head, such as a meta tag. Subcontributions of type markup can also be added to config or dyn-cs contributions.

```
<contribution type="head">
            <sub-contribution type="markup">
                <uri value="res:/HelloWorld/markup/helloWorldHead.html" />
            </sub-contribution>
        </contribution>
```

```
"contributions":[{
            "type":"head",
            "sub-contributions":[{
                "type":"markup",
                "uris":[{
                    "value":"/markup/helloWorldHead.html"
                }]
            }]
        }],
```

e. Optional: Add a JavaScript file to the head section of the theme because it must be available for portlets to use, for example, a js library such as Dojo or jQuery). If you add more than one JavaScript file of the same type, use a separate subcontribution for each one.

```
<contribution type="head">
            <sub-contribution type="js">
                <uri value="res:/HelloWorld/js/helloWorldHead.js" />
            </sub-contribution>
        </contribution>
```

```
"contributions":[{
            "type":"head",
            "sub-contributions":[{
                "type":"js",
                "uris":[{
```

```
                    "value":"/js/helloWorldHead.js"
                }]
            }]
        }]
```

f. Optional: Define a JavaScript file in the config area. It performs better than adding it to the head, but the js is not available until after the portlets have loaded. If you add more than one JavaScript file of the same type, use a separate subcontribution for each one.

```
<contribution type="config">
            <sub-contribution type="js">
                <uri value="res:/HelloWorld/js/helloWorldBody_root.js" />
            </sub-contribution>
        </contribution>
```

```
"contributions":[{
            "type":"config",
            "sub-contributions":[{
                "type":"js",
                "uris":[{
                    "value":"/js/helloWorldBody_root.js"
                }]
            }]
        }]
```

g. Optional: Define an alternative JavaScript file for other locales. Within the same subcontribution, add a second URI with the appropriate locale attribute.

```
<contribution type="config">
            <sub-contribution type="js">
                <uri value="res:/HelloWorld/js/helloWorldBody_root.js" />
                <!-- define alternate js for when the Portal is using different languag
                <uri lang="en" value="res:/HelloWorld/js/helloWorldBody_en.js" />
                <uri lang="de" value="res:/HelloWorld/js/helloWorldBody_de.js" />
                <uri lang="es" value="res:/HelloWorld/js/helloWorldBody_es.js" />
            </sub-contribution>
        </contribution>
```

```
"contributions":[{
            "type":"config",
            "sub-contributions":[{
                "type":"js",
                "uris":[{
                    "value":"/js/helloWorldBody_root.js"
                },
                // define alternate js for when the Portal is using different languages
                {
                    "value":"/js/helloWorldBody_en.js",
                    "lang":"en"
                },
                {
                    "value":"/js/helloWorldBody_de.js",
                    "lang":"de"
                },
                {
                    "value":"/js/helloWorldBody_es.js",
                    "lang":"es"
                }]
            }]
        }],
```

h. Optional: Define an alternative JavaScript file for debugging purposes. To debug the client-side code of a theme, supply a second entry for a debug subcontribution type debug to provide a more readable version of the JavaScript file. For more information, see *Enabling module tracing*.

```
<contribution type="config">
                <sub-contribution type="js">
                    <uri value="res:/HelloWorld/js/helloWorldBody_root.js" />
                    <!-- define alternate js for debugging purposes in LTR and RTL environmen
                    <uri type="debug" value="res:/HelloWorld/js/helloWorldBody_debug.js" />
                    <uri type="debug,rtl" value="res:/HelloWorld/js/helloWorldBody_debug_rtl.
                </sub-contribution>
            </contribution>
```

```
"contributions":[{
                "type":"config",
                "sub-contributions":[{
                    "type":"js",
                    "uris":[{
                        "value":"/js/helloWorldBody_root.js"
                    },
                    // define alternate js for debugging purposes in LTR and RTL environments
                    {
                        "value":"/js/helloWorldBody_debug.js",
                        "type":"debug"
                    },
                    {
                        "value":"/js/helloWorldBody_debug_rtl.js",
                        "type":"debug,rtl"
                    }]
                }]
            }],
```

i. Optional: Add a dynamic JavaScript config object.

```
<contribution type="config">
                <sub-contribution type="config_dynamic">
                    <uri value="res:/HelloWorld/jsp/helloWorldBodyConfig.jsp" />
                </sub-contribution>
            </contribution>
```

```
"contributions":[{
                "type":"config",
                "sub-contributions":[
                {
                    "type":"config_dynamic",
                    "uris":[{
                        "value":"/config/helloWorldBodyConfig.js"
                    }]
                }
            }]
```

j. Optional: Add a static JavaScript config object.

```
<contribution type="config">
                <sub-contribution type="config_static">
                    <uri value="res:/HelloWorld/jsp/helloWorldBodyStatic.jsp" />
                </sub-contribution>
            </contribution>
```

```
"contributions":[{
                "type":"config",
                "sub-contributions":[
                {
                    "type":"config_static",
                    "uris":[{
                        "value":"/config/helloWorldBodyStatic.js"
                    }]
                }]
            }]
```

k. Optional: Define a menu contribution. Create a contribution of type menu with a subcontribution of type json. The subcontribution references a JSON file that defines the individual menu entries. See *Menu framework*.

```
<contribution type="menu">
                <sub-contribution type="json">
                    <uri value="res:/HelloWorld/js/helloWorld.json" />
                </sub-contribution>
            </contribution>
```

```
"contributions":[{
                "type":"menu",
                "sub-contribution":[{
                    "type":"json",
                    "uris":[{
                        "value":"/js/helloWorld.json"
                    }]
                }]
            }]
```

l. Optional: Define a dynamic spot. When the module is turned on, the output of this subcontribution replaces the default contents of the dynamic spot that is identified by the ref-id attribute. See *Dynamic contents spots*.

```
<contribution type="dyn-cs">
                <sub-contribution type="markup" ref-id="some_dynamic_spot_id">
                    <uri value="res:/HelloWorld/jsp/helloWorldDynamicSpot.jsp" />
                </sub-contribution>
            </contribution>
```

```
"contributions":[{
                "type":"dyn-cs",
                 "sub-contribution":[{
                    "type":"markup",
                    "ref-id":"some_dynamic_spot_id",
                    "uris":[{
                        "value":"/html/helloWorldDynamicSpot.html"
                    }]
                }]
            }]
```

7. Optional: Give the module an activation handler. Use either the extensionID, used in these examples, or class attribute to indicate the ModuleActiveChecker implementation; both are supported. By default, all modules that are defined are active. An inactive module is treated the same as a module that is not defined. Therefore, inactive modules are not started during portal run time. Specify a key for activation or deactivation of the module with an entry in a resource environment provider in the application server. See *Adding resource environment provider properties*. Substitute the `RESOURCE_ENV_PROVIDER_NAME` with the name of the resource environment provider, and *KEY_IN_RESOURCE_ENV_PROVIDER* with the key within the resource environment provider. Valid key values are true if the module is active, or false if the module is inactive. For example, if you want to specify the `my.module.is.active` key for module activation in the resource environment provider ConfigService, the entry is:

```
<repentry rep="ConfigService" key="my.module.is.active"/>
```

```
<moduleActivation extensionID="com.ibm.portal.resourceaggregator.util.ResourceEnvironmentProvi
            <parameter name="rep" value="RESOURCE_ENV_PROVIDER_NAME" />
            <parameter name="key" value="KEY_IN_RESOURCE_ENV_PROVIDER"/>
        </moduleActivation>
```

```
"moduleActivation":{
            "extensionID":"com.ibm.portal.resourceaggregator.util.ResourceEnvironmentProviderM
            "parameters":[{
                "name":"rep",
                "value":"RESOURCE_ENV_PROVIDER_NAME"
            },
            {
```

```
            "name":"key",
            "value":"KEY_IN_RESOURCE_ENV_PROVIDER"
        }]
    },
```

8. Optional: Give the module a runtime activation handler with a condition. This allows a module to be turned on or off on a per page basis, which is based on the state of the current page. Currently, the runtime activation handler supports checking device class, which can be a string or an equation. See Device class equations.

```
<runtimeActivation>
            <condition deviceClass="tablet"/>
        </runtimeActivation>{code}
```

```
"runtimeActivation":[{
            "condition":{
                "deviceClass":"tablet"
            }
        }]
```

Module schema definition:

You can use these elements to define theme modules.

Defining a module

You can also define a schema for modules that are independent of their XML or JSON format. For the JSON format, there is a JSON schema definition available at *WebDAV Theme*\themes\Portal8.5\contributions\schema.

In the following examples, use the following syntax.

- 1 means only one is required
- 0..1 means one or zero
- 0..* means that any number is allowed
- 1..* means at least one is required

Element: module

The module that provides contributions.

Attributes:

- ID (1): The system-wide identifier for this module. The ID value is a String that must be unique in the system.
- version (0..1): The version of the module that is provided with this contribution. The version value uses the format major.minor.revision.

Parent

- extension

Child-elements

- 0..* capability
- 0..* prereq
- 0..* title
- 0..* description
- 0..* contribution
- 0..1 moduleActivation
- 0..1 runtimeActivation

Element: capability

The declaration of a capability that this module delivers.

Attributes:

- ID (1): The unique name to be used for this capability.
- value (1): the version of the capability that is provided by this module. The capability value uses a dotted decimal string.

Parent

- module

Element: prereq

A module that is required by another module that defines the requirement.

Attributes:

- ID (1): The ID of the required module.
- minversion (0..1): The minimum version of the prereq module that is required by the module that defines the requirement. The minversion value uses a dotted decimal string.
- type (0..1): Setting the `type="optional"` stops error logging when there is a missing prereq.

Parent

- module

Element: title

A title for the defining module.

Attributes:

- value (1): A title for the module.
- lang (1): The language string for the title, for example, en_US, de_de, or es_es.

Parent

- module

Element: description

A description of the defining module.

Attributes:

- value (1): A string that describes the module.
- lang (1): The language string for the description, for example, en_us, de_de, or es_es.

Parent

- module

Element: contribution

Defines a theme contribution of the module.

Attributes:

- type (1): The type of content spot that this module contributes to. The value is either head, config, menu, or dyn-cs. The valid values for type are:

  head These contributions are added to the head section of the theme. This contribution type can have multiple subcontributions.

config These contributions are added to the end of the body section of the theme. This contribution type can have multiple subcontributions.

dyn-cs This contribution outputs a theme dynamic spot. See *Dynamic content spots*. This contribution type can have one subcontribution.

menu This contribution defines a JSON theme menu. See *Menu framework*. This contribution type can have one subcontribution.

Parent

- module

Child-elements

- 0..* subcontribution

Element: subcontribution

Part of the overall contribution. For example, in the head contribution, subcontributions can add CSS, JavaScript, or head-specific markup.

Attributes

- uri (0..1 or 1..* if there is no URI child): The URI to the subcontribution resource. The URI can also be provided as a child element of the subcontribution node. Use an attribute, or a child, but not both to specify the URI. More than one URI can be supplied, but only one is selected and returned.
- ref-id (0..1): The ID of the dynamic spot to override. Use it only for dyn-cs subcontributions.
- type (1): Declares what type of contribution is provided. The valid values for type are:
  - markup: HTML markup for head, config, or dyn-cs contributions.
  - css: Cascading style sheet files for head contributions only.
  - js: JavaScript libraries, functions, and objects for head and config contributions. These contributions are loaded after contributions with a config_* value.
  - config_static: JavaScript configuration for head and config contributions that are not request-dependent variable definitions. These contributions are collected in a separate file.
  - config_dynamic: JavaScript configuration for head and config contributions that are request-dependent variable definitions. These contributions are injected into the markup.
  - json: JSON definitions for menu contributions only.
- deviceClass (0..1): Controls when the defined subcontribution is aggregated on the page that is based on active device classes. It can be a single device class or logical equation.
- values:
  - markup: HTML markup
  - css: Cascading style sheet statements.
  - js: JavaScript libraries, functions, and objects. These contributions are loaded after contributions with a config_* value.
  - config_static: JavaScript configuration contributions that are not request-dependent variable definitions. These contributions are collected in a separate file.

– config_dynamic: JavaScript configuration contributions that are request-dependent variable definitions. These contributions are injected into the markup.

Parent

- contribution

Child-elements

- 0..1 uri

Element: uri

A URI pointing to a theme module resource. `{war:context-root}` can be used here. For more information, see *Writing modules*.

Attributes:

- value (1): The URI of the resource to be used.
- lang (0..1): The language to which the defined URI applies, for example, en_US.
- type (0..1): The type to which this URI applies. For the subcontribution type of CSS the URI type can be `rtl`, triggered when the Portal is rendering in a right to left language. Or `debug`, triggered when the Portal is in debug mode. For JavaScript, the URI type can be `debug`.
- deviceClass (0..1): The type of device to which the URI applies, such as `smartphone`, `tablet`, or a device class equation.

Parent

- subcontribution

Different schemes can be used within the value attribute.

Use this scheme with caution because it can severely affect performance. The remote file is included on the server and not on the browser. When this scheme is used, remote requests are sent from the virtual machine of the portal during portal page rendering to the remote system in a synchronous manner. Therefore, the response time of the remote system affects portal page rendering time and the number of available threads available for other requests. Use HTTP only for static sources that can be cached in a proxy or HTTP server in front of portal.

The Ajax proxy is used to perform the remote call so that the configuration must be updated for the remote request to occur.

- http: Use HTTP to reference files that are accessible using HTTP.
- res: Use the res scheme to access an arbitrary resource, such as a JSP, a servlet, or a static file, in a web module that is installed on the server.
- dav: Use the dav scheme to access resources in the portal file store. You can also use replacement tokens for the URI value, for example:

```
res:{rep=WP
CommonComponentConfigService;key=cc.theme.context}/themes/html/PageBuilder2/modules/css/markup/head_m.jsp
```

The code

```
{rep=WP CommonComponentConfigService;key=cc.theme.context}
```

is replaced with the value of the custom property **cc.theme.context** in the resource environment provider **WP CommonComponentConfigService**.

Element: moduleActivation

Declaration of a class that controls whether the module is active. Use either the extensionID or class attribute to indicate the activation implementation; both are supported.

Attributes:

- extensionID (0..1 or 0 if class is used): The class name that implements the ModuleActiveChecker interface.
- class (0..1): The class name that implements the ModuleActiveChecker interface.

Parent

- module

Child-elements

- 0..* parameters

Element: parameter

Declaration of a resource environment entry that specifies whether the module is active or not whose values are `true` or `false`. The default value is `true`.

Attributes:

- name (1): The name of the parameter.
- value (1): The value of the parameter.

Parent

- moduleActivation

Element: runtimeActivation

A switch that is triggered during run time to enable or disable a module and its capabilities, depending on a condition of the current page state.

Attributes:

- none

Parent

- module

Child-elements:

- condition (1..*)

Example:

```
<runtimeActivation>
         <condition deviceClass="worklight"/>
</runtimeActivation>"
```

Element: condition

A condition for which the runtimeActivation must be activated.

Attributes:

- deviceClass (1): The type of device to which the URI applies, such as `smartphone`, `tablet`, or a device class equation.

Parent

- runtimeActivation

Profile schema definition:

You can write a profile schema with valid JSON.

Components of a profile file

The JSON schema definition is available in *WebDAV Theme*/themes/Portal8.5/profiles/schema.

The profile definition has five main components.

non-deferred modules
The set of modules that load with the initial page rendering.

deferred modules
The set of modules that render after the initial page rendering.

title (optional)
The title of the profile.

description (optional)
The description of the profile.

metadata (optional)
Here you can define whether this profile is hidden or not.

You can see the properties in the following example.

```
{
 "moduleIDs" : ["moduleID_1", "moduleID_2", "moduleID_3"],
 "deferredModuleIDs" : ["moduleID_4", "moduleID_5", "moduleID_6"],
 "titles": [{ "lang": "en", "value": "title_en" },
              { "lang": "de", "value": "title_de" }],
 "descriptions": [{ "lang": "en", "value": "desc_en" },
                    { "lang": "de", "value": "desc_de" }],
    "metadata":{
 "com.ibm.portal.Hidden": "true"
 }
}
```

Simple modules:

Simple modules for the resource aggregator framework are provided in the WebDAV folder. You can define modules quickly with a limited set of features with these simple modules.

If you need all features of the framework, you must define your modules through a plugin.xml or JSON file in the contributions folder.

Module quick start

Each subdirectory within the modules folder defines one module.

```
WebDAV Root
+ modules
+-- module A
+-- module B
+-- ...
+-- module Z
```

The getting_started_module is pre-defined so you can start quickly. Add your JavaScript, CSS, or markup file to one of the subfolders, and your resources are integrated into WebSphere Portal Express.

You must invalidate the resource aggregator cache before your changes are integrated. Click the **Administration menu** icon. Then, click **Portal Analysis** > **Theme Analyzer**. Then, click **Utilities** > **Control Center** > **Invalidate cache** to

invalidate the cache. CF06 Auto invalidation recognizes your changes automatically for WebDAV based themes. No further action is required. For more information, see Utilities.

Module directory

After you create your own module by creating a new directory in the modules directory, you must make sure that the module is integrated with WebSphere Portal Express. Add it to the profile currently set on your page. To verify the profile for any page, you can use the Theme Analyzer Page Explorer. Click the **Administration menu** icon. Then, click **Portal Analysis** > **Theme Analyzer**. Then, click **Examine page profile information**.

Simple contributions can be of three contribution types, `head`, `config`, `menu`.

For more information, on simple modules read the `readme.txt` file in the `getting_started_module` folder.

Simple module directory structure

Simple modules can be customized with the following files and directories.

/module-id
: Each directory in `/modules` defines a new module, but does not support versions.

  /module-id/**prereqs.properties**
  : Optional file that defines the `prereqs` of this module. In the file, there is one `module-id` per new line. Module versions are not supported. For example,

    ```
    module-id
    module-id
    ```

  /module-id/**capabilities.properties**
  : Optional file that defines the capabilities of this module. In the file, there is a property such as **style** with the `name` and `value` that is separated by an equal sign. Add one per line.

    ```
    name=value
    name=value
    ```

  /module-id/**localization.properties**
  : Optional file that defines title and description of this module. In the file, there is a property such as **style**. For example,

    ```
    title.locale=title
    description.locale=description
    ```

    Replace *locale* and with the local representing code for the location. For example, for the United States, use `en_us`. Replace *title* with the title of the module.

  /module-id/**head**
  : For more information, see *Head contribution*.

  /module-id/**config**
  : For more information, see *Config contribution*.

  /module-id/**menu**
  : Files that are stored in this directory are made available to the menu framework.

/*module-id*/menu/*.json
JSON resources that contain the menu definition are served in alphabetical order.

"Head contribution"
The `head` folder contains files that are served as head contribution through the resource aggregator framework. Those resources appear in the head tag of the markup served to the browser.

"Config contribution" on page 2563
The `config` folder contains files that are served as config contribution through the resource aggregator framework. Those resources are usually before the closing body tag.

Head contribution:

The `head` folder contains files that are served as head contribution through the resource aggregator framework. Those resources appear in the head tag of the markup served to the browser.

Folder: /*module-id*/head

The following section provides a complete list of supported files within the head section.

/*module-id*/head
Files that are stored in this directory are served in the head section.

/head/*.js
JavaScript files are served in alphabetical order. They are grouped by file name and if they have the same name they belong to the same group. Within this group, the following extension variations exist. There are two sets of six resources. The sets cannot be mixed.

***.js** Defines the main compressed JavaScript content.

***.js.uncompressed.js**
Defines the main debug JavaScript content.

***.rtl.js**
Defines the compressed JavaScript to be used for right-to-left languages.

***.rtl.js.uncompressed.js**
Defines the debug JavaScript to be used for right-to-left languages.

***.*locale*.js**
Defines the compressed JavaScript content for a specific language. Replace *locale* and with the local representing code for the location. For example, for the United States, use `en_us`.

***.*locale*.js.uncompressed.js**
Defines the debug JavaScript content for a specific language. Replace *locale* and with the local representing code for the location. For example, for the United States, use `en_us`.

The following group is an alternative. You can use either of these variation group, but you cannot mix them.

Alternative

The following group is an alternative. You can use either of these variation group, but you cannot mix them.

***.min.js**
Defines the main compressed JavaScript content.

***.js** Defines the main debug JavaScript content.

***.rtl.min.js**
Defines the compressed JavaScript to be used for right-to-left languages.

***.rtl.js**
Defines the debug JavaScript to be used for right-to-left languages.

***.*locale*.min.js**
Defines the compressed JavaScript content for a specific language. Replace *locale* and with the local representing code for the location. For example, for the United States, use `en_us`.

***.*locale*.js**
Defines the debug JavaScript content for a specific language. Replace *locale* and with the local representing code for the location. For example, for the United States, use `en_us`.

/head/*.css
CSS files are served in alphabetical order. They are grouped by file name. If they have the same name, they belong to the same group. Within a group, the following extension variations exist. There are two sets of six resources. The sets cannot be mixed.

***.css** Defines the main compressed CSS content.

***.css.uncompressed.css**
Defines the main debug CSS content.

***.rtl.css**
Defines the compressed CSS to be used for right-to-left languages.

***.rtl.css.uncompressed.css**
Defines the debug CSS to be used for right-to-left languages.

***.*locale*.css**
Defines the compressed CSS content for a specific language. Replace *locale* and with the local representing code for the location. For example, for the United States, use `en_us`.

***.*locale*.css.uncompressed.css**
Defines the debug CSS content for a specific language. Replace *locale* and with the local representing code for the location. For example, for the United States, use `en_us`.

Alternative

The following group is an alternative. You can use either of these variation group, but you cannot mix them.

***.min.css**
Defines the main compressed CSS content.

***.css** Defines the main debug CSS content.

***.rtl.min.css**
Defines the compressed CSS to be used for right-to-left languages.

***.rtl.css**
Defines the debug CSS to be used for right-to-left languages.

.locale***.min.css**
Defines the compressed CSS content for a specific language. Replace *locale* and with the local representing code for the location. For example, for the United States, use `en_us`.

.locale***.css**
Defines the debug CSS content for a specific language. Replace *locale* and with the local representing code for the location. For example, for the United States, use `en_us`.

/head/*.html
HTML files are served in alphabetical order. They are grouped by file name and if they have the same name they belong to the same group. Within a group, the following extension variations exist.

***.html** Defines the main HTML content.

***.rtl.html**
Defines the HTML to be used for right-to-left languages.

.locale***.html**
Defines the HTML content for a specific language. Replace *locale* with the local representing code for the location. For example, for the United States, use `en_us`.

/head/*device-class-name*
This optional directory scopes the resources by device class. You can use one individual device class, but it has no equation support for the directory or the files in it. You can scope the contribution to a particular device class when the incoming request is recognized as a device class name. This contribution can include JavaScript, HTML, or CSS. See the previous sections for more information.

Config contribution:

The `config` folder contains files that are served as config contribution through the resource aggregator framework. Those resources are usually before the closing body tag.

Folder: /*module-id***/config**

The following section provides a complete list of supported files within the config section.

/*module-id***/config**
Files that are stored in this directory are served in the config section of the resource aggregator.

/config/*.js

JavaScript files are served in alphabetical order. They are grouped by file name and if they have the same name they belong to the same group. Within this group, the following extension variations exist. There are two sets of six resources. The sets cannot be mixed.

***.js** Defines the main compressed JavaScript content.

***.js.uncompressed.js**
Defines the main debug JavaScript content.

***.rtl.js**
Defines the compressed JavaScript to be used for right-to-left languages.

***.rtl.js.uncompressed.js**
Defines the debug JavaScript to be used for right-to-left languages.

***.*locale*.js**
Defines the compressed JavaScript content for a specific language. Replace *locale* and with the local representing code for the location. For example, for the United States, use `en_us`.

***.*locale*.js.uncompressed.js**
Defines the debug JavaScript content for a specific language. Replace *locale* and with the local representing code for the location. For example, for the United States, use `en_us`.

The following group is an alternative. You can use either of these variation group, but you cannot mix them.

Alternative
The following group is an alternative. You can use either of these variation group, but you cannot mix them.

***.min.js**
Defines the main compressed JavaScript content.

***.js** Defines the main debug JavaScript content.

***.rtl.min.js**
Defines the compressed JavaScript to be used for right-to-left languages.

***.rtl.js**
Defines the debug JavaScript to be used for right-to-left languages.

***.*locale*.min.js**
Defines the compressed JavaScript content for a specific language. Replace *locale* and with the local representing code for the location. For example, for the United States, use `en_us`.

***.*locale*.js**
Defines the debug JavaScript content for a specific language. Replace *locale* and with the local representing code for the location. For example, for the United States, use `en_us`.

/config/*.html
HTML files are served in alphabetical order. They are grouped by file name and if they have the same name they belong to the same group. Within a group, the following extension variations exist.

***.html** Defines the main HTML content.

***.rtl.html**
Defines the HTML to be used for right-to-left languages.

***.*locale*.html**
Defines the HTML content for a specific language. Replace *locale* with the local representing code for the location. For example, for the United States, use `en_us`.

/config/*device-class-name*
This optional directory scopes the resources by device class. You can use one individual device class, but it has no equation support for the directory or the files in it. You can scope the contribution to a particular device class when the incoming request is recognized as a device class name. This contribution can include JavaScript, HTML, or CSS. See the previous sections for more information.

Dynamically extending an existing menu item from a module:

You can use a module to add menu items to a menu where the menu item displays only on certain pages,

About this task

There are three changes that are needed to add menu items with a module.

1. Define a module that includes the menu item you want to add.
2. Include the module in a profile.
3. Apply the profile to a page hierarchy.

Procedure

1. Define the module as a JSON file or as part of a `plugin.xml` file. If you define it as a JSON file, put this file in WebDAV at `fs-type1/themes/`*YourTheme*`/contributions/`. If you define it as a `plugin.xml` system module, put it in the application `.war` file in the `WEB-INF` directory.
   - Define the override module as a JSON file.

```
{
"modules":[{
  "id":"menuModule",
  "contributions":[{
   "type":"menu",
   "sub-contributions":[{
    "type":"json",
    "ref-id":"pageAction",
    "uris":[{
     "value":"/menuTest/menuItem.json"
    }]
   }]
  }]
 }]
}
```

ref-id The reference ID of the menu you want to use - this example uses the Action menu for the page.

value A pointer to the file that includes your menu item specifications - this example points to a path relative to your theme location.

- Define the override module in a `plugin.xml` file.

```
<extension point="com.ibm.portal.resourceaggregator.module" id="staticMenuModule" >
    <module id="staticMenuModule">
        <contribution type="config">
         <sub-contribution type="config_dynamic" ref-id="pageAction">
         <uri value="res://wps/defaultTheme80/themes/html/menuTest/menuAction.jsp" />
         </sub-contribution>
        </contribution>

        <contribution type="menu">
         <sub-contribution type="json" ref-id="pageAction">
         <uri value="res://wps/defaultTheme80/themes/html/menuTest/menuItem.json" />
         </sub-contribution>
        </contribution>
    </module>
</extension>
```

extension id
Any meaningful ID.

module id
Any meaningful ID.

sub-contribution ref-id
The reference ID of the menu you want to use, this example uses the **Action** menu for the page.

uri value
A pointer to the files is required to create your menu or other content. In the previous example, there are two contribution types listed. For the `menu` contribution and its `json` subcontribution, this example shows that the uri points to a JSON format file within the theme that defines one or more menu items in the JSON menu definition file syntax.

2. Create the `menuitem.json` that you defined in step 1. For more information, see *Server-side framework*.
3. Include the override module in a profile. For example, in `profile_menuModule.json`:

```
{
 "moduleIDs": [
  "menuModule",
  ...
```

The `profile_menuModule.json` file must be copied to WebDAV at `fs-type1/themes/`*`YourTheme`*`/profiles/`.

4. Open the desired menu json and add an entry for the module to the menu. Use the following example.

```
{
"type":"ModuleRef",
"id":"menuModule"
}
```

Theme Optimization Analyzer

Use the Theme Optimization Analyzer to view, but not edit, all parts of the theme optimization framework inside of WebSphere Portal. With the portlet, you can easily see which pages have specific profiles that are set or inherited. You can also see which profiles are available and belong to which theme.

Additionally you can browse and explore all aspects of the available modules: You can see what modules are loaded for a specific profile or all modules of the whole system. You can drill down into the dependency hierarchy to understand interdependencies and get different views on it, such as a parent view. The module explorer also features a rich search set so that you can easily find modules that contribute certain resources or capabilities, or browse all exposed data.

If you encounter problems, you can also export your set of data as a compressed file and share it with others. They can then import your data set and examine your profiles and modules.

Home is the welcome screen of the portlet and provides a selection of features that are supported by this portlet.

"Examine page profiles"
Use **Examine page profile information** to explore the whole page hierarchy of the system and display information about the set themes and profiles per page. It also differentiates between inherited and explicitly set themes and profiles, and locates errors such as unknown profiles that might be set. It is a great tool to find errors in your deployment.

"Examine modules" on page 2568
Use the modules sections to explore the modules of a specific profile or all of the modules available with the system, theme, or both.

"Examine contributions" on page 2577
Use the **Contributions** section to explore contributions defined as part of modules. You can view the various types of contributions and subcontributions that are provided and the Reference Identifiers that are provided as part of the subcontributions within a tree.

"Examine capabilities" on page 2583
Use the **Capabilities** section to explore capabilities that are defined as part of modules. You can view the various names and versions in one list, with the modules that exposed them shown.

"Validation reports" on page 2589
Use the validation portlet to verify that your theme contains no errors. The validation report analyzes your theme and theme components for known issues and reports the number of errors, warnings, and informational messages. It also includes a detailed explanation about how to fix the errors that occur.

"Utilities" on page 2599
If you need help with a theme issue, you can export your system data. Support personnel can then import and view your data as a simulated system to troubleshoot problems.

Examine page profiles:

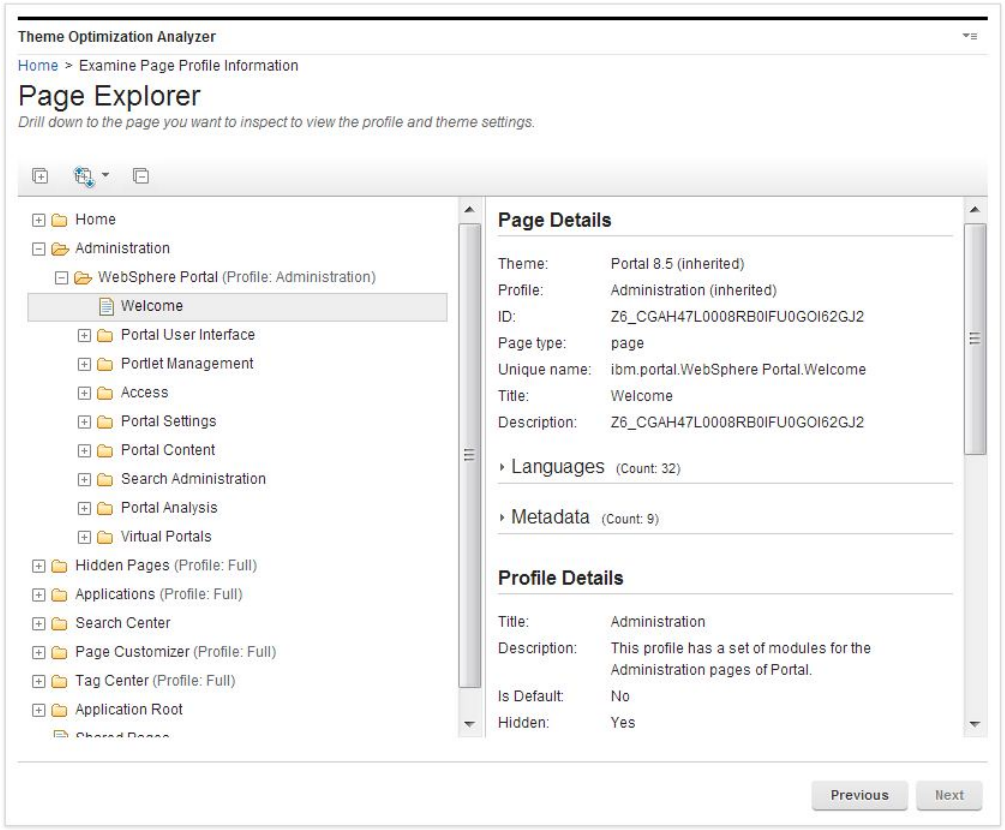
Use **Examine page profile information** to explore the whole page hierarchy of the system and display information about the set themes and profiles per page. It also differentiates between inherited and explicitly set themes and profiles, and locates errors such as unknown profiles that might be set. It is a great tool to find errors in your deployment.

When you select the **Examine page profile information** on the **Home** screen, the Page Explorer screen displays.

Note: In simulation mode, the **Examine page profile information** section is not available and cannot be clicked.

Page explorer

The page explorer displays the page hierarchy of your system in a split view. The tree view shows the page hierarchy and the details view shows information about the branch that is selected in the tree view.



In the tree view, when a page has a specified profile that is not inherited from the parent, the profile name displays in parentheses after the page name.

The details view displays details for the selected page as well as the theme and profile that are used on the page. If a theme or profile is inherited, it is displayed in parentheses after the theme or profile name. When a theme or profile is set specifically, no comment is displayed.

When you expand a tree or branch, if it is large and takes more than 30 seconds to expand, the expansion process is stopped. You must expand those branches individually.

The page explorer provides specialized view of your website pages to shows theme information. It is not intended to provide general information about pages.

Examine modules:

Use the modules sections to explore the modules of a specific profile or all of the modules available with the system, theme, or both.

Reference Identifiers that are provided as part of the subcontributions within a tree.

CF03 You have the option in this screen to continue with **Examine modules by page**, **Examine modules by profile**, or **Examine all modules**.

CF03

Modules by pages

Use the Examine modules by page section to explore modules that are defined on a specific page. Modules can be contributed to a page with its profile or portlets. Select a page in the Select Page screen and then advance to the Module Explorer to explore the modules in that page.

The Select Page screen displays the site's page hierarchy in the tree view and shows details on the selected branch in the details view. You can now learn more about the pages before you select one to examine.

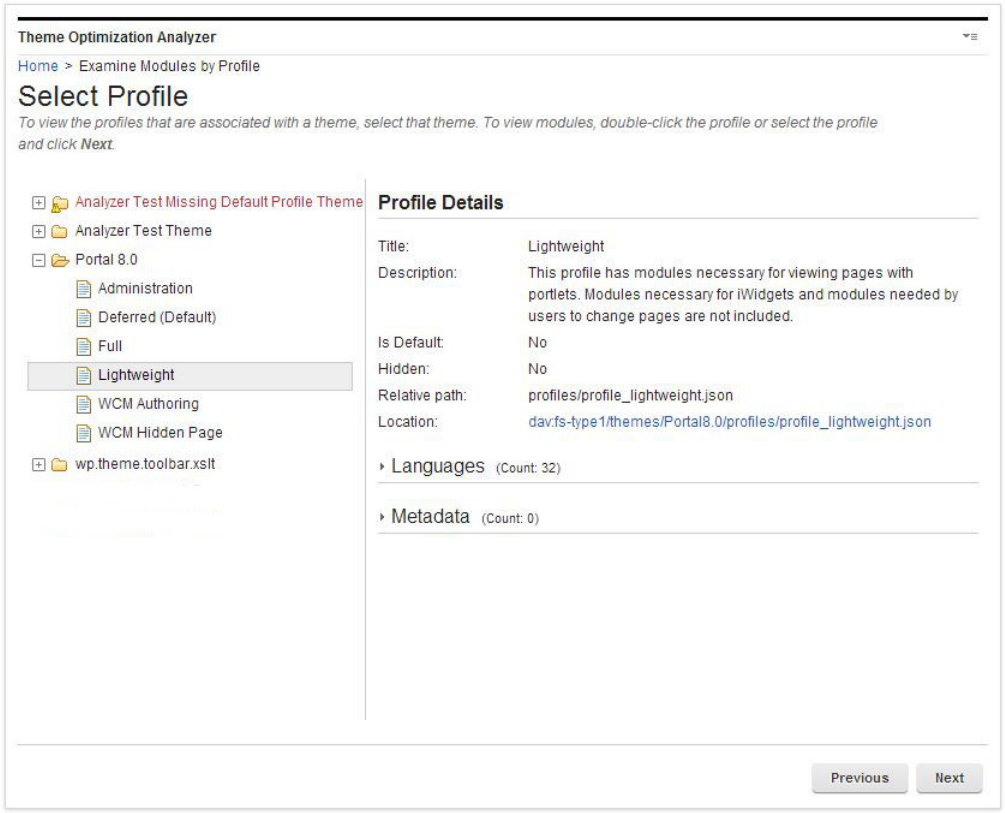
In the tree view, when a page has a specified profile that is not inherited from the parent, the profile name displays in parentheses after the page name. All other information is shown in the details view, such as the page, profile, and theme information.

After you select a page, the Module Explorer displays.

Modules by profile

Use the **Examine modules by profile** section to explore the modules of a specific profile. Select a profile in the **Select Profile** screen and then advance to the Module Explorer to explore the modules in that profile.

The **Select Profile** screen displays the themes and their profiles in the tree view and shows details on the selected branch in the details view. You can now learn more about the various artifacts that are installed in the system before you select a profile to examine.



The tree view also shows in parentheses which theme and profile is the default so this information is easily visible. All other information is shown in the details view, such as the location of the profile JSON file. The profile JSON file is an element that can be clicked and opens a new window and serves the resource in the browser when clicked.

After you select a profile, the Module Explorer screen will display.

All modules

Use the **Examine all modules** section to explore all modules available with the system, theme, or both. Select a theme or the system module branch on the **Select Theme** screen and then advance to the Module Explorer.

The **Select Theme** screen displays a system modules branch and all themes in the tree view. Select a theme to view more information about the various artifacts that are installed in the system before you decide on a theme and continue to the Module Explorer.

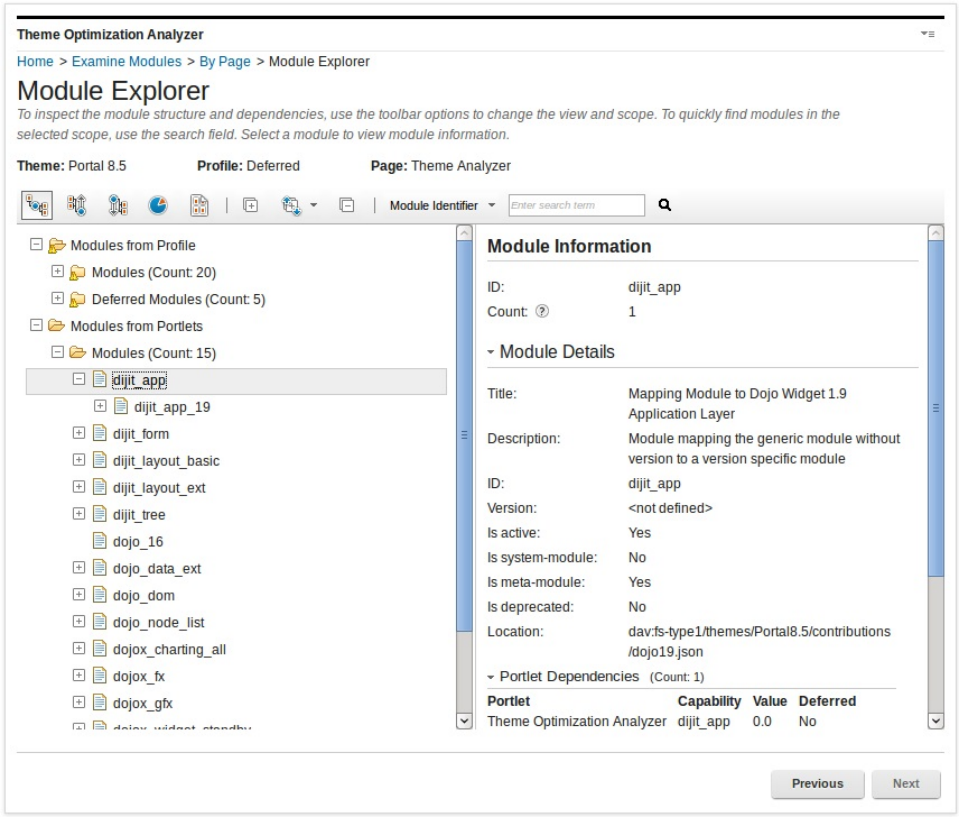
Module explorer

The module explorer displays the module hierarchy of your system in a split view. The tree view shows the module hierarchy, the details view shows details on the selected module from the tree view.

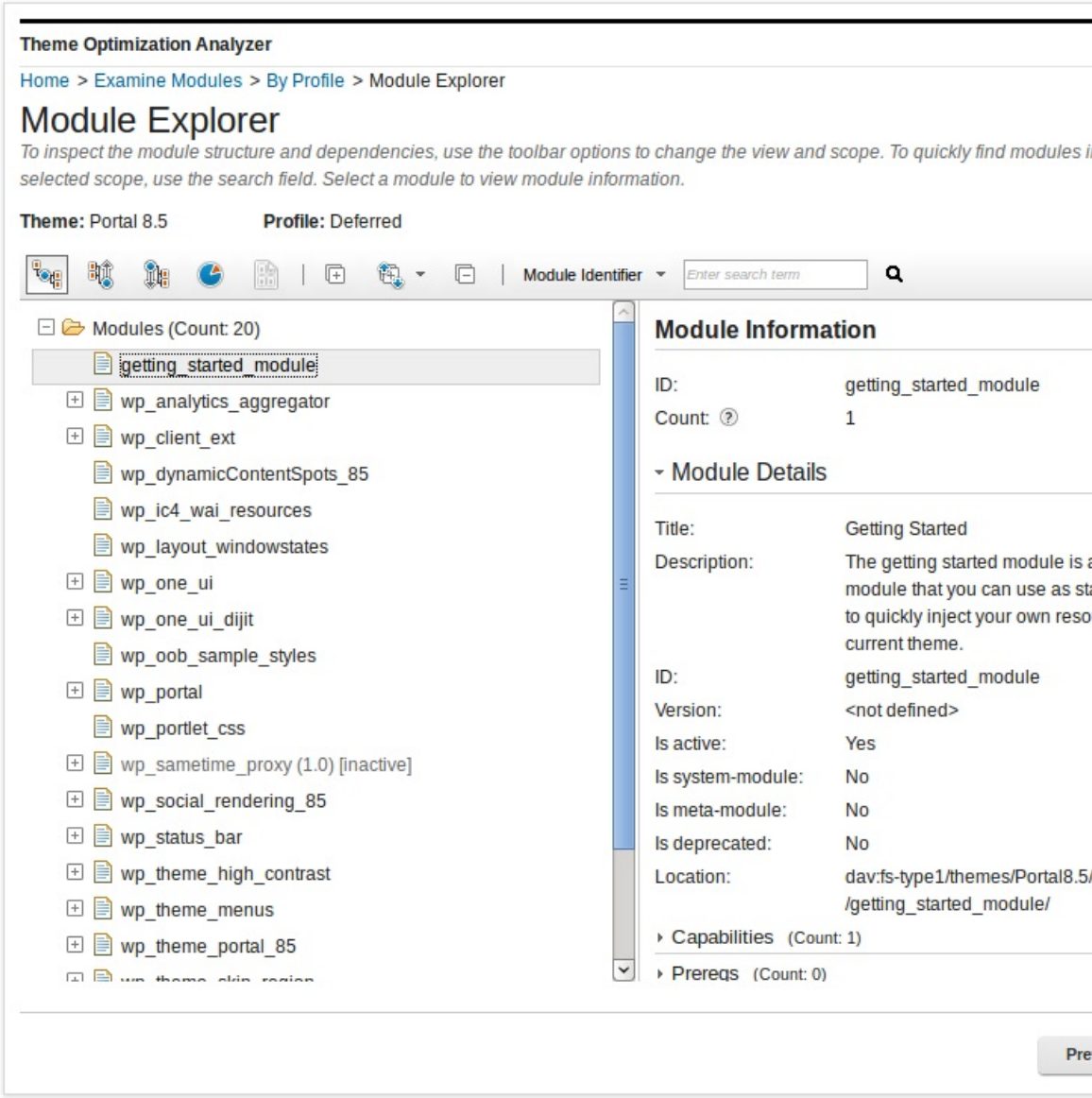
You can get to the module explorer from different paths in the UI.

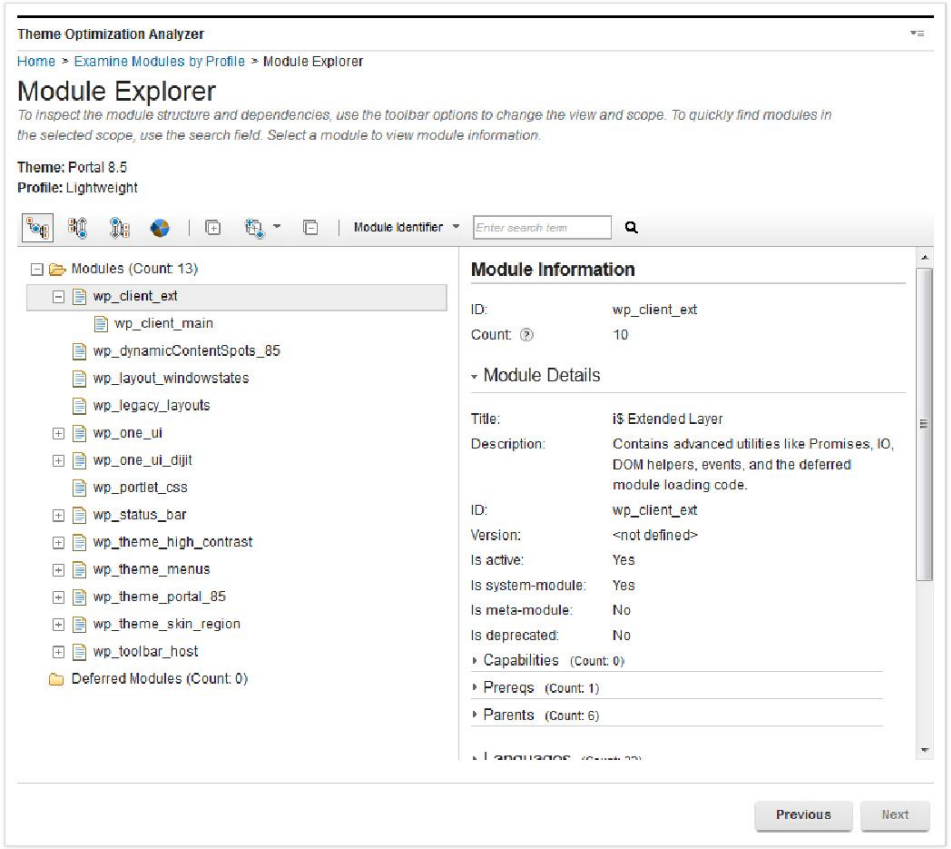
1. CF03 When you examine the modules that are scoped by page, you have two root branches, each with two childs:

- **Modules from Profile**: The root for all modules in the profile.
  - **Modules**: All modules that are in the non-deferred section of the profile.
  - **Deferred Modules**: All modules that are in the deferred section of the profile.
- **Modules from Portlets**: The root for all modules that are provided by capabilities that are assigned by the page's portlets.
  - **Modules**: All modules that are provided by non-deferred capabilities that are assigned by the page's portlets.
  - **Deferred Modules**: All modules that are provided by deferred capabilities that are assigned by the page's portlets.



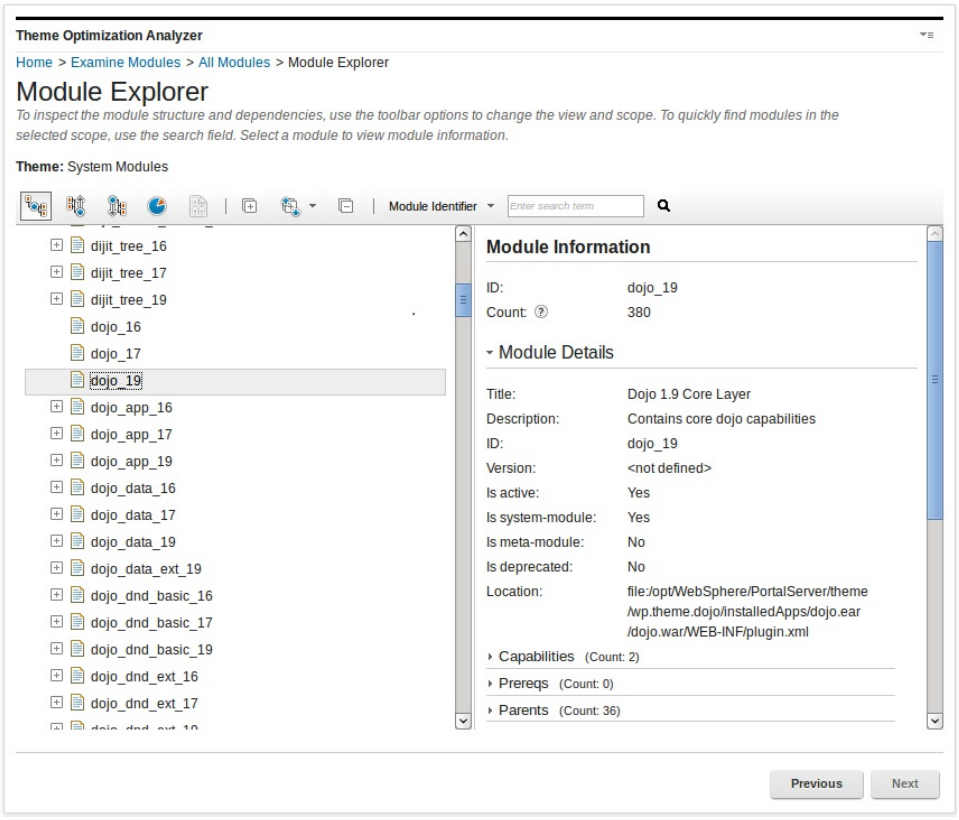
2. When you examine modules by profile, you have two root branches:
   - **Modules**: The root for all modules in the none deferred section of the profile.
   - **Deferred Modules**: The root for all modules in the deferred section of the profile.

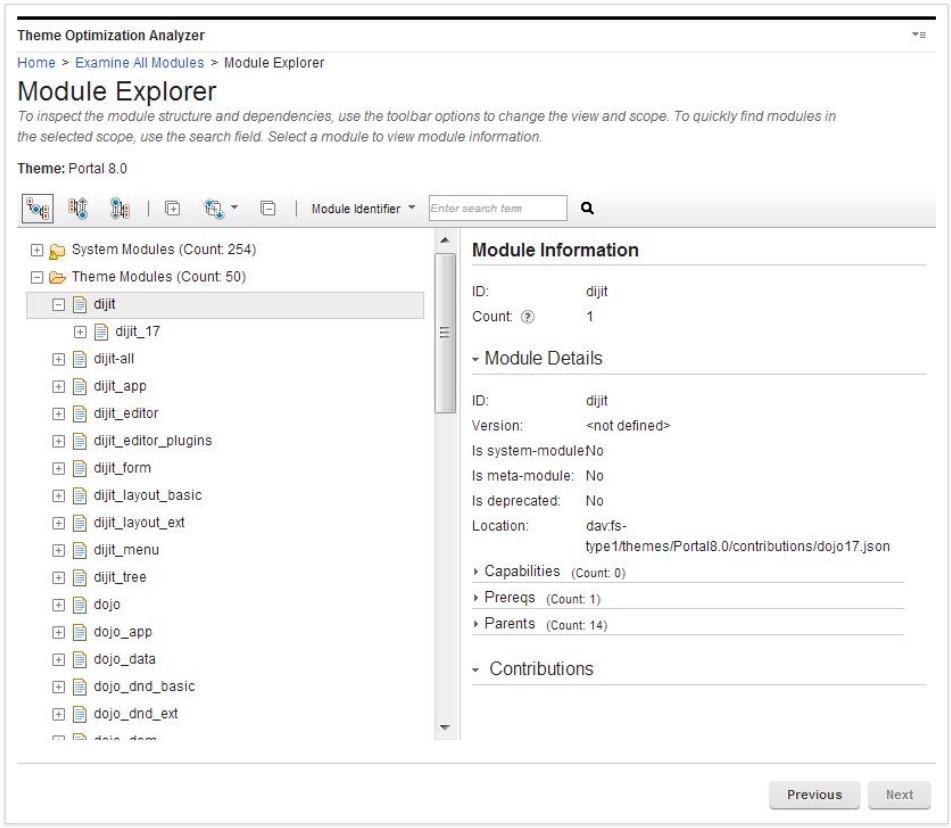

Theme Optimization Analyzer
Home > Examine Modules > By Profile > Module Explorer
Module Explorer
To inspect the module structure and dependencies, use the toolbar options to change the view and scope. To quickly find modules i
selected scope, use the search field. Select a module to view module information.
Theme: Portal 8.5
Profile: Deferred
Module Identifier
Enter search term
Modules (Count: 20)
getting_started_module
wp_analytics_aggregator
wp_client_ext
wp_dynamicContentSpots_85
wp_ic4_wai_resources
wp_layout_windowstates
wp_one_ui
wp_one_ui_dijit
wp_oob_sample_styles
wp_portal
wp_portlet_css
wp_sametime_proxy (1.0) [inactive]
wp_social_rendering_85
wp_status_bar
wp_theme_high_contrast
wp_theme_menus
wp_theme_portal_85
Module Information
ID:
getting_started_module
Count:
1
Module Details
Title:
Getting Started
Description:
The getting started module is a
module that you can use as sta
to quickly inject your own reso
current theme.
ID:
getting_started_module
Version:
<not defined>
Is active:
Yes
Is system-module:
No
Is meta-module:
No
Is deprecated:
No
Location:
dav:fs-type1/themes/Portal8.5/
/getting_started_module/
Capabilities (Count: 1)
Prereqs (Count: 0)



3. When you examine the system and theme modules, you have one or two branches:
   - **`System Modules`**: Contains all modules that are defined globally through a `plugin.xml`.
   - **`Theme Modules`**: Contains all modules that are defined within the themes contribution folder as part of the JSON files. This branch is only shown when a theme is selected.

CF03


Theme Optimization Analyzer
Home > Examine Modules > All Modules > Module Explorer
Module Explorer
To inspect the module structure and dependencies, use the toolbar options to change the view and scope. To quickly find modules in the selected scope, use the search field. Select a module to view module information.
Theme: System Modules
Module Identifier
Enter search term
dijit_tree_16
dijit_tree_17
dijit_tree_19
dojo_16
dojo_17
dojo_19
dojo_app_16
dojo_app_17
dojo_app_19
dojo_data_16
dojo_data_17
dojo_data_19
dojo_data_ext_19
dojo_dnd_basic_16
dojo_dnd_basic_17
dojo_dnd_basic_19
dojo_dnd_ext_16
dojo_dnd_ext_17
Module Information
ID: dojo_19
Count: 380
Module Details
Title: Dojo 1.9 Core Layer
Description: Contains core dojo capabilities
ID: dojo_19
Version: <not defined>
Is active: Yes
Is system-module: Yes
Is meta-module: No
Is deprecated: No
Location: file:/opt/WebSphere/PortalServer/theme /wp.theme.dojo/installedApps/dojo.ear /dojo.war/WEB-INF/plugin.xml
Capabilities (Count: 2)
Prereqs (Count: 0)
Parents (Count: 36)
Previous
Next


Theme Optimization Analyzer
Home > Examine All Modules > Module Explorer
Module Explorer
To inspect the module structure and dependencies, use the toolbar options to change the view and scope. To quickly find modules in the selected scope, use the search field. Select a module to view module information.
Theme: Portal 8.0
Module Identifier
Enter search term
System Modules (Count: 254)
Theme Modules (Count: 50)
dijit
dijit_17
dijit-all
dijit_app
dijit_editor
dijit_editor_plugins
dijit_form
dijit_layout_basic
dijit_layout_ext
dijit_menu
dijit_tree
dojo
dojo_app
dojo_data
dojo_dnd_basic
dojo_dnd_ext
Module Information
ID: dijit
Count: 1
Module Details
ID: dijit
Version: <not defined>
Is system-module: No
Is meta-module: No
Is deprecated: No
Location: dav:fs-type1/themes/Portal8.0/contributions/dojo17.json
Capabilities (Count: 0)
Prereqs (Count: 1)
Parents (Count: 14)
Contributions
Previous
Next

When you expand a tree or branch, if it is large and takes more than 30 seconds to expand, the expansion process is stopped. You must expand those branches individually.

The module hierarchy has three main views:

Complete hierarchy
This view displays the root and a list of modules underneath as child branches. The child branches for each module then represent the prerequisite dependencies that each module defines. You can drill down until no prerequisites are defined. Therefore, within the tree various modules might display multiple times since they can be defined as a prerequisite for many different modules.

For example: The `wp_toolbar` module, which is referenced in the deferred profile, defines the `dojo` module as a prerequisite. Then, the `dojo` module defines the `dojo_17` module as a prerequisite. When you look at the module explorer, you can drill down into the **Deferred Modules** section, which displays the `wp_toolbar` module. If you drill down further, you can see the `dojo` module is child of `wp_toolbar` and the `dojo_17` module is child of the `dojo` module.

Parent view of a selected branch
This view displays the parents of a module. The parent is a module that has a prerequisite of a child module. Module A has a prerequisite of module B, which means that B 's parent is A. It is useful to figure out who is using a specific module and verify whether the usage is correct.

Note: The parent is displayed as a child branch within the view, the tree is upside down.

For instance: When you look at the `wp_client_main` module in the parent view, it has a child branch in the tree called `wp_status_bar`. Therefore, `wp_status_bar` is the parent of `wp_client_main`.

Child view of a selected branch
This view behaves identically to the complete hierarchy view except that it focused on a single branch as the parent. The dependencies of just this single branch are displayed.

Size explorer

This view shows a pie chart representation of the relative sizes of the modules and size details.

The size details panel shows three representations of a module's size. Each section of the panel computes the size differently.

The first section shows the total download size of the module and its dependencies. Duplicate dependencies are included once in the size calculation.

The middle section shows the total download size for each child of the currently selected module. Dependencies that are shared between siblings are included in the size calculation of each child module.

The last section shows all descendants of the currently selected module and the size of each module, not including the size of the module's prerequisites.

For example, assume that each module size is 1 KB. The hierarchy looks like the following, and `module_0` is the current module.

- module_0
  - module_1
    - module_2
    - module_3
  - module_2
    - module_3
  - module_3

The size of `module_0` in the first section is 4 KB, which includes the size of the module and each of its descendants. `module_2` is a prerequisite for both `module_0` and `module_1` so it is counted only once. The middle section includes `module_1`, `module_2`, and `module_3`. The size of `module_1` is 3 KB. The last section includes all of `module_0`'s descendants or `module_1`, `module_2`, and `module_3` and the size of each of those modules is 1 KB.

You can view sizes as compressed or uncompressed and you can switch between them by clicking the compression icon in the toolbar.

You can browse through the Size Explorer by clicking the segments of the pie chart, or the links in the child section for more details. To return to a higher level, click the module from the Size Explorer breadcrumb trail or click the **Go to Parent** link.

Portlet explorer.
The view displays a list of all portlets on the page. The childs of the portlets are the capabilities depended on by each portlet. Under each capability, are the modules that define the capability. When a portlet is selected, the details view shows its ID, titles, descriptions, capabilities, and preferences.

The details view displays the details for the selected module such as:

- ID: Displays the module ID.
- Version: Displays the module version, if one was defined.
- Is system-module: Yes if the module is globally defined through a `plugin.xml` file.
- Is meta-module: Yes if the module does not contain any contributions, but instead contains only prerequisites.
- Is deprecated: Yes, if the module was deprecated.
- Location: Displays the file path for the module.
- Capabilities: Displays all capabilities for the module.
- Prereqs: Displays all prerequisites for the module.
- Parents: Displays all parent modules for the module
- Contributions: Displays all contributions by contribution and sub contribution. The resources are elements that can be clicked, which open a new window and serves the resource in the browser when clicked.

  Note: In simulation mode, the resources cannot be clicked.

Searching within the module explorer

Use the search bar to search the tree view and the details view for various information. The following search scopes are supported:

- **`Module Identifier`**: Search for a specific module by name. For example, `wp_client_main`.

- **`Resource`**: Search for the module that exposes a certain resource. For example, enter `master.css` to identify the module that exposes this resource. This search is useful if you find an error in a JavaScript file in your browser. You can easily find the module that this resource belongs to.
- **`Capability`**: Search for the modules that are associated with a specific capability. For example, if you want to know which module provides the `dojo` capability. Then, you know which module you must add as a prerequisite for your own module.
- **`Reference Identifiers`**: Search for which dynamic content spot or module reference is exposed by which module.

Enter a term into the search field in the toolbar and press Enter or click the **Find** icon. The found item is then highlighted for a short time with a yellow background. Pressing Enter again or clicking **Find** icon again will find the next item. If no item is found, either a pop-up dialog opens to instruct you to start the search from the beginning or the background of the search field is marked in red.

Examine contributions:

Use the **Contributions** section to explore contributions defined as part of modules. You can view the various types of contributions and subcontributions that are provided and the Reference Identifiers that are provided as part of the subcontributions within a tree.

You have the option in this screen to continue with CF03 **Examine contributions by page, Examine contributions by profiles**, or **Examine all contributions**.

CF03
Contributions by pages

Use the Examine contributions by page section to explore contributions that are defined on a specific page. Modules can be contributed to a page via its profile or portlets. Select a page in the Select Page screen and then advance to the Contribution Explorer to explore the contributions in that page.

The Select Page screen displays the site's page hierarchy in the tree view and shows details on the selected branch in the details view. You can now learn more about the pages before you select one to examine.

In the tree view, when a page has a specified profile that is not inherited from the parent, the profile name displays in parentheses after the page name. All other information is shown in the details view, such as the page, profile and theme information.

After you select a page, the Contribution Explorer screen displays.

Contributions by profiles

Use the **Examine contributions by profiles** section explore contributions that are defined as part of modules for a specific profile. Select a profile in the **Select Profile** screen and then advance to the Contribution Explorer to explore the contributions in that profile.

The **Select Profile** screen displays the themes and their profiles in the tree view and shows details on the selected branch in the details view. You can now learn more about the various artifacts that are installed in the system before you select a profile to examine.

The tree view also shows in parentheses which theme and profile is the default so this information is easily visible. All other information is shown in the details view, such as the location of the profile JSON file. The profile JSON file is an element that can be clicked and opens a new window and serves the resource in the browser when clicked.

After you select a profile, the Contribution Explorer screen will display.

All contributions

Use the **Examine all contributions** section to explore all contributions that are provided by modules available with the system, theme, or both. Select a theme or the system module branch in the **Select Theme** screen and then advance to the Contributions Explorer.

The **Select Theme** screen displays a system modules branch and all themes in the tree view. Select a theme to view more information about the various artifacts that are installed in the system before you decide on a theme and continue to the Contribution Explorer. To view the contributions in a theme, double-click the theme or select the theme and click **Next**.

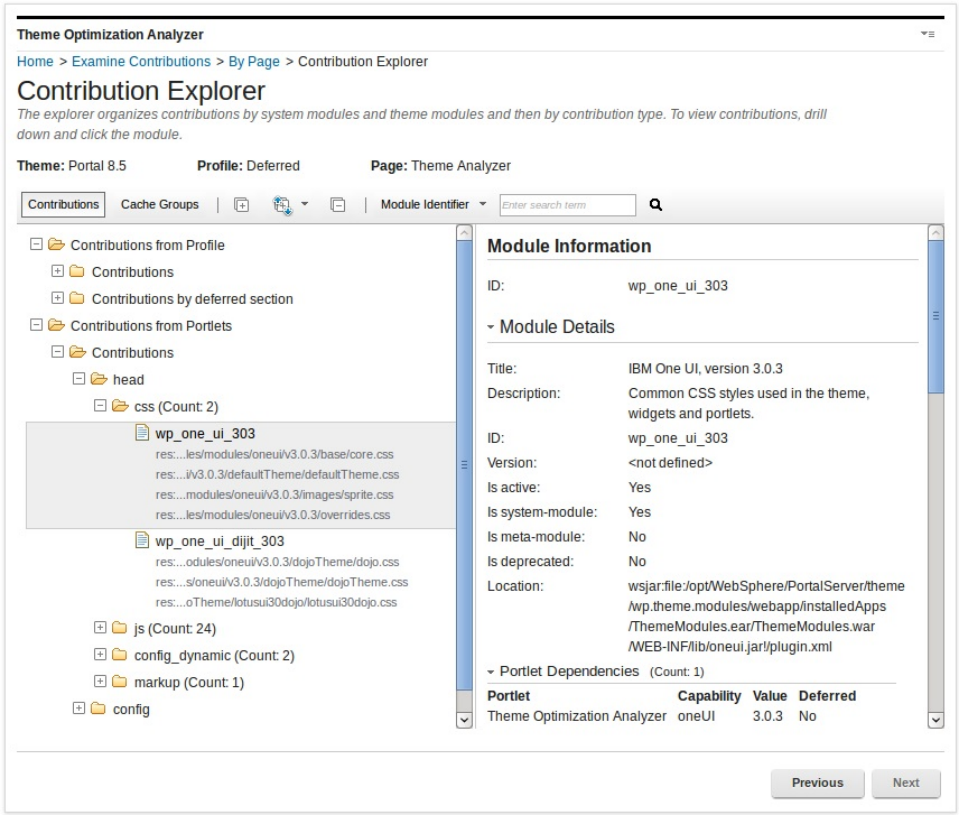
Contribution explorer

The contribution explorer displays the contribution type hierarchy of your system in a split view. The tree view shows the contribution and subcontribution types with modules as the leaf, the details view shows details on the selected module from the tree view.

You can get to the contribution explorer from different paths in the UI.

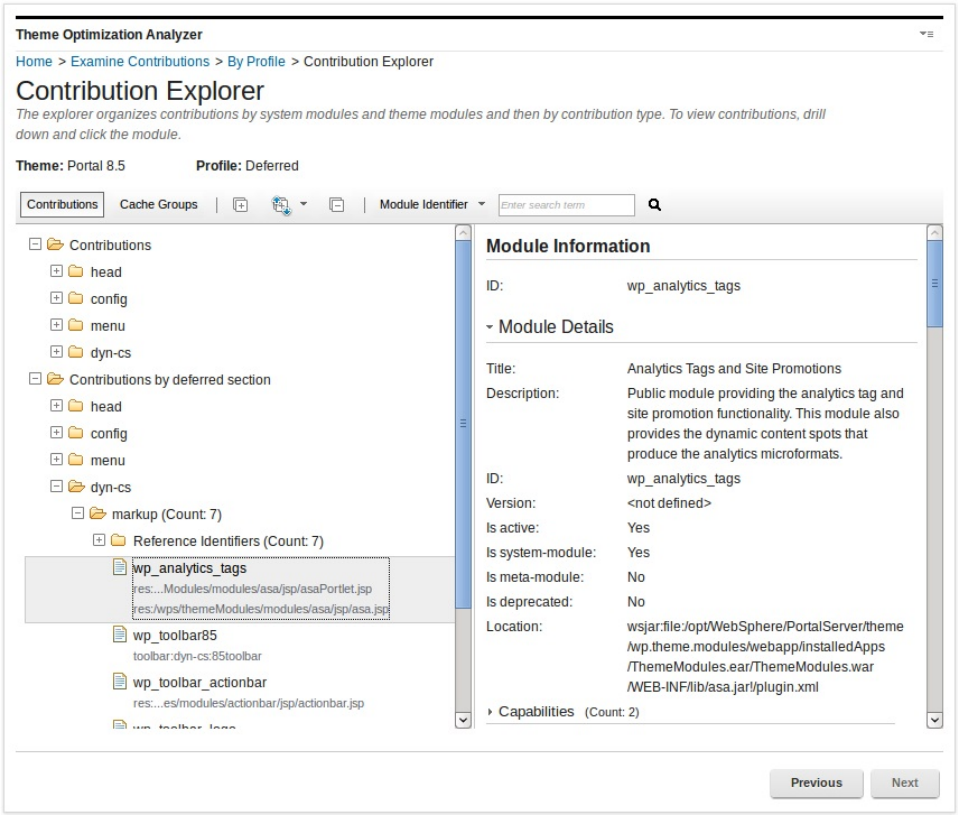
1. CF03 When you examine the contributions that are scoped by page, you have two root branches, each with two children:
   - **Contributions from Profile**: The root for all contributions that are provided by modules in the profile.
     - **Contributions**: All contributions that are provided by modules in the non-deferred section of the profile.
     - **Contributions by a deferred section**: All contributions that are provided by modules in the deferred section of the profile.
   - **Contributions from Portlets**: The root for all contributions that are provided by capabilities assigned by the page's portlets.
     - **Contributions**: All contributions that are provided by non-deferred capabilities assigned by the page's portlets.
     - **Contributions by a deferred section**: All contributions that are provided by deferred capabilities assigned by the page's portlets.

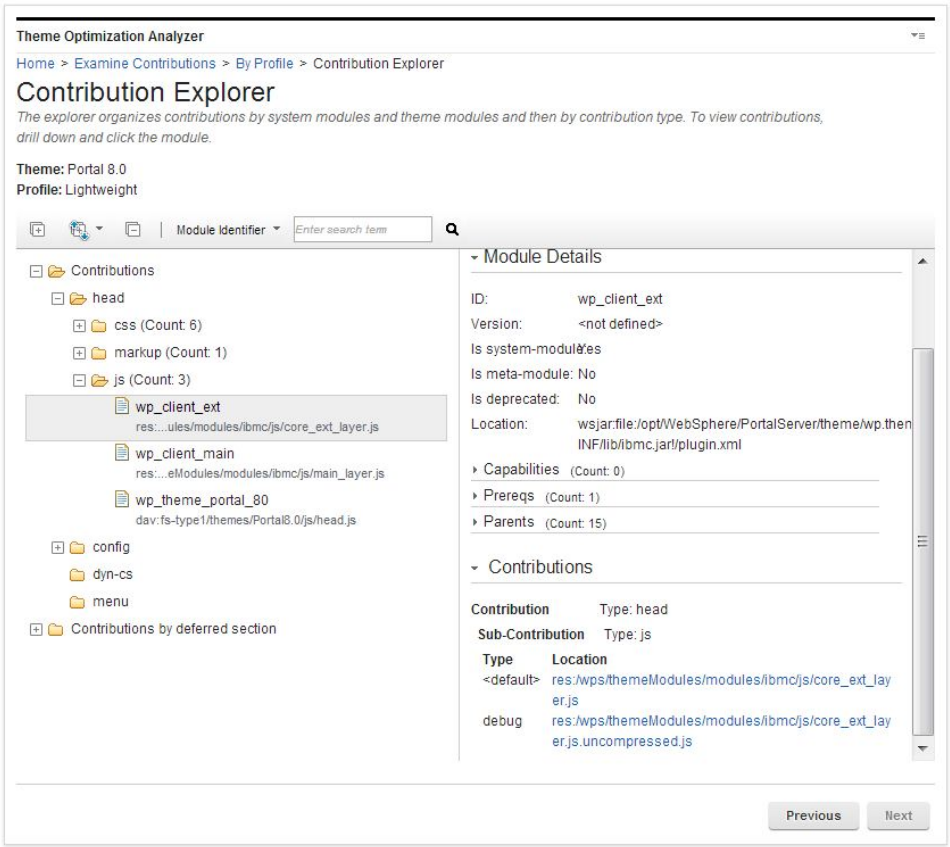
   CF03



2. When you examine the contributions that are scoped by profile, you have two root branches:
   - **`Contributions`**: The root for all contributions that are provided by modules in the non-deferred section of the profile.
   - **`Contributions by a deferred section`**: The root for all contributions that are provided by modules in the deferred section of the profile.

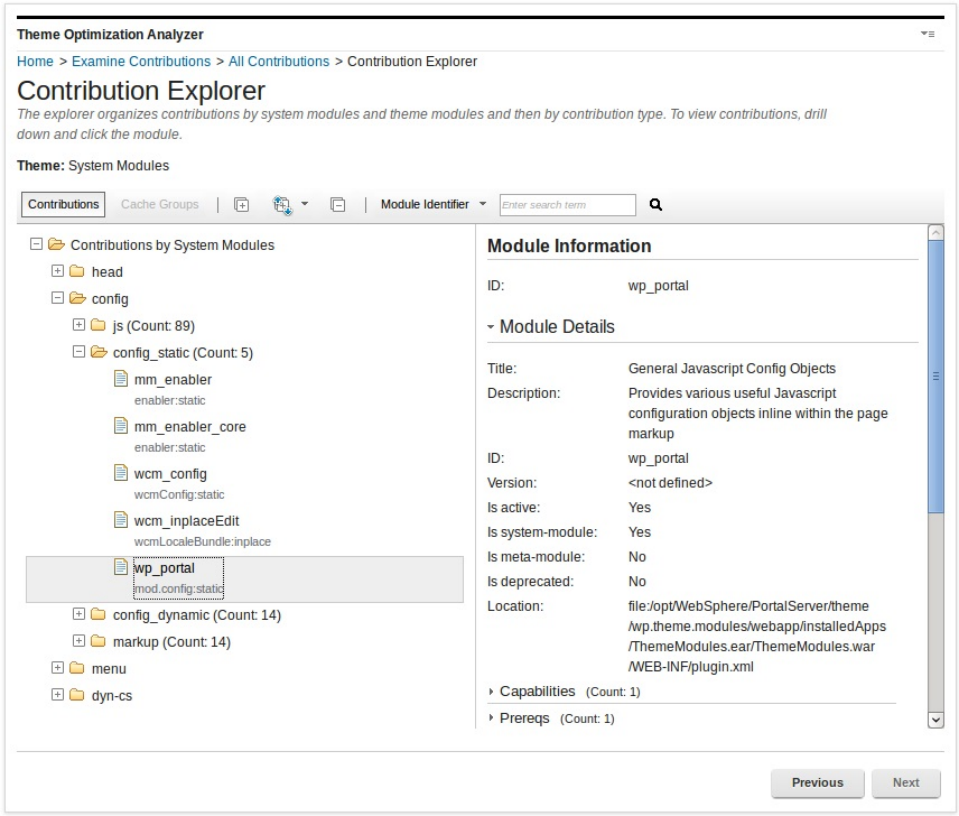
CF03

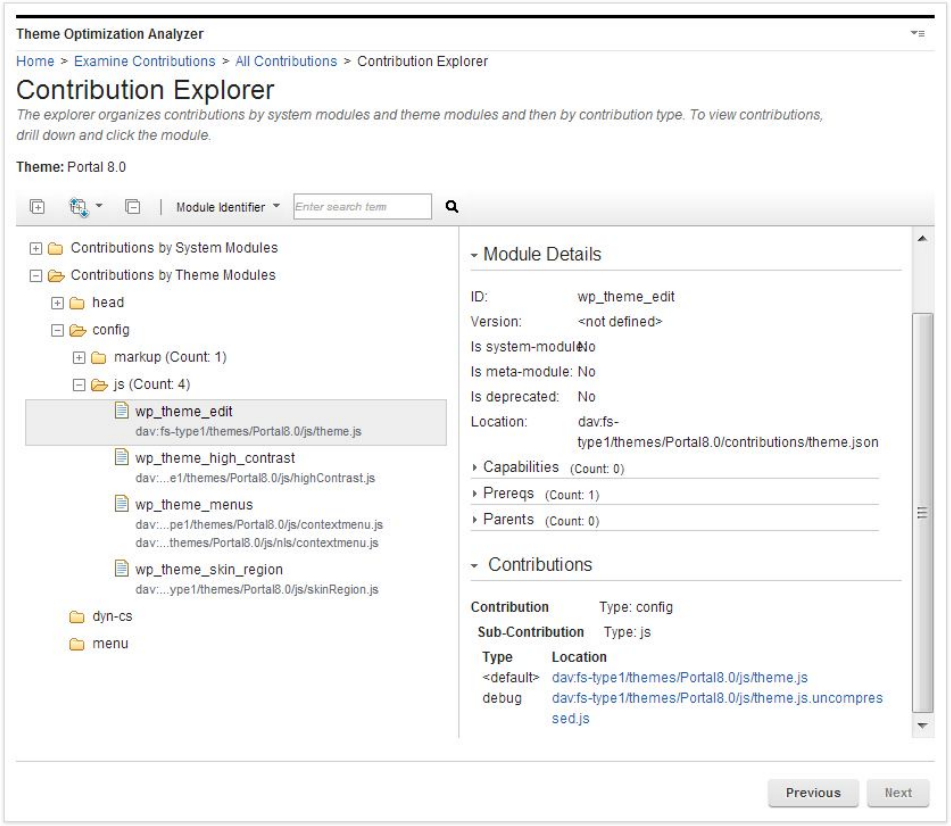

Theme Optimization Analyzer
Home > Examine Contributions > By Profile > Contribution Explorer
Contribution Explorer
The explorer organizes contributions by system modules and theme modules and then by contribution type. To view contributions, drill down and click the module.
Theme: Portal 8.5
Profile: Deferred
Contributions
Cache Groups
Module Identifier
Enter search term
Contributions
head
config
menu
dyn-cs
Contributions by deferred section
head
config
menu
dyn-cs
markup (Count: 7)
Reference Identifiers (Count: 7)
wp_analytics_tags
res:...Modules/modules/asa/jsp/asaPortlet.jsp
res:/wps/themeModules/modules/asa/jsp/asa.jsp
wp_toolbar85
toolbar:dyn-cs:85toolbar
wp_toolbar_actionbar
res:...es/modules/actionbar/jsp/actionbar.jsp
Module Information
ID: wp_analytics_tags
Module Details
Title: Analytics Tags and Site Promotions
Description: Public module providing the analytics tag and site promotion functionality. This module also provides the dynamic content spots that produce the analytics microformats.
ID: wp_analytics_tags
Version: <not defined>
Is active: Yes
Is system-module: Yes
Is meta-module: No
Is deprecated: No
Location: wsjar:file:/opt/WebSphere/PortalServer/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/asa.jar!/plugin.xml
Capabilities (Count: 2)
Previous
Next


Theme Optimization Analyzer
Home > Examine Contributions > By Profile > Contribution Explorer
Contribution Explorer
The explorer organizes contributions by system modules and theme modules and then by contribution type. To view contributions, drill down and click the module.
Theme: Portal 8.0
Profile: Lightweight
Module Identifier
Enter search term
Contributions
head
css (Count: 6)
markup (Count: 1)
js (Count: 3)
wp_client_ext
res:...ules/modules/ibmc/js/core_ext_layer.js
wp_client_main
res:...eModules/modules/ibmc/js/main_layer.js
wp_theme_portal_80
dav:fs-type1/themes/Portal8.0/js/head.js
config
dyn-cs
menu
Contributions by deferred section
Module Details
ID: wp_client_ext
Version: <not defined>
Is system-module: Yes
Is meta-module: No
Is deprecated: No
Location: wsjar:file:/opt/WebSphere/PortalServer/theme/wp.then
INF/lib/ibmc.jar!/plugin.xml
Capabilities (Count: 0)
Prereqs (Count: 1)
Parents (Count: 15)
Contributions
Contribution Type: head
Sub-Contribution Type: js
Type Location
<default> res:/wps/themeModules/modules/ibmc/js/core_ext_layer.js
debug res:/wps/themeModules/modules/ibmc/js/core_ext_layer.js.uncompressed.js
Previous
Next

3. When you examine the system and theme contributions, you have one or two branches:
   - **`Contributions by System Modules`**: Contains all contributions that are provided by modules that are defined globally through a `plugin.xml` file.
   - **`Contributions by Theme Modules`**: Contains all contributions that are provided by modules that are defined within the themes contribution folder as part of the JSON files. This branch is only shown when a theme is selected.

CF03





When you expand a tree or branch, if it is large and takes more than 30 seconds to expand, the expansion process is stopped. You must expand those branches individually.

The details view displays the details for the selected module such as:

- ID: Displays the module ID.
- Version: Displays the module version, if one was defined.
- Is system-module: Yes if the module is globally defined through a `plugin.xml` file.
- Is meta-module: Yes if the module does not contain any contributions, but instead contains only prerequisites.
- Is deprecated: Yes, if the module was deprecated.
- Location: Displays the file path for the module.
- Capabilities: Displays all capabilities for the module.
- Prereqs: Displays all prerequisites for the module.
- Parents: Displays all parent modules for the module
- Contributions: Displays all contributions by contribution and sub contribution. The resources are elements that can be clicked, which open a new window and serves the resource in the browser when clicked.

  Note: In simulation mode, the resources cannot be clicked.

Searching within the contribution explorer

Use the search bar to search the tree view and the details view for various information. The following search scopes are supported:

- **`Module Identifier`**: Search for a specific module by name. For example, `wp_client_main`.
- **`Resource`**: Search for the module that exposes a certain resource. For example, enter `master.css` to identify the module that exposes this resource. This search is useful if you find an error in a JavaScript file in your browser. You can easily find the module that this resource belongs to.
- **`Capability`**: Search for the modules that are associated with a specific capability. For example, if you want to know which module provides the `dojo` capability. Then, you know which module you must add as a prerequisite for your own module.
- **`Reference Identifiers`**: Search for which dynamic content spot or module reference is exposed by which module.

Enter a term into the search field in the toolbar and press Enter or click the **Find** icon. The found item is then highlighted for a short time with a yellow background. Pressing Enter again or clicking **Find** icon again will find the next item. If no item is found, either a pop-up dialog opens to instruct you to start the search from the beginning or the background of the search field is marked in red.

Cache groups

For every combined request that the theme optimization framework produces, you can see the overall ability to cache that request and how those caches are calculated. You can select a cache group or resources and examine it.

Examine capabilities:

Use the **Capabilities** section to explore capabilities that are defined as part of modules. You can view the various names and versions in one list, with the modules that exposed them shown.

You have the option in this screen to continue with CF03 **Examine contributions by page, Examine capabilities by profile** or **Examine all capabilities**.

CF03
Capabilities by pages

Use the Examine capabilities by page section to explore capabilities that are defined on a specific page. Capabilities can be contributed to a page via its profile or portlets. Select a page in the Select Page screen and then advance to the Capability Explorer to explore the capabilities on that page.

The Select Page screen displays the site's page hierarchy in the tree view and shows details on the selected branch in the details view. You can now learn more about the pages before you select one to examine.

In the tree view, when a page has a specified profile that is not inherited from the parent, the profile name displays in parentheses after the page name. All other information is shown in the details view, such as the page, profile and theme information.

After you select a page, the Capability Explorer screen displays.

Capabilities by profile

Use the **Examine capabilities by profile** section to explore capabilities that are defined as part of modules for a specific profile. To do that you must first select a profile in the Profile Selection screen and then advance to the Capability Explorer.

The **Select Profile** screen displays the themes and their profiles in the tree view and shows details on the selected branch in the details view. You can now learn more about the various artifacts that are installed in the system before you select a profile to examine.

The tree view also shows in parentheses which theme and profile is the default so this information is easily visible. All other information is shown in the details view, such as the location of the profile JSON file. The profile JSON file is an element that can be clicked and opens a new window and serves the resource in the browser when clicked.

After you select a profile, the Capability Explorer screen will display.

All Capabilities

Use the **Examine all capabilities** section to explore all capabilities that are provided by modules available with the system, theme, or both. Select a theme or the system module branch in the **Select Theme** screen and then advance to the Capability Explorer.

The **Select Theme** screen displays a system modules branch and all themes in the tree view. Select a theme to view more information about the various artifacts that are installed in the system before you decide on a theme and continue to the Capability Explorer. To view the capabilities in a theme, double-click the theme or select the theme and click **Next**.

Capability explorer

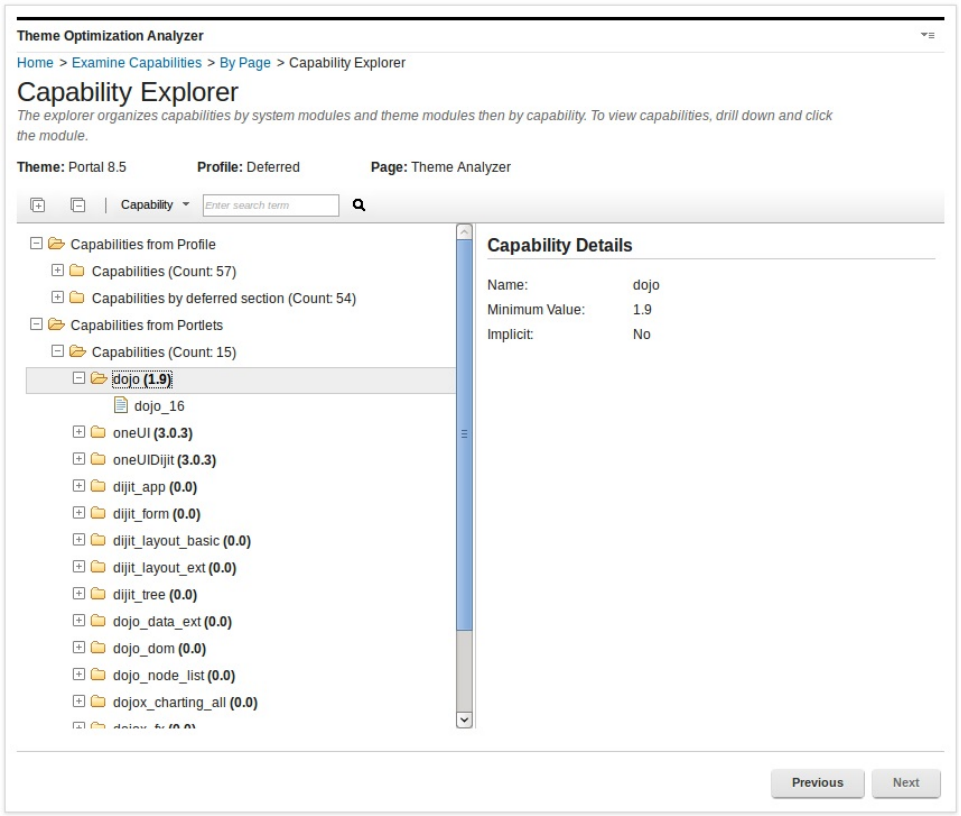
The capability explorer displays the capability names and versions that are provided by a set of modules in a split view. The tree view shows the capabilities and respective module that defines the capability, the details view shows details on the selected module from the tree view.

You can get to the capability explorer from different paths in the UI.

1. When you examine the capabilities that are scoped by page, you have two root branches, each with two children:
   - **`Capabilities from Profile`**: The root for all capabilities that are provided by modules in the profile.
     - **`Capabilities`**: All capabilities that are provided by modules in the non-deferred section of the profile.
     - **`Capabilities by a deferred section`**: All capabilities that are provided by modules in the deferred section of the profile.
   - **`Capabilities from Portlets`**: The root for all capabilities that are assigned by the page's portlets.
     - **`Capabilities`**: All non-deferred capabilities that are provided by the page's portlets.

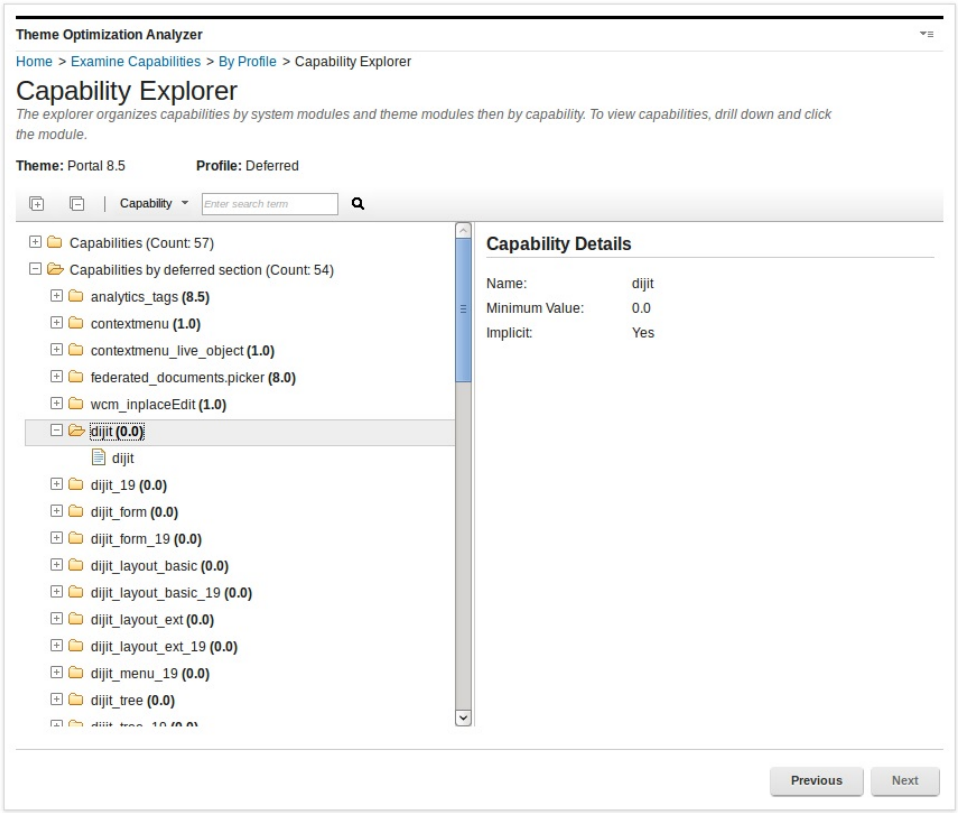
– **Capabilities by a deferred section**: All deferred capabilities that are provided by the page's portlets.

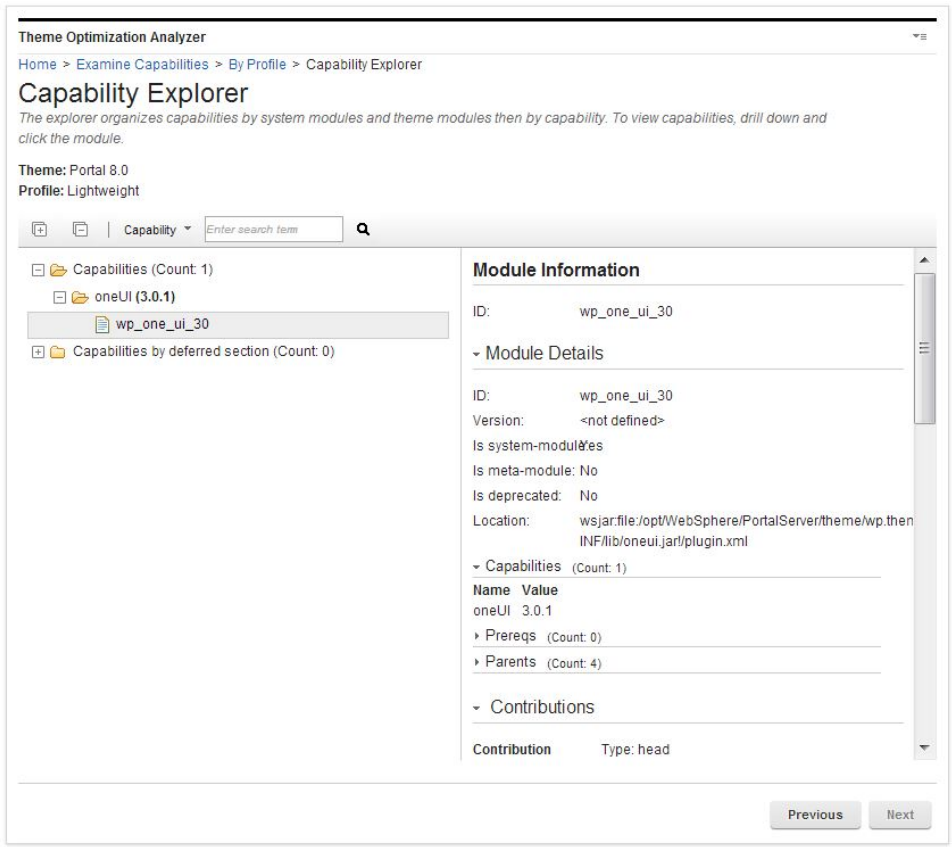
CF03



2. When you examine the capabilities that are scoped by profile, you have two root branches:
   - **Capabilities**: The root for all capabilities that are provided by modules in the none deferred section of the profile.
   - **Capabilities by a deferred section**: The root for all capabilities that are provided by modules in the deferred section of the profile.

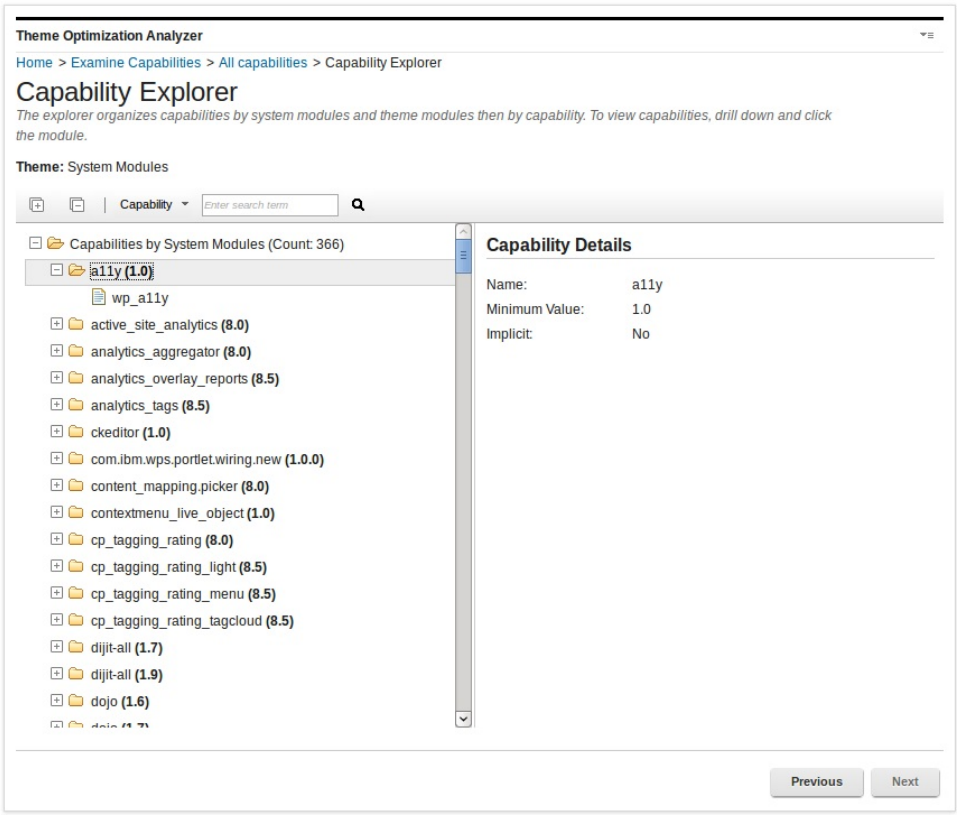
CF03

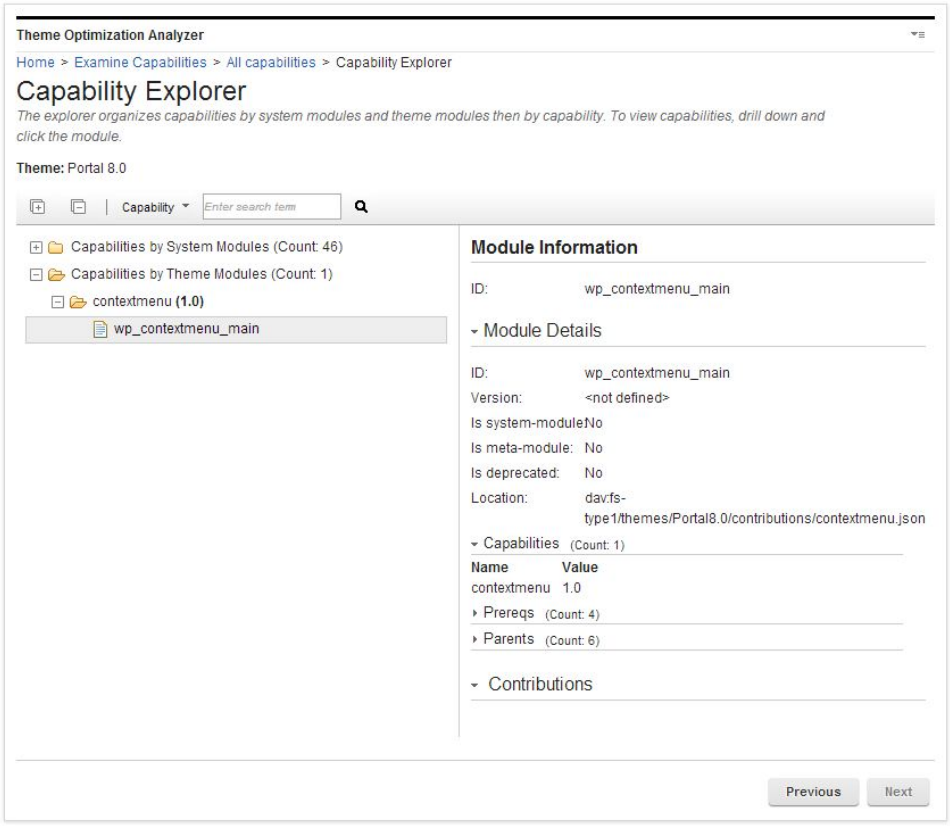

Theme Optimization Analyzer
Home > Examine Capabilities > By Profile > Capability Explorer
Capability Explorer
The explorer organizes capabilities by system modules and theme modules then by capability. To view capabilities, drill down and click the module.
Theme: Portal 8.5
Profile: Deferred
Capability
Enter search term
Capabilities (Count: 57)
Capabilities by deferred section (Count: 54)
analytics_tags (8.5)
contextmenu (1.0)
contextmenu_live_object (1.0)
federated_documents.picker (8.0)
wcm_inplaceEdit (1.0)
dijit (0.0)
dijit
dijit_19 (0.0)
dijit_form (0.0)
dijit_form_19 (0.0)
dijit_layout_basic (0.0)
dijit_layout_basic_19 (0.0)
dijit_layout_ext (0.0)
dijit_layout_ext_19 (0.0)
dijit_menu_19 (0.0)
dijit_tree (0.0)
Capability Details
Name: dijit
Minimum Value: 0.0
Implicit: Yes
Previous
Next


Theme Optimization Analyzer
Home > Examine Capabilities > By Profile > Capability Explorer
Capability Explorer
The explorer organizes capabilities by system modules and theme modules then by capability. To view capabilities, drill down and click the module.
Theme: Portal 8.0
Profile: Lightweight
Capability
Enter search term
Capabilities (Count: 1)
oneUI (3.0.1)
wp_one_ui_30
Capabilities by deferred section (Count: 0)
Module Information
ID: wp_one_ui_30
Module Details
ID: wp_one_ui_30
Version: <not defined>
Is system-module: Yes
Is meta-module: No
Is deprecated: No
Location: wsjar:file:/opt/WebSphere/PortalServer/theme/wp.then
INF/lib/oneui.jar!/plugin.xml
Capabilities (Count: 1)
Name Value
oneUI 3.0.1
Prereqs (Count: 0)
Parents (Count: 4)
Contributions
Contribution Type: head
Previous
Next

3. When you examine the system and theme capabilities, you have one or two branches:
   - **`Capabilities by System Modules`** Contain all capabilities that are provided by modules that are defined globally through a `plugin.xml`.
   - **`Capabilities by Theme Modules`** Contain all capabilities that are provided by modules that are defined within the themes capabilities folder as part of the JSON files. This branch is only shown when a theme is selected.

CF03





When you expand a tree or branch, if it is large and takes more than 30 seconds to expand, the expansion process is stopped. You must expand those branches individually.

The details view displays the details for the selected module such as:

- ID: Displays the module ID.
- Version: Displays the module version, if one was defined.
- Is system-module: Yes if the module is globally defined through a `plugin.xml` file.
- Is meta-module: Yes if the module does not contain any contributions, but instead contains only prerequisites.
- Is deprecated: Yes, if the module was deprecated.
- Location: Displays the file path for the module.
- Capabilities: Displays all capabilities for the module.
- Prereqs: Displays all prerequisites for the module.
- Parents: Displays all parent modules for the module
- Contributions: Displays all contributions by contribution and sub contribution. The resources are elements that can be clicked, which open a new window and serves the resource in the browser when clicked.

  Note: In simulation mode, the resources cannot be clicked.

Searching within the capability explorer.

Use the search bar to search the tree view and the details view for various information. The following search scopes are supported:

- **`Module Identifier`**: Search for a specific module by name. For example, `wp_client_main`.
- **`Resource`**: Search for the module that exposes a certain resource. For example, enter `master.css` to identify the module that exposes this resource. This search is useful if you find an error in a JavaScript file in your browser. You can easily find the module that this resource belongs to.
- **`Capability`**: Search for the modules that are associated with a specific capability. For example, if you want to know which module provides the `dojo` capability. Then, you know which module you must add as a prerequisite for your own module.
- **`Reference Identifiers`**: Search for which dynamic content spot or module reference is exposed by which module.

Enter a term into the search field in the toolbar and press Enter or click the **Find** icon. The found item is then highlighted for a short time with a yellow background. Pressing Enter again or clicking **Find** icon again will find the next item. If no item is found, either a pop-up dialog opens to instruct you to start the search from the beginning or the background of the search field is marked in red.

Validation reports:

Use the validation portlet to verify that your theme contains no errors. The validation report analyzes your theme and theme components for known issues and reports the number of errors, warnings, and informational messages. It also includes a detailed explanation about how to fix the errors that occur.

Home screen

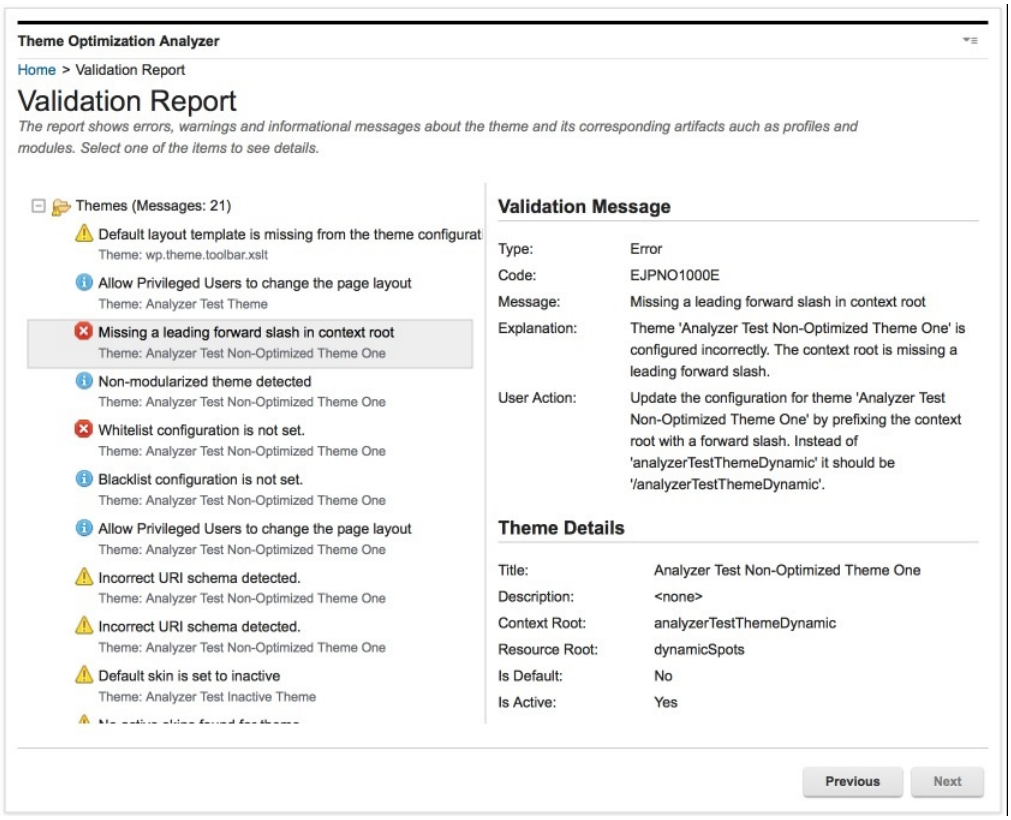
After the validation analyzer runs, the report includes a number badge. The badge shows the total number of messages in the report. The background color indicates the severity within the report. Red means that there is an error. Orange means that there is a warning. Green means that there are informational messages. If no badge is present, no messages were found.

Validation report

The expanded report shows the validation report of your system in a split view. The tree view shows the messages in categories. The details view shows information about the message that is selected in the tree view. The details view also displays related information that is important for the message. These can be theme, profile, module, or skin details.

A check mark by a category means that there are no errors or warnings in that category. In the following screen capture, the Theme Optimization Analyzer portlet Validation Report includes 21 messages. The selected message, `Missing a leading forward slash in context root`, is described in the details section. The error code is `EJPNO1000E`. The explanation is `Theme 'Analyzer Test Non-Optimized Theme One'`

is configured incorrectly. The context root is missing a leading forward slash. It also includes a User Action that tells the user how to fix the error. In this case, it is Update the configuration for theme 'Analyzer Test Non-Optimized Theme One' by prefixing the context root with a forward slash. Instead of 'analyzerTestThemeDynamic', it should be '/analyzerTestThemeDynamic'.



Possible messages

The following topics describe all of the possible messages.

"EJPNO1000E" on page 2591
Missing a leading slash in context root.

"EJPNO1001E" on page 2592
WebDAV directory names must not contain spaces.

"EJPNO1002E" on page 2592
Missing default profile.

"EJPNO1003W" on page 2592
Missing module referenced in profile.

"EJPNO1004I" on page 2592
Duplicate modules found in profile.

"EJPNO1005E" on page 2593
Circular dependency detected.

"EJPNO1007W" on page 2593
Incorrect URI schema detected.

"EJPNO1008W" on page 2593
Incorrect URI schema detected.

EJPNO1000E:

Missing a leading slash in context root.

Explanation

Your theme is configured incorrectly. The context root is missing a leading forward slash.

User action

Update the configuration for your theme by prefixing the context root with a forward slash.

EJPNO1001E:

WebDAV directory names must not contain spaces.

Explanation

Theme metadata with a key you're using for your theme indicates that there are spaces in the WebDAV directory names.

User action

Update the WebDAV directory name and remove all spaces. Update the theme metadata with the key to match the modified WebDAV directory name.

EJPNO1002E:

Missing default profile.

Explanation

The referenced profile in the theme is missing and cannot be accessed.

User action

Update your theme metadata with the key `'resourceaggregation.profile'` to point to a valid profile file.

Related tasks:

"Changing theme metadata" on page 2722
The first step of theme configuration is through theme metadata properties. Changes to the metadata are specific to a single theme, and the entries and values, therefore, can vary from theme to theme.

EJPNO1003W:

Missing module referenced in profile.

Explanation

A module you are using is referenced in the module list of your profile but does not exist within the system.

User action

Update your profile to point to the correct module instead or remove the missing module reference.

EJPNO1004I:

Duplicate modules found in profile.

Explanation

A module you are using is referenced multiple times in the module list of another profile.

User action

Update your profile and remove the duplicate entries of the module.

EJPNO1005E:

Circular dependency detected.

Explanation

A module dependency chain has been detected in a module you are using that contains a circular dependency. To prevent an endless loop the module hierarchy processing has been interrupted and your modules will potentially not work correctly.

User action

Update your module prereqs to resolve the circular dependency. The erroneous module hierarchy is named in this section.

EJPNO1007W:

Incorrect URI schema detected.

Explanation

An unsupported URI schema was detected in your theme. The theme metadata key is using a RES schema. Recommended schemas are WAR and DAV, RES is not supported.

User action

Update your theme metadata with the key to a supported schema.

Related tasks:

"Adapt the scripts that register the custom theme and skins" on page 2818
You must adapt the scripts that register the custom theme and skins that were moved from the file store.

EJPNO1008W:

Incorrect URI schema detected.

Explanation

An unsupported URI schema was detected in a skin you are using. The skin metadata key is using a RES schema. Recommended schemas are WAR and DAV, RES is not supported.

User action

Update your skin metadata with the key the tool suggests to a supported schema.

Related tasks:

"Adapt the scripts that register the custom theme and skins" on page 2818
You must adapt the scripts that register the custom theme and skins that were moved from the file store.

EJPNO1009W:

Incorrect URI detected.

Explanation

An incorrect URI was detected in your theme. This URI is using the WAR schema but contains a leading forward slash.

User action

Update your theme metadata with the correct key.

EJPNO1010W:

Incorrect URI detected.

Explanation

An incorrect URI was detected in a skin you are using. This URI is using the WAR schema but contains a leading forward slash.

User action

Update your skin metadata with the correct key.

EJPNO1011I:

Allow Privileged Users to change the page layout.

Explanation

The theme contains static content deployed in a WAR. It was detected that a value set for parameter **refreshPageLayout.template.regexp** does not match the theme. With the current setting users must have the Markup Editor role that is assigned to be able to change a layout.

User action

Open the WebSphere Integrated Solutions Console. **Open Resources** > **Resource Environment** > **Resource Environment Providers** > **WP ConfigService** > **Custom Properties**. Create or adapt a regular expression for the **refreshPageLayout.template.regexp** parameter.

Related tasks:

"Adapt the scripts that register the custom theme and skins" on page 2818
You must adapt the scripts that register the custom theme and skins that were moved from the file store.

EJPNO1012W:

Default layout template is missing from the theme configuration.

Explanation

The theme does not have a default layout template defined. This can cause pages to be created with the incorrect type.

User action

Create a theme metadata with the key **com.ibm.portal.layout.template.ref**.
xmlAccess Snippet:

```
<div class="themeOptAnalyzerFixedText"><parameter name="com.ibm.portal.layout.template.href" type=
```

Related tasks:

"Changing theme metadata" on page 2722
The first step of theme configuration is through theme metadata properties. Changes to the metadata are specific to a single theme, and the entries and values, therefore, can vary from theme to theme.

EJPNO1013E:

Whitelist configuration is not set.

Explanation

The theme contains static content deployed in a WAR. For security reasons, the WAR data source does not serve content until a special context parameter is set. This context parameter defines which files from your web module WebSphere Portal is able to serve. You must define a whitelist using a regular expression that matches the files that you want to make available.

User action

The parameters are set in the `web.xml` file of the web module that contains the static theme content. For example, serve all files that are not part of the WEB-INF folder.

```
<div class="themeOptAnalyzerFixedText"><web-app><br/>...<br/><context-param><br/>  <para
```

Related tasks:

"Add static content to your custom theme" on page 2817
Add static content for you custom theme that you created based on one of the ready-to-use themes.

EJPNO1014I:

Blacklist configuration is not set.

Explanation

The theme contains static content deployed in a WAR. With a blacklist you can remove certain entries from the set of files that are available in the whitelist. A blacklist is helpful if you want to serve a folder but not a certain file within that folder.

User action

The parameters are set in the `web.xml` file of the web module that contains the static theme content. For example, serve all files that are not part of the WEB-INF folder.

```
<div class="themeOptAnalyzerFixedText"><web-app><br/>...<br/><context-param><br/>  <para
```

Related tasks:

"Add static content to your custom theme" on page 2817
Add static content for you custom theme that you created based on one of the

ready-to-use themes.

EJPNO1015E:

Missing prereq detected.

Explanation

A missing module dependency has been detected in a module which defines a prereq that doesn't exist. The missing module dependency check has been performed within the scope of your theme and your profile.

User action

Update your module dependencies, either remove the referenced module or make the missing module available.

EJPNO1016I:

Non-modularized theme detected.

Explanation

A theme was detected as a non-modularized theme as it is missing the `resourceaggregation.profile` theme metadata key.

User action

If the theme is a non-modularized style theme it is strongly encouraged to upgrade to the modularized theme.

EJPNO1017W:

Default skins is set to inactive.

Explanation

The default skin for your theme is set to inactive. This may result in portlets not displaying until an active skin is applied.

User action

Update your skin to active or change the default skin for the theme to an active skin from the allowed-skin list.

EJPNO1018W:

No active skins found for theme.

Explanation

No active skins were detected from the list of allowed skin for your theme.

User action

Either update the list of allowed skins to include an active skin or set one of your allowed skins to active.

EJPNO1019W:

Invalid URI detected.

Explanation

An invalid URI was detected in your theme. The value of this metadata key cannot be parsed as a URI reference.

User action

Update your theme metadata with the incorrect key to a proper URI.

EJPNO1020W:

Invalid URI detected.

Explanation

An invalid URI was detected in your skin. The value of this metadata key cannot be parsed as a URI reference.

User action

Update your skin metadata with the key to a proper URI.

EJPNO1021W:

Undefined contribution hierarchy in profile.

Explanation

The profile has multiple modules defining the same dynamic content spot ID without a hierarchy or order between them.

User action

A hierarchy or order should be defined between the modules that define the same dynamic content spot ID. To define the dynamic content spot overlay, the module with the dynamic content spot ID definition that should be used needs to list all other modules with the same dynamic content spot ID definition as prereqs.

EJPNO1022E:

System plug-in collision

Explanation

A plug-in ID was detected in more than one `plugin.xml` file.

User action

Update your plug-in definitions by using unique plug-in IDs for each plug-in to avoid duplicate IDs.

EJPNO1023E:

System module collision

Explanation

A duplicate module id and version was detected.

User action

Update your module definitions by using unique module id and version combinations for each module to avoid duplicate module definitions.

EJPNO1024W:

Themelist URL used for rendering

Explanation

The theme metadata key "{1}", which is used when rendering the theme, is set to "{2}". This is a valid URL, but is pointing to the themelist WebDAV entry point. The themelist entry point is intended for performing administrative operations with themes (such as creating a new theme, deleting a theme, or modifying a theme). This themelist URL should not be used for runtime rendering operations, as it adds additional overhead that is unnecessary for non-administrative rendering operations.

User action

Update your theme metadata with the key "{0}" to point to the fs-type1 WebDAV entry point. For example: "{1}".

EJPNO1025W:

Skinlist URL used for rendering

Explanation

The skin metadata key "{1}", which is used when rendering the skin, is set to "{2}". This is a valid URL, but is pointing to the skinlist WebDAV entry point. The skinlist entry point is intended for performing administrative operations with skins (such as creating a new skin, deleting a skin, or modifying a skin). This skinlist URL should not be used for runtime rendering operations, as it adds additional overhead that is unnecessary for non-administrative rendering operations.

User action

Update your skin metadata with the key "{0}" to point to the fs-type1 WebDAV entry point. For example: `dav:fs-type1/themes/`*`custom theme`*`/skins/`*`custom skin`*`/`
.

EJPNO1026E:

Theme context root not found.

Explanation

Theme "{0}" is configured with context-root "{1}". An application with this context root cannot be found.

User action

Verify that the context-root setting of theme "{0}" matches the context root defined for the theme application and that the application is started.

Utilities:

If you need help with a theme issue, you can export your system data. Support personnel can then import and view your data as a simulated system to troubleshoot problems.

The import and export features are compatible with later versions, so if you have a recent version of IBM WebSphere Portal Express you can themes from previous versions. For example, an exported Portal 7002 theme can be viewed on a WebSphere Portal Express 8.0 system.

Export You can export your data to share with IBM support or your local support organization.

Import
You can import data from an external system to create a simulated system to aid in troubleshooting.

Note: When you import display data, the portlet enters a simulation mode. When the portlet is in simulation mode, page browsing is disabled and resource contributions cannot be clicked or viewed.

Control Center
The Control Center provides some short cuts for changing settings, which can help with theme analysis and debugging. These shortcuts are explained in detail on the Control Center page. Tools in the control center affect only the node on which you are currently working.

Invalidate Cache

Allows you to clear theme optimization-related caches so that changes you make locally to a theme's modules and profiles can be made available immediately, without restarting the server. Modify your theme then click the **Click to invalidate** link.

CF06 The framework can recognize updates to module and profile definitions of WebDAV based themes automatically. It significantly reduces the need for you to invalidate the cache manually. However, this feature does not work on `.WAR` file based themes.

For example, updating the file `dav:fs-type1/themes/Portal8.5/contributions/theme.json` through any WebDAV client, is recognized from the default pattern and causes an automatic theme invalidation.

For more information, see Configuration for resource aggregation.

CF07 **Invalidate system modules**
Allows you to invalidate system modules that are defined in `plugin.xml` files during the theme invalidation process. However, this feature skips all `.WAR` files.

For more information, see "Configuration for resource aggregation" on page 2725.

Remote Debugging
Allows you to see each module contribution as an individual request, and if there is a debug uncompressed version of the contribution it uses that resource. This is the same as setting the trace string to `com.ibm.wps.resourceaggregator.CombinerDataSource.RemoteDebug=all`.

Development Mode
Disables all caches of the theme optimization framework. It is equivalent to setting the `resourceaggregation.development.mode` property to true within the IBM WebSphere Portal Express ConfigService resource environment provider.

Reports
Allows you to see further information about the theme optimization framework, which is written to the trace log. There are three types of reports, profiles, modules, and meta-modules. Click the link in each section to turn on the reports, and click again to turn them off.

Client tracing
Client tracing enables traces for JavaScript code. The difference between client tracing and enabling tracing with the administration portlet is that the administration portlet enables Java tracing. You can add multiple strings by adding strings with a pipe. When you click add, the trace is persisted immediately. You can choose to enable the trace scope for the current browser or all users on the server. If you set the trace scope to Server, any client who accesses the server will have JavaScript traces on that server only. If you set it to current browser, you can remotely debug a server. You must have uncompressed Java resources enabled.

Advanced Utilities
Provide an option to export all titles and descriptions of all known modules in the system to a `.CSV` file.

Troubleshooting modular themes

You can debug your modules to improve performance.

A modular theme is a IBM WebSphere Portal Express theme that uses the Resource Aggregator theme optimization framework to allow the encapsulation of function into units that are called modules. You can combine those modules by defining theme profiles, and assign profiles to pages so that only the minimum necessary download data is done for a page. The download of the set of modules for any page is combined to be as efficient as possible.

Theme Optimization Analyzer Portlet

This portlet visualizes all parts of the theme optimization framework. With this portlet, you can see how the profiles are applied to specific themes or pages and see the defined resources for modules. This portlet features a validation report and the ability to export and import profiles to replicate another environment.

The portlet is included with the current version of WebSphere Portal Express. Click the **Administration menu** icon. Then, click **Portal Analysis** > **Theme Analyzer**.

Client console and debugging tools

For debugging errors in the client-side execution of JavaScript or the client-side application of cascading style sheets, use a client-side console and debugging tool. Most browsers have their own development tools that are built in, or available as an add-on. For example, the Firebug add-on can be downloaded and installed into Mozilla Firefox. If there are client-side errors in the JavaScript, they are displayed in the console of these tools. By default, the resources are still being compressed and aggregated so it can be difficult to identify the exact resource that causes the error.

"Turning off aggregation and compression in client-side debug mode"
Turning on debug mode disables compression and makes modules easier to debug.

"Reloading the profile and module in development mode without caching" on page 2602
When you are debugging an issue, sometimes you must actively update the profile or module definitions.

"Debugging your module systematically" on page 2602
Use a pattern to determine an issue within your custom modules. This pattern minimizes the time that is required to investigate and debug.

"Verifying JSON file Syntax" on page 2604
If a JSON file is not formatted correctly it cannot be processed by the server, and is not loaded into your theme.

"Syntax in modules and profile definitions" on page 2605
If your theme is not behaving as it should, it could be because there are syntax problems within the module or profile files.

Related information:

Theme optimization analyzer on the IBM Solutions Catalog

Turning off aggregation and compression in client-side debug mode:

Turning on debug mode disables compression and makes modules easier to debug.

About this task

In normal operation, the JavaScript and CSS modules in a profile to render a page view are fetched by the ResourceAggregator by building an aggregated request for all the contributions of the same type. It usually includes all the JavaScript, and all the css style definitions. It downloads them together. The JavaScript and CSS is compressed to reduce the size, which makes them difficult to read and debug. By turning on debug mode, the decompressed versions of the resources are loaded if they are defined, and they are loaded individually instead of being aggregated.

Note: If you are using Internet Explorer 7, 8, or 9, the resources continue to be aggregated into one request in debug mode.

To activate debug mode, you can use the Theme Analyzer, or manually enable the feature by turning on a trace string so that debugging is enabled for all users. You can also set a specific cookie so that debugging is only enabled for that user's cookie. In the Theme Analyzer, click **UtilitiesControl Center**. In the Remote

Debugging section, you can turn debugging on and off for all users or just for an individual client.

Procedure

To turn on debug mode for all users, all themes, and all pages, system wide, enable tracing with the following trace string.

```
com.ibm.wps.resourceaggregator.CombinerDataSource.RemoteDebug=all
```

You can set the string in the WebSphere Integrated Solutions Console for the current running instance. Or, you can save it to the persistent server runtime definition. You can also set this trace string with the Enable tracing portlet.

Results

When a user sets a cookie that is named **com.ibm.portal.resourceaggregator.client.debug.mode** to `true`, debug versions of module contributions are loaded if they are defined. Modules are loaded without using separate links and script tags. All resources are downloaded as a combined unit in that case.

Reloading the profile and module in development mode without caching:

When you are debugging an issue, sometimes you must actively update the profile or module definitions.

About this task

You can invalidate the cache using the Theme Analyzer to update the profile or module definitions. Or, you can enable development mode, which disables all caches in the system and refreshes all profile and module information for every request.

Procedure

1. Open the WebSphere Integrated Solutions Console.
2. Select **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Select the **WP ConfigService** resource environment provider.
4. Click **Custom properties**.
5. Change the **resourceaggregation.development.mode** entry to **true**.
6. Save the changes.
7. Restart the WebSphere Portal Express server.

Debugging your module systematically:

Use a pattern to determine an issue within your custom modules. This pattern minimizes the time that is required to investigate and debug.

Procedure

1. Use the console in the client. Most browsers have their own development tools built-in. If there are client-side errors in the JavaScript, they are displayed in the console of these tools. The resources are still being aggregated so it can be difficult to identify the exact resource that causes the error.

2. Turn on tracing. With tracing, you can see the specific resources that are aggregated on the page and you can see the location of where an error in the console originates. You can follow the code flow and track the expected values within the code.
3. Investigate the logs. If the error is not apparent in the client-side console, then investigating the WebSphere Portal Express system logs is another likely place to track down an issue. These logs display any server-side errors and tracing that is enabled.
   a. Open the WebSphere Portal Express logs in the following locations, *wp_profile_root*//logs/WebSphere_Portal/SystemOut.log and *wp_profile_root*/logs/WebSphere_Portal/SystemErr.log.
   b. You can enable more tracing through the Enable Tracing portlet in the Administration section of WebSphere Portal Express. Click the **Administration menu** icon. Then, click **Portal Analysis** > **Enable Tracing**.

   For more information, see *Logging and tracing*.
4. Add logging to your resources. The WebSphere Portal Express code logs throughout the product for serviceability. However, your custom code might need more logging that is inserted to track down an issue. For client-side code, use a console log statement. A typical console log statement for displaying a value with a label is `console.debug("example value: "+value);`. Using this code displays a string value. To display the object in the console to see the values that are stored in the object, you can use

   ```
   console.log("the ibmCfg object:");
   console.debug(ibmCfg);
   ```

   The first line tracks the instance of the object. The second line adds the object to the console. Then, you can use the console to see the attributes and their values.
5. Ensure that the resources are loaded on the page. If a particular feature of the page is not visible or does not seem to function properly, sometimes the resources are not on the page. To verify, you can view the requests on the page and see the individual JavaScript or CSS. Ensure the files that are needed for the function are included on the page. If resources are missing, edit the profile to include the necessary modules.
6. Investigate with the Theme Optimization Analyzer. The theme optimization analyzer portlet can display many attributes of the theme, pages, profiles, and modules. You can investigate a module and view the details for visibility that is based on device class and contributions of resources. Each resource is listed with its path in the server. So, you can use this information to determine whether it is being aggregated on the page or to add more logging.
7. Export XML definition for resources. After you investigate the issue, modifying the resource attributes is the next solution. Resources that would require a change to their XML definition are the theme, page, client, or device class. To export the XML definition of a resource, you can use XMLAccess. This is a command-line tool that is at *wp_profile_root*/PortalServer/bin/xmlaccess.bat in Windows and *wp_profile_root*/PortalServer/bin/xmlaccess.sh in Linux. You can use the following scripts to export a specific resource that is related to the theme framework but more examples are located here *PortalServer_root*/doc/xml-samples/.
8. Export all theme definitions to modify the advanced features such as the metadata, resource root, default layout, or theme template definition. You can also modify other attributes through this definition.

```
<?xml version="1.0" encoding="UTF-8"?>
<request
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="PortalConfi
type="export" create-oids="true">
<portal action="locate">
<theme action="export" active="true" objectid="*"></theme>
</portal>
</request>
```

9. Export page definition to modify features specific to the theme such as theme template override, dynamic content spot overrides, or device class filtering.

```
<?xml version="1.0" encoding="UTF-8"?>
<request
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="PortalConfi
type="export" create-oids="true">
<portal action="locate">
<content-node action="export" uniquename="ibm.portal.SamplePage"/>
</portal>
</request>
```

10. Export all defined clients to modify the device class assigned. You can view this information in the Manage Clients portlets as well.

```
<?xml version="1.0" encoding="UTF-8"?>
<request
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="PortalConfi
type="export" create-oids="true">
<portal action="locate">
<client action="export" uniquename="*"></client>
</portal>
</request>
```

11. Export all defined device classes to view the available groups and change the title or ID.

```
<?xml version="1.0" encoding="UTF-8"?>
<request type="export" create-oids="true"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="PortalConfig_7.0.0_2.xsd">
<portal action="locate">
<device-class action="export" uniquename="*" />
</portal>
</request>
```

Verifying JSON file Syntax:

If a JSON file is not formatted correctly it cannot be processed by the server, and is not loaded into your theme.

A correctly formatted JSON file is both syntactically correct JSON. It also contains only syntactically correct WebSphere Portal Express Theme Optimization Framework Module definitions, theme profile definitions, or JSON menu Framework menu definitions.

To avoid problems, verify both the basic JSON syntax, and the specific WebSphere Portal Express syntax of your JSON file before you deploy it to your WebSphere Portal Express environment.

To validate the basic JSON syntax, use a tool such as JSONLint. For more information, see the JSONLint link.WebSphere Portal Express was extended to allow comments in the JSON file. A strict JSON validator such as JSONLint flags these comments as incorrect syntax. Remove any such comments before you pass the JSON to a validating parser.

To validate the WebSphere Portal Express specific definition syntax, you can use a JSON schema validator, which takes as input two JSON files. It compares the input to be validated, and a schema against which to validate. JSON schema files are available for module definitions, profiles, and menu definitions.

Related information:

JSONLint

Syntax in modules and profile definitions:

If your theme is not behaving as it should, it could be because there are syntax problems within the module or profile files.

Several types of problems could occur, including invalid elements or attributes, invalid element child relationships, invalid attribute values, or invalid attribute value combinations. Some elements and attributes are required, and some must not appear more than once. For more information, see *Theme modules*.

The modules and profiles are validated at runtime when the theme is first loaded. The runtime checks are sometimes able to detect problems that can slip through the schema validations. Perform the schema validations before deploying the JSON files. When syntax errors are encountered at runtime, they are written as warnings in the `SystemOut.log` file. These warnings begin with message identifier prefix `EJPNK` so searching for that string can help you locate them quickly. For example

```
EJPNK0060W: The attribute "version" in module test_analyzer_module_invalidsyntax_1 is not a valid
```

All syntax warnings are also included in the Theme Optimization Analyzer Validation Report.

Specify profiles

You can specify the profiles that you want to use two different ways depending on whether you are an administrator or a user. Profiles help define which modules are used to render a page.

Specify the profile with a module if you have administration access. Otherwise, use the user interface.

Profiles define a set of modules to be loaded by the theme optimization framework during page rendering. An administrator can use the admin portlets or XML access to set a profile. A content author can also set a profile.

CF03 You can set dependencies on features in portlets and profiles. The features are automatically loaded onto the page in an aggregated way. Your profile does not need to contain more modules than are required by the theme. Modules that are only required by a certain portlet can be loaded by that portlet.

"Included profiles" on page 2606
Portal includes profiles that contain modules that change how your portal site is rendered.

"Specify profiles with metadata" on page 2607
If you are an administrator, you can define which modules are used to render a page. Profiles specify which modules are loaded on a page or whether they are deferred to after a page loads.

"Specifying profiles with the user interface" on page 2607
You can change which sets of modules are used to render a page quickly through the user interface.

"Changing the theme default profile" on page 2608
You can change the profile for the theme or a specific page to define the modules loaded.

Included profiles:

Portal includes profiles that contain modules that change how your portal site is rendered.

Portal 8.5 default theme

Basic Content

This profile contains everything that is required for viewing and editing content and capabilities such as tagging and rating, person card, and social rendering.

Basic Content with Dojo

This profile contains everything that is required for viewing and editing content and capabilities such as tagging and rating, person card, and social rendering. This profile includes Dojo.

Deferred with Dojo

This profile has the full set of modules for the theme, but defers loading most of these modules unless needed. This profile includes Dojo. This profile balances function and performance.

Deferred (Default)

This profile has the full set of modules for the theme, but defers loading most of these modules unless needed. This profile balances function and performance.

Lightweight with Dojo

This profile has modules necessary for viewing pages with portlets. Modules necessary for iWidgets and modules that are needed by users to change pages are not included. This profile includes Dojo.

Lightweight

This profile has modules necessary for viewing pages with portlets. Modules necessary for iWidgets and modules that are needed by users to change pages are not included. Assigning the lightweight profile to a page does not imply that the page can be no longer edited. The theme of the page contains modules that are needed for rendering. The lightweight profile contains the `wp_toolbar_host_view` module. The `wp_toolbar_host_view` module does not contain any inline editing artifacts or Drag and Drop artifacts on the main page. However, you can still edit the page using the toolbar, but the toolbar is not loaded into the main page. It is in a different iframe. It is in the view frame, while the toolbar lives in another iframe. If you want make sure the page can no longer be edited, you can assign the user role so the user does not have access, or set the page metadata to disable edit mode for a page. Do set the page metadata, set the `theme.disable.edit.mode = true`.

Web Dock

This profile has the set of modules to be used for the Web Dock applications.

CF03 Web Dock was removed in combined cumulative fix 03. It is not required because the Web Dock capability is automatically downloaded on

demand if you set up the Resource Aggregation for portlet feature in your theme. Then, you must add the module to an existing profile.

Specify profiles with metadata:

If you are an administrator, you can define which modules are used to render a page. Profiles specify which modules are loaded on a page or whether they are deferred to after a page loads.

The list of modules is determined by:

- If a profile path exists, it is obtained from the **resourceaggregation.profile** page metadata.
- If no profile is set on the page, or the theme does not determine the profile, then the default profile is defined by the **resourceaggregation.profiles.default** theme metadata.

The list of available profiles within a theme is determined by loading all files in the folder defined by the **resourceaggregation.profiles.default** theme metadata. If that metadata is not defined, the `profiles/` default folder is scanned for profiles.

Important: Only `profile.json` files are allowed in these folders.

The profile format

The profile files must be valid JSON files. The following sample shows the properties:

```
{
   "moduleIDs" : ["moduleID_1", "moduleID_2", "moduleID_3"],
   "titles": [{
        "value": "title_en",
        "lang": "en"
             },
            {
        "value": "title_de",
        "lang": "de"
             }],

   "descriptions": [{
        "value": "desc_en",
        "lang": "en"
             },
            {
        "value": "desc_de",
        "lang": "de"
             }]
}
```

Specifying profiles with the user interface:

You can change which sets of modules are used to render a page quickly through the user interface.

Procedure

1. Turn on Edit Mode.
2. Click **Page** > **Edit Page Properties**.
3. In the Manage Page Properties window, click the **Advanced** tab.
4. In the Theme settings section, select the profile you want to use in the list of profiles.

5. Select from the visible profiles and click **Save**.

Changing the theme default profile:

You can change the profile for the theme or a specific page to define the modules loaded.

WebDAV:
Procedure

1. Connect your WebDAV client to `http://<server>:<portal>/wps/mycontenthandler/dav/themelist/`.
2. Navigate to the folder for your theme and copy the `metadata.properties` file to your local drive.
3. Edit the local copy of the file and modify it to point to the profile desired. For Example:

   ```
   com.ibm.portal.themetype=CSA2
   resourceaggregation.profile=profiles/profile_lightweight.json
   ```

4. Copy the local copy of the `metadata.properties` file back into the folder for your theme in the `themelist` folder.

XMLAccess:
Procedure

1. Export the theme. You can export all themes defined for WebSphere Portal Express using the following script, or insert the specific theme object ID you want to export:

   ```
   <?xml version="1.0" encoding="UTF-8"?>
   <request
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd" type="export"
   create-oids="true">
       <portal action="locate">
          <theme action="export" objectid="*" />
       </portal>
   </request>
   ```

2. Modify the parameter **resourceaggregation.profile** to point to the new profile. For Example:

```
<parameter name="resourceaggregation.profile" type="string" update="set"><![CDATA[profiles/profile_lightweight.json]]></parameter>
```

3. Import the XML file using the command line or **Import XML**.
4. Restart WebSphere Portal Express.

Modules that are provided with the modularized theme

WebSphere Portal Express provides a set of ready-to-use modules.

Use these dependency guidelines when you add and remove modules from your profile:

- If you add a module that requires another module that is in not in the profile, the automatic dependency injection adds the required module at run time.
- If you remove a module that is required by another module that is in the profile, the automatic dependency injection adds the module at run time, even if it is not listed in the profile.
- If you remove a module that the theme requires, but is not required by any other modules in the profile, the module is removed and the theme might break.

Theme modules provided with the Portal theme

The following lists describe the modules that are included with the Portal theme. Information about each module includes the module ID and the location in the theme of that module and some details about the module.

85 Theme

These modules are used by the WebSphere Portal Express 8.5 theme. For more information, see the module descriptions.

The `plugin.xml` file location varies and is documented in the module description.

Table 396. List of 85 Theme modules and prerequisites.

Module	Description
wp_theme_portal_85	The Portal 85 theme CSS. Location: `dav:fs-type1:/themes/portal8.5/css`
wp_theme_edit	Adds the ability to go into page edit mode. Location: `dav:fs-type1:/themes/portal8.5/js`
wp_theme_menus	The menu framework that was introduced in 7002. Location: `dav:fs-type1:/themes/portal8.5/js`
wp_portlet_css	Earlier portlet CSS support. Location: `dav:fs-type1:/common-resources/ibm/css/portal`
wp_legacy_layouts	Earlier 7.0 static page layout CSS. Location: `dav:fs-type1:/common-resources/ibm/css/portal`
wp_layout_windowstates	Maximize or Minimize portlet support that is implemented as a server-side data source.
wp_portal	Supplies JavaScript global configuration objects for use by other features; URLs, locale information, and user information. Implemented as a server-side data source.

Table 396. List of 85 Theme modules and prerequisites. (continued)

Module	Description
wp_liveobject_framework	Live object framework provides the feature of adding a special handler to any class selector on a tag. For example, if you have a span tag and its class contains vCard, then this framework makes this markup live. Then, when you hover on the designated text, the following hover text is shown, **click here for person card**. When you click the hover text, it shows the person card. The person card feature is available out-of-box, along with other requirements. Developers can extend this framework and add their own handlers. Location: *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/plugin.xml
wp_oob_sample_styles	Styles for default web content samples. Location: *PortalServer_root*/pzn.ext/wp.templating.wcm/shared/app/wp.wcm.templating.jar
wp_theme_skin_region	Provides accessibility support Location: dav:fs-type1:/themes/portal8.5/js
wp_theme_high_contrast	Provides accessibility support when you use high contrast. Location: dav:fs-type1:/themes/portal8.5/js
wp_custom_page_style	The view counterpart of the Edit toolbar shelf that sets the style for a specific page. This module reads the style for a page and makes sure that the corresponding CSS is loaded into the page.
getting_started_module	The getting started module is a pre-defined module that you can use as starting place to quickly inject your own resources into the current theme. For more information, see *Simple modules*. Location: dav:fs-type1:/themes/Portal8.5/modules/getting_started_module
wp_liveobject_framework_core	Live Text Framework core part that provides parsing capability. It parses page for a class selector and returns the retrieved nodes to class handler. Location: *PortalServer_root*/ui/wp.tagging.liveobject/semTagEar/Live_Object_Framework.ear/liveobjects.war/WEB-INF/plugin.xml

Table 396. List of 85 Theme modules and prerequisites. (continued)

Module	Description
wp_openajaxhub	The key feature of OpenAjax Hub 2.0 is its publish/subscribe engine that includes a Managed Hub mechanism that allows a host application to isolate untrusted components into secure sandboxes. Location: *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/openajaxhub.jar

Simple menus

These modules are used by the toolbar and theme menus. For more information, see the module descriptions. The `plugin.xml` file location varies and is documented in the module description.

Table 397. List of menu modules and prerequisites

Module	Description
wp_simple_contextmenu_main	Gathers all the modules that are required to make up the simple menus. Location: dav:fs-type1:/themes/Portal8.5/contributions/simple_contextmenu.json
wp_simple_contextmenu_css	Provides CSS styling for the simple menus Location: dav:fs-type1/themes/Portal8.5/contributions/simple_contextmenu.json
wp_simple_contextmenu_js	Provides support to display a simple pop-up menu. Location: *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/simple_contextmenu.jar
wp_simple_contextmenu_ext	Provides extensions that plug into the simple menu, such as badge support. Location: *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/simple_contextmenu.jar
wp_simple_contextmenu_templates	Provides templates that are used to render the simple menu. Location: dav:fs-type1/themes/Portal8.5/contributions/simple_contextmenu.json

Menus

These modules are used by theme menus in your Portal, such as in place edit. These menus are also called Component Action menus or CAM. For more information, see the module descriptions. The `plugin.xml` file location varies and is documented in the module description.

Table 398. List of menu modules and prerequisites

Module	Description
wp_contextmenu_main	Gathers all the modules that are required to make up the complete **Component Action** menu. Location: `dav:fs-type1:/themes/Portal8.5/contributions/contextmenu.json`
wp_contextmenu_css	Provides CSS styling for the **Component Action** menu. Location: `dav:fs-type1/themes/Portal8.5/contributions/contextmenu.json`
wp_contextmenu_js	Provides support to display an open action menu for page components, for example a portlet or content item. Location: *PortalServer_root*`/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/contextmenu.jar`
wp_contextmenu_templates	Provides templates that are used to render the **Component Action** menu in specific contexts, for example, a menu for a component or an inline edit menu. Location: `dav:fs-type1/themes/Portal8.5/contributions/contextmenu.json`
wp_contextmenu_config_lof	Provides the configuration for the Live Object Framework service to handle **Component Action** menu instances on a page. Location: *PortalServer_root*`/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/contextmenu.jar`
wp_contextmenu_live_object	Provides Live Object Framework service to handle **Component Action** menu instances on a page. Location: *PortalServer_root*`/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/contextmenu.jar`
wp_skin_cam	Allows the Component Action Menu to be opened by clicking an icon in the portlet skin. Location: *PortalServer_root*`/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/skincomponentactionmenu.jar`

Dynamic Content Spots

These modules provide mappings for dynamic content spots in the theme.

Table 399. These modules provide mappings for dynamic content spots in the theme.

Module	Description
wp_dynamicContentSpots_85	Defines all dynamic content spots for the 8.5 theme. You can overlay any dynamic content spots through other modules. Location: *PortalServer_root*/theme/wp.theme.themes/default85/installedApps/DefaultTheme85.ear/DefaultTheme85.war/WEB-INF/plugin.xml
wp_dynamicContentSpots_toolbar85	Defines all dynamic content spots for the 8.5 toolbar theme. You can overlay any dynamic content spot through other modules. Location: *PortalServer_root*/toolbar/wp.toolbar.themes/toolbar85/installedApps/ToolbarTheme85.ear/ToolbarTheme85.war/WEB-INF/plugin.xml

Toolbar

These modules provide resources for the toolbar.

Table 400. These modules provide resources for the toolbar.

Module	Description
wp_a11y	Accessibility utility functions APIs. Location: *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/a11y.jar
wp_admin_edit	Module providing theme artifacts that are related to the admin area. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/plugin.xml
wp_hiddencontent	Module defining CSS to show or hide the hidden content items in the hidden layout container. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/hiddencontent.jar

Table 400. These modules provide resources for the toolbar. (continued)

Module	Description
wp_movecontrols	This module provides a JavaScript API, which allows for moving layout controls from one layout container to another within a page layout. Location: *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/movecontrols.jar
wp_portlet_applications	Support module for Applications portlets. Location: *wp_profile_root*/installedApps/*cell*/PA_Applications.ear/wp.portlet.applic.war/WEB-INF/plugin.xml
wp_portlet_changelayout	Support module for Change Layout portlet. This support module shows the orphaned portlets section. Location: *wp_profile_root*/installedApps/*cell*/PA_Orphaned.ear/wp.portlet.change.war/WEB-INF/plugin.xml
wp_portlet_newcontent	Support module for New Content portlet. Location: *wp_profile_root*/installedApps/*cell*/PA_com.ibm.wps.portle.ear/wp.portlet.conten.war/WEB-INF/plugin.xml
wp_portlet_newpage	Support module for New Page portlet. Location: *wp_profile_root*/installedApps/*cell*/PA_New_Page.ear/wp.portlet.newpag.war/WEB-INF/plugin.xml
wp_portlet_overview	Support module for Overview portlet. Location: *wp_profile_root*/installedApps/*cell*/PA_com.ibm.wps.portle.ear/wp.portlet.conten.war/WEB-INF/plugin.xml
wp_portlet_projects	Support module for Project portlet. Location: *wp_profile_root*/installedApps/*cell*/PA_com.ibm.wps.portle.ear/wp.portlet.conten.war/WEB-INF/plugin.xml
wp_portlet_sitemap	Support module for Sitemap portlet. Location: *wp_profile_root*/installedApps/*cell*/PA_com.ibm.wps.portle.ear/wp.portlet.conten.war/WEB-INF/plugin.xml
wp_portlet_style	Support for mode styles portlet. Location: *wp_profile_root*/installedApps/*cell*/PA_Styles.ear/wp.portlet.styles.war/WEB-INF/plugin.xml

Table 400. These modules provide resources for the toolbar. (continued)

Module	Description
wp_portlet_vanityurl	Support for the Vanity URL portlet. Location: *wp_profile_root*/installedApps/*cell*/PA_VanityUrl.ear/wp.portlet.vanity.war/WEB-INF/plugin.xml
wp_portlet_wiring	Dependencies of the wiring portlet Location: *wp_profile_root*/installedApps/*cell*/PA_Wiring.ear/wp.portlet.wiring.war/WEB-INF/plugin.xml
wp_portlet_wiring_resources	Resources that are provided by the portlet. Location: *wp_profile_root*/installedApps/*cell*/PA_Wiring.ear/wp.portlet.wiring.war/WEB-INF/plugin.xml
wp_state_page	Module that shows the page-specific public render parameters to the client and also provides JavaScript APIs that allow for reading and writing these parameters. Location: *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/state.jar!/plugin.xml
wp_state_page_modes	Module that annotates the HTML body element of the DOM with microformats that reflect the page modes that are currently active. The supported page modes are page edit mode, or microformat edit-mode, page information mode, or microformat info-mode," and page help mode or microformat "help-mode". Unlike portlet modes, multiple page modes can be active at the same time. Location: *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/state.jar!/plugin.xml
wp_theme_utils	Module providing a set of JavaScript utility APIs, which can be used to work with the DOM of the different page parts, which are potentially running in different iframes. Location: *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/themeutils.jar

Table 400. These modules provide resources for the toolbar. (continued)

Module	Description
wp_toolbar85	Module providing a dynamic content spot that allows for embedding the main window of the site toolbar into a theme. The id for the dynamic content spot is `85toolbar`. Location: *PortalServer_root*`/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/toolbar.jar`
wp_toolbar85_dynamic	Module that displays the main control panel of the site toolbar without relying on a dynamic content spot. Location: *PortalServer_root*`/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/toolbar.jar`
wp_toolbar_actionbar	Module providing a dynamic content spot that renders the main action bar of the site toolbar in the 8.5 theme. The action bar provides access to the site toolbar navigation, project menu, and more menu and also allows for activating page edit mode and page information mode. The id for the dynamic content spot is `85toolbar_actionbar`. Location: *PortalServer_root*`/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/actionbar.jar`
wp_toolbar_changepagelayout	Module providing the Change Page Layout function for menus. Location: *PortalServer_root*`/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/menuactions.jar`
wp_toolbar_common	Module providing common resources such as CSS styles, images, and JavaScript for the site toolbar. The resources are supposed to be used in the toolbar theme or by portlets that run in the site toolbar. Location: *PortalServer_root*`/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/common.jar`

Table 400. These modules provide resources for the toolbar. (continued)

Module	Description
wp_toolbar_common_wcm	Module providing common WCM-related resources such as CSS styles, images, and JavaScript for the site toolbar. The resources are supposed to be used in the toolbar theme or by portlets that run in the site toolbar. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/common.jar
wp_toolbar_contextmenu_js	A theme module that defines a simple JavaScript API for creating menus and managing their lifecycle. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/contextmenu.jar
wp_toolbar_controlactions	Module providing the resources that are common to all layout control and portlet-specific menu contributions. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/menuactions.jar
wp_toolbar_createpage	Module providing the "Create Page" function for menus. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/menuactions.jar
wp_toolbar_deletecontrol	Module providing the "Delete Control" function for portlet menus. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/menuactions.jar
wp_toolbar_deletepage	Module providing the Delete Page function for menus. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/menuactions.jar
wp_toolbar_editpageproperties	Module providing the "Edit Page Properties" function for menus. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/menuactions.jar

Table 400. These modules provide resources for the toolbar. (continued)

Module	Description
wp_toolbar_hiddencontent	Module providing the functions for showing and hiding the layout container that contains the portlets and content items that are currently hidden. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/menuactions.jar
wp_toolbar_hidecontrol	Module providing the functions for moving portlets or content items into the hidden layout container of the page. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/menuactions.jar
wp_toolbar_host	Meta module that groups all theme modules that are needed to make a page interoperable with the site toolbar. Add this module to the theme profile of your page if you want to edit the page by using the site toolbar. This module contains all contributions for both, view mode and edit mode. This module must be added to the non-deferred section of the theme profile. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/toolbar.jar
wp_toolbar_host_dynamic	Meta module that groups all theme modules that are needed to make a page interoperable with the site toolbar. Add this module to the theme profile of your page if you want to edit the page by using the site toolbar. This module contains all contributions for both, view mode and edit mode. This module must be added to the non-deferred section of the theme profile. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/toolbar.jar

Table 400. These modules provide resources for the toolbar. (continued)

Module	Description
wp_toolbar_host_dynamic_view	Module that groups all theme modules that are needed to make a page interoperable with the site toolbar. Add this module to the theme profile of your page if you want to edit the page by using the site toolbar. This module contains all contributions that are needed for view mode. This module must be added to the non-deferred section of the theme profile. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/toolbar.jar
wp_toolbar_host_edit	Module that groups all theme modules that are needed to make a page interoperable with the site toolbar. Add this module to the theme profile of your page if you want to edit the page by using the site toolbar. This module contains all contributions that are needed for edit mode. This module can be added to the deferred section of the theme profile. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/toolbar.jar
wp_toolbar_host_view	Module that groups all theme modules that are needed to make a page interoperable with the site toolbar. Add this module to the theme profile of your page if you want to edit the page by using the site toolbar. This module contains all contributions that are needed for view mode. This module must be added to the non-deferred section of the theme profile. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/toolbar.jar
wp_toolbar_logo	Module providing a dynamic content spot to embed the toolbar logo into a theme. The identifier of the dynamic content spot is 85toolbar_logo. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/logo.jar

Table 400. These modules provide resources for the toolbar. (continued)

Module	Description
wp_toolbar_menuactions	Module providing the resources that are common to all toolbar-specific menu contributions. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/menuactions.jar
wp_toolbar_moremenu	Module defining the more menu of the toolbar. This module provides a content spot to embed the more menu into a theme. The ID of the dynamic content spot is 85toolbar_moremenu. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/moremenu.jar
wp_toolbar_movecontrol	Module providing the Move Control functions for portlet menus. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/menuactions.jar
wp_toolbar_movepage	Module providing the Move Page function for menus. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/menuactions.jar
wp_toolbar_pageactions	Module providing the resources that are common to all page-specific menu contributions. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/menuactions.jar
wp_toolbar_projectmenu	Module for integrating the project menu into the toolbar. This module provides a content spot to embed the project menu into a theme. The ID of the dynamic content spot is 85toolbar_projectmenu. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/projectmenu.jar

Table 400. These modules provide resources for the toolbar. (continued)

Module	Description
wp_toolbar_sitepreview	Module providing the "Preview as User" and "Preview as Unauthenticated" functions for menus. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/menuactions.jar
wp_toolbar_sitepreview_contentspot	Module providing the dynamic content spot to embed the site preview function into the theme. The ID of the dynamic content spot is 85toolbar_preview. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/preview.jar
wp_toolbar_tab	Meta module that groups all theme modules that are needed to add a page to the site toolbar. Add this module to the theme profile of your page if your page is to run in the site toolbar. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/toolbar.jar
wp_toolbar_utils	Module providing toolbar utility functions for opening or closing the toolbar, engaging edit mode, or opening a toolbar tab. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/toolbar.jar
wp_toolbar_validator	Module to validate the state of the toolbar tab frame. This module is needed for handling context switches, such as session timeouts, correctly. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/toolbar.jar
wp_toolbar_viewframe_validator	Module to validate the state of the view frame. This module is needed for handling context switches, such as session timeouts, correctly. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/toolbar.jar

Table 400. These modules provide resources for the toolbar. (continued)

Module	Description
wp_toolbar_wiring	Module providing the functions for managing communication endpoints and portlet wires. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/menuactions.jar
wp_photon_iframe	XSLT transformation for iFrame transport. Location: *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/photon.jar
wp_photon_promisor	A promiser implementation. Location: *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/photon.jar
wp_photon_xslt	XML and XLST utility functions Location: *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/photon.jar
wp_toolbar_ghost	Module that groups all theme modules that are needed to make a page interoperable with the site toolbar without rendering it. This module must be added to the non-deferred section of the theme profile. Location: *PortalServer_root*/toolbar/wp.toolbar.modules/webapp/installedApps/ToolbarModules.ear/ToolbarModules.war/WEB-INF/lib/toolbar.jar
wp_draft_page_ribbon	Adds `Draft Page` in text that appears along the sides of a page that has a draft in the current project. Location: *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/draftribbon.jar

Drag-and-drop

These modules provide the toolbar drag-and-drop function.

Location: *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/dnd.jar

Table 401. These modules provide the toolbar drag-and-drop function.

Module	Description
wp_dnd_css	This module provides the CSS that is used to show drag sources and drop zones.
wp_dnd_main	This module parses the markup of a page layout to convert the layout containers into valid HTML 5 drop zones.
wp_dnd_util	This module provides JavaScript utility APIs for implementing drag-and-drop features.

IBM Web Content Manager

These modules provide Web Content Manager functions.

Table 402. List of Web Content Manager modules and prerequisites

Module	Description
wcm_config	Web Content Manager Config is a resource that is intended to be used by Web Content Manager theme modules. Location: *PortalServer_root*/wcm/prereq.wcm/wcm/shared/app/ilwwcm-services-impl.jar
wcm_inplaceEdit	In-place editing enables users with edit access to a content item to edit fields of that item from within the web page itself instead of using the authoring portlet. This feature is available when you display content with a web content viewer portlet. Location: *wp_profile_root*/installedApps/*cell*/wcm.ear/wcm-inplaceEdit.war/WEB-INF/plugin.xml

Content Mapping

Description: Provides content mapping support.

PortalServer_root/theme/wp.theme.modules/contentmapping/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/contentmapping.jar

Table 403. List of Content Mapping modules and prerequisites.

Module	Description
wp_content_mapping_picker	Provides the Content Mapping Picker dialog that allows one to select a piece of content from Web Content Manager.
wp_content_targeting_cam	Provides resources that are required for the Content Targeting dialog that is started from the **Component Action** menu. Location: *wp_profile_root*/installedApps/*cell*/PA_wp.pzn.ui.actions.ear/wp.pzn.ui.actions.war/WEB-INF/plugin.xml

Federation

Description: Provides federated document picker support.

PortalServer_root/theme/wp.theme.modules/federation/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/federation.jar

Table 404. List of Federation modules and prerequisites.

Module
wp_federated_documents_picker

Search

These modules provide JavaScript code for the search box widget and provide a JSP to generate the search box markup that can be started as a dynamic content spot.

The plugin.xml file location is *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/search.jar

Table 405. List of Search modules and prerequisites.

Module	Description
wp_search	Search widget
wp_searchbar	Lightweight inline search bar that redirects to the search page to show results. Location: *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/search.jar

Analytics

These modules provide Analytics support.

PortalServer_root/theme/wp.theme.modules/asa/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/asa.jar

Table 406. List of Analytics modules and prerequisites.

Module	Description
wp_analytics	Meta module that provides all analytics features. Use this module if you want to enable all analytics features including the aggregator, analytics tags and site promotions, and the overlay reports.
wp_analytics_aggregator	Inserts the reference to the analytics aggregator and its dependencies into the page.
wp_analytics_bootstrap	Internal module that contains the bootstrap code that is needed for analytics. Must not be directly referenced in theme profiles.
wp_analytics_overlay_reports	Public module that provides the analytics overlay reports for portlets and pages.

Table 406. List of Analytics modules and prerequisites. (continued)

Module	Description
wp_analytics_tags	Public module that provides the analytics tag and site promotion functions. This module also provides the dynamic content spots that produce the analytics micro-formats.
wp_analytics_tags_dialog	Public module that provides the dialog for working with analytics tags and site promotions.

Personalization

These modules provide resources for Personalization.

Location: *PortalServer_root*`/pzn/prereq.pzn/installedApps/Personalization_Workspace.ear/pznauthorportlet.war/WEB-INF/plugin.xml`

Table 407. List of modules that provide resources for Personalization.

Module	Description
wp_pzn_geolocation	Provides resources that are required for Geolocation Personalization.

Social lists

Description: The `wp_social_rendering` theme module provides the CSS styles that are used by social lists. It defines the capability with the name `social_rendering` and the version `8.5`.

`PortalServer_root``/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/modules/sr/css/sr.css`

No prerequisites are required to use this theme module.

Web Dock

These modules provide resources to the web dock application.

Location: *wp_profile_root*`installedApps/`*`cell`*`/PA_WebDockPortServlet.ear/WebDockPortlet.war/WEB-INF/plugin.xml`

Table 408. List of modules that provide resources to web Dock.

Module	Description
wp_webdock	This module provides resources that are required by web dock application.

IBM MobileFirst Integration

Description: Provides the modules for MobileFirst application integration.

Location for all wp_worklight_* modules: `dav:fs-type1/themes/Portal8.5/contributions/worklight61.json`

Location for all wl_* modules:*PortalServer_root*/theme/wp.theme.worklight.ext/installableApps/wp.theme.worklight.ext.ear/wp.theme.worklight.ext.war/WEB-INF

Table 409. List of MobileFirst Integration modules and prerequisites.

Module	Description
wp_worklight	Mapping module to MobileFirst integration extension
wp_worklight_android	Mapping module to MobileFirst integration extension for Android
wp_worklight_css	Mapping module to MobileFirst CSS
wp_worklight_css_android	Mapping module to MobileFirst CSS for Android
wp_worklight_css_ios	Mapping module to MobileFirst CSS for iOS
wp_worklight_ios	Mapping module to MobileFirst integration extension for iOS
wp_worklight_jsonstore	Mapping module to MobileFirst client JSONStore
wp_worklight_plugins	Mapping module to MobileFirst integration extension
wp_worklight_plugins_android	Mapping module to MobileFirst integration extension for Android
wp_worklight_plugins_ios	Mapping module to MobileFirst integration extension for iOS
wl_android_6	Provides MobileFirst client and Cordova JavaScript resources for Android devices
wl_android_61	Provides MobileFirst client and Cordova JavaScript resources for Android devices
wl_client_css_android_6	Provides MobileFirst client CSS for Android devices, specifically for the diagnostic window, and modal dialog
wl_client_css_android_61	Provides MobileFirst client CSS for Android devices, specifically for the diagnostic window, and modal dialog
wl_client_css_ios_6	Provides MobileFirst client CSS for iOS devices, specifically for the diagnostic window, and modal dialog
wl_client_css_ios_61	Provides MobileFirst client CSS for iOS devices, specifically for the diagnostic window, and modal dialog
wl_client_jsonstore_android_6	Provides MobileFirst client JSONStore for Android devices
wl_client_jsonstore_android_61	Provides MobileFirst client JSONStore for Android devices
wl_client_jsonstore_ios_6	Provides MobileFirst client JSONStore for iOS devices
wl_client_jsonstore_ios_61	Provides MobileFirst client JSONStore for iOS devices
wl_config	Data source that injects the initialization for client and Cordova API

Table 409. List of MobileFirst Integration modules and prerequisites. (continued)

Module	Description
wl_cordova_css_android_6	Provides Cordova client CSS, specifically for the tab bar component
wl_ios_6	Provides MobileFirst client and Cordova JavaScript resources for iOS devices
wl_ios_61	Provides MobileFirst client and Cordova JavaScript resources for iOS devices
wl_plugins_android_61	Provides MobileFirst plug-in JavaScript resources for Android devices
wl_plugins_ios_61	Provides MobileFirst plug-in JavaScript resources for iOS devices

Sametime

These modules provide the code for integrating with IBM Sametime `stlinks` support and new proxy support.

The `plugin.xml` file location is *PortalServer_root*`/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/sametime.jar`

Table 410. List of Sametime modules and prerequisites.

Module	Description
wp_sametime_links	Earlier STlinks support.
wp_sametime_proxy	New Sametime proxy support.

Web Application Integrator

These modules provide the Web Application Integrator.

Location: *PortalServer_root*`/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/ic4_wai.jar!/plugin.xml`

Table 411. These modules provide the Web Application Integrator.

Module	Description
wp_ic4_wai_resources	Provides resources to enable Connections integration with WAI.

Client Utils

These modules provide JavaScript utility code with no dependencies on the Dojo Toolkit in the i$ global namespace. These modules are useful for light-weight themes with no framework dependencies. The code includes type checks, configuration that merges, IO utilities, JSON parsing, DOM helpers, Promises, and eventing.

The `plugin.xml` file location is *PortalServer_root*`/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/ibmc.jar`

Table 412. List of Client Utils modules and prerequisites.

Module	Description
wp_client_main	Contains basic utilities.
wp_client_ext	Contains advanced utilities like Promises, IO, DOM helpers, events, and the deferred module loading code.
wp_client_dnd	Contains drag-and-drop capabilities. Location: *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/ibmc.jar
wp_client_logging	Contains logging capabilities for error, warning, and information messages. Location: *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/ibmc.jar
wp_client_selector	Contains a JavaScript CSS selector engine. Location: *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/ibmc.jar
wp_client_tracing	Contains tracing capabilities. Location: *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/ibmc.jar

Portal Client

These modules provide utilities and base code for other modules, including Tagging and Rating.

The plugin.xml file location is *PortalServer_root*/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/portalclient.jar

Table 413. List of Portal Client modules and prerequisites.

Module	Description
wp_portlet_client_model	Public client programming model. Includes REST service and state handling. Used as a base prerequisite to other functional modules.
wp_portal_client_utils	Common utilities for other modules to use (XML handling, authentication), used as base prerequisite to other functional modules.
wp_portal_client_rest_utils	Client-side data stores used by other Portal features, such as Tagging and Rating, to access the Portal REST modules on the server.

Table 413. List of Portal Client modules and prerequisites. (continued)

Module	Description
wp_portal_ui_utils	Common UI elements. Used as a base prerequisite for some theme dialogs, such as the Content Mapping Picker and Tagging and Rating
wp_tagging_rating	Tagging and Rating widgets
wp_tagging_rating_opensearch	Open Search plug-in for Tagging and Rating
wp_tagging_rating_light	Provides all lightweight inline tagging and rating widgets.
wp_tagging_rating_menu	Provides tagging and rating menu items in the actions menu.
wp_tagging_rating_tagcloud	Provides tag cloud in Tag center page.
wp_template_select_dialog	A dialog that is started from the New Page dialog that allows a user to pick on a page template, which to base their newly created page.

Dialog API

These modules provide the theme dialog function.

Location: *PortalServer_root*`/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/dialog.jar`

Table 414. These modules provide the theme dialog function

Module	Description
wp_dialog_css	Provides the CSS styling for dialogs that are displayed by the dialog API.
wp_dialog_draggable	Provides support to allow dialogs that are created with the dialog API to create dialogs to display page content.
wp_dialog_main	Provides an API to create dialogs to display page content
wp_dialog_util	Provides utilities to support the dialog API.

OneUI

These modules provide the CSS for OneUI.

Meta-Module

- wp_one_ui
- wp_one_ui_dijit

The `plugin.xml` file location is *PortalServer_root*`/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/oneui.jar`

Table 415. List of OneUI modules and prerequisites.

Module	Description
wp_one_ui_21	Provides OneUI v2.1 CSS.

Table 415. List of OneUI modules and prerequisites. (continued)

Module	Description
wp_one_ui_30	Provides OneUI v3.0 CSS.
wp_one_ui_dijit_30	Provides dijit support for OneUI.
wp_one_ui_303	Provides OneUI v3.03 CSS.
wp_one_ui_dijit_303	Provides dijit support for OneUI v3.03.

Dojo

These modules are used for separate layers that are built from a Dojo build profile. The `djconfig` object is provided by a Portal data source. The POC URL for Dojo 1.6 is `dojo:config@v1.6`. For Dojo 1.7, it is `dojo:config@v1.7`. For Dojo 1.9, it is `dojo:config@v1.9`. Look in the `/dojo/build.txt` to see which files are in each layer. Each module contributes to the head section.

Dojo Meta-Modules

These Dojo modules are not associated with a specific Dojo release. In WebSphere Portal Express, a meta-module paradigm was added for Dojo support. The user can define which version of Dojo to use, 1.9, 1.7 or 1.6. The meta-modules do not have the Dojo version specified.

The meta module definitions are stored in the following files:

`dojo19.json`
: In *PortalServer_root*`/theme/wp.theme.dojo/installedApps/dojo.ear/dojo.war/v1.9`

`dojo17.json`
: In *PortalServer_root*`/theme/wp.theme.dojo/installedApps/dojo.ear/dojo.war/v1.7`

`dojo16.json`
: In *PortalServer_root*`/theme/wp.theme.dojo/installedApps/dojo.ear/dojo.war/v1.6`

To activate a version, copy the corresponding file into the WebDAV `dav:fs-type1:/themes/portal8.5/contribution` folder. Delete the previous file and restart the server.

Note: You can have only one file in the folder at one time because the `dojo16.json`, `dojo17.json`, and `dojo19.json` files are not supported at the same time. Specific Dojo modules are listed in the Dojo 1.9, Dojo 1.7 and Dojo 1.6 sections. For a list of Dojo classes that are provided by each meta-module, see more information Dojo classes that are provided by the Dojo modules.

The following list shows the meta-modules.

- dojo
- dojo_app
- dojo_data
- dojo_data_ext
- dojo_dnd_basic
- dojo_dnd_ext

- dojo_dom
- dojo_fmt
- dojo_fx
- dojo_node_list
- dojo_promise
- dojo_request
- dojo_selector_lite
- dijit
- dijit_app
- dijit_editor
- dijit_editor_plugins
- dijit_form
- dijit_layout_basic
- dijit_layout_ext
- dijit_menu
- dijit_tree
- dijit_all
- dojox_all
- dojox_app
- dojox_aspect
- dojox_calendar
- dojox_charting
- dojox_charting_all
- dojox_collections
- dojox_data_all
- dojox_data_basic
- dojox_dgauges
- dojox_fx
- dojox_gfx
- dojox_gfx3d
- dojox_grid_all
- dojox_html_basic
- dojox_images
- dojox_io
- dojox_layout_ext
- dojox_layout_basic
- dojox_mobile
- dojox_mobile_app
- dojox_mobile_compat
- dojox_mobile_app_compat
- dojox_string
- dojox_uuid
- dojox_widget_standby
- dojox_xml

Dojo 1.9 modules

The `plugin.xml` file location is *PortalServer_root*`/theme/wp.theme.dojo/installedApps/dojo.ear/dojo.war/v1.9`

Table 416. List of Dojo 1.9 modules and prerequisites.

Module
dojo_19
dojo_app_19
dojo_data_19
dojo_data_ext_19
dojo_dnd_basic_19
dojo_dnd_ext_19
dojo_dom_19
dojo_fmt_19
dojo_fx_19
dojo_node_list_19
dojo_promise_19
dojo_request_19
dojo_selector_lite
dijit_19
dijit_app_19
dijit_editor_19
dijit_editor_plugins_19
dijit_form_19
dijit_layout_basic_19
dijit_layout_ext_19
dijit_menu_19
dijit_theme_basic_19
dijit_theme_claro_19
dijit_tree_19
dijit_all_19
dojox_app_19
dojox_aspect_19
dojox_calendar_19
dojox_collections_19
dojox_data_basic_19
dojox_dgauges_19
dojox_fx_19
dojox_gfx_19
dojox_gfx3d_19
dojox_html_basic_19
dojox_images_19

Table 416. List of Dojo 1.9 modules and prerequisites. (continued)

Module
dojox_io_19
dojox_layout_basic_19
dojox_layout_ext_19
dojox_string_19
dojox_widget_standby
dojox_uuid_19
dojox_xml_19
dojox_mobile_19
dojox_mobile_app_19
dojox_mobile_compat_19
dojox_mobile_app_compat_19
dojox_charting_19
dojox_charting_all_19
dojox_data_all_19
dojox_grid_all_19
dojox_all_19

Dojo 1.7 modules

The `plugin.xml` file location is *PortalServer_root*`/theme/wp.theme.dojo/installedApps/dojo.ear/dojo.war/v1.7`

Table 417. List of Dojo 1.7 modules and prerequisites.

Module
dojo_17
dojo_app_17
dojo_data_17
dojo_dnd_basic_17
dojo_dnd_ext_17
dojo_dom_17
dojo_fmt_17
dojo_fx_17
dojo_node_list_17
dijit_17
dijit_app_17
dijit_editor_17
dijit_editor_plugins_17
dijit_form_17
dijit_layout_basic_17
dijit_layout_ext_17
dijit_menu_17

Table 417. List of Dojo 1.7 modules and prerequisites. (continued)

Module
dijit_tree_17
dijit_all_17
dojox_aspect_17
dojox_collections_17
dojox_data_basic_17
dojox_fx_17
dojox_gfx_17
dojox_gfx3d_17
dojox_html_basic_17
dojox_io_17
dojox_layout_basic_17
dojox_string_17
dojox_uuid_17
dojox_xml_17
dojox_mobile_17
dojox_mobile_app_17
dojox_mobile_compat_17
dojox_mobile_app_compat_17
dojox_charting_17
dojox_charting_all_17
dojox_data_all_17
dojox_grid_all_17
dojox_all_17

Dojo 1.6 modules

The `plugin.xml` file location is *PortalServer_root*`/theme/wp.theme.dojo/installedApps/dojo.ear/dojo.war/v1.6`

Table 418. List of Dojo 1.6 modules and prerequisites.

Module
dojo_16
dojo_app_16
dojo_data_16
dojo_dnd_basic_16
dojo_dnd_ext_16
dojo_dom_16
dojo_fmt_16
dojo_fx_16
dojo_node_list_16
dijit_16

Table 418. List of Dojo 1.6 modules and prerequisites. (continued)

Module
dijit_app_16
dijit_editor_16
dijit_editor_plugins_16
dijit_form_16
dijit_layout_basic_16
dijit_layout_ext_16
dijit_menu_16
dijit_theme_tundra_16
dijit_tree_16
dojox_aspect_16
dojox_charting_16
dojox_collections_16
dojox_data_basic_16
dojox_fx_16
dojox_gfx_16
dojox_gfx3d_16
dojox_html_basic_16
dojox_io_16
dojox_layout_basic_16
dojox_string_16
dojox_uuid_16
dojox_xml_16
dojox_mobile_16
dojox_mobile_app_16
dojox_mobile_compat_16
dojox_mobile_app_compat_16

JQuery

These modules provide jQuery.

Location: *PortalServer_root*`/theme/wp.theme.jquery/installedApps/wp.theme.jquery.ear/wp.theme.jquery.war/WEB-INF/plugin.xml`

Table 419. These modules provide jQuery.

Module	Description
jquery_1_10_2	Provides jQuery v1.10.2 core resources.

Modularized Page Builder

These modules provide support for the modularized Page Builder themes from 7.0.

The plugin.xml file location is *PortalServer_root*/ui/wp.mashup.cc.deploy/installedApps/MashupCommonComponent.ear/mm.builder.v3001.war/WEB-INF

Table 420. List of Modularized Page Builder modules and prerequisites.

Module	Description
mm_template_layout	Widget that handles layout refreshing, this module is only used for Modularized Page Builder themes from 7.0
mm_customize_shelf	Base widget for the site toolbar, this module is only used for Modularized Page Builder themes from 7.0
mm_page_toolbar	Widget for the **Customize**, **Hidden Content**, **Save & Exit**, and **Cancel**. This module is only used for Modularized Page Builder themes from 7.0
mm_content_set_list	Menu framework list widget from 7.0. This module is only used for Modularized Page Builder themes from 7.0
mm_content_set_menu	Menu framework widget from 7.0. This module is only used for Modularized Page Builder themes from 7.0
mm_controlled_nav_widget	Navigation widget from 7.0. This module is only used for Modularized Page Builder themes from 7.0

Theme

These modules provide earlier Portal 7.0 theme skins.

The plugin.xml file location varies and is documented in the module description.

Table 421. List of Theme modules and prerequisites.

Module	Description
wp_pagebuilder_standard_skin_70	Earlier 7.0 standard skin Location: dav:fs-type1/skins/Standard/
wp_pagebuilder_noskin_skin_70	Earlier 7.0 no skin Location: dav:fs-type1/skins/NoSkin/

Mashups Enabler

Provides the modules for Enabler from the Mashups 3.0.0.1 release.

The plugin.xml file location is *PortalServer_root*/ui/wp.mashup.cc.deploy/installedApps/MashupCommonComponent.ear/mm.enabler.war.v3001.war/WEB-INF

Table 422. List of Mashups Enabler modules and prerequisites.

Module	Description
mm_open_ajax_hub	Open Ajax Hub
mm_enabler	Full enabler

Table 422. List of Mashups Enabler modules and prerequisites. (continued)

Module	Description
mm_enabler_core	Enabler core; provides only iWidget container support but no model APIs
mm_enabler_ext	Enabler support is not included in the mm_enabler

Mashups Builder

Provides the modules for Builder from the Mashups 3.0.0.1 release.

The `plugin.xml` file location is *PortalServer_root*`/ui/wp.mashup.cc.deploy/installedApps/MashupCommonComponent.ear/mm.builder.v3001.war/WEB-INF`

Table 423. List of Mashups Builder modules and prerequisites.

Module	Description
mm_builder	Builder base
mm_builder_ext	Builder support
mm_builder_dialogs	Builder dialog base widget
wp_theme_widget	Menu support for iWidgets

User Interface

These modules provide user interface code.

The `plugin.xml` file location is *PortalServer_root*`/ui/wp.mashup.cc.deploy/installedApps/MashupCommonComponent.ear/mm.builder.v3001.war/WEB-INF`

Page handling

Table 424. List of User interface modules and prerequisites for page handling.

Module	Description
mm_move_page	Move Page widget
mm_new_page_dialog	New Page dialog widget
mm_delete_page	Menu contribution for deleting a page

Wiring

Table 425. List of User interface modules and prerequisites for wiring.

Module	Description
mm_builder_wiring	Wiring widget

Portlet handling

Table 426. List of User interface modules and prerequisites for portlet handling.

Module	Description
mm_delete_control	Menu contribution for deleting a portlet

Page Builder

These modules contain code for running Page Builder functions.

The `plugin.xml` file location is *PortalServer_root*`/theme/wp.theme.modules/webapp/installedApps/ThemeModules.ear/ThemeModules.war/WEB-INF/lib/pagebuilder.jar`

Table 427. List of Page Builder modules and prerequisites.

Module	Description
wp_pagebuilder_base	Base code and utilities that are used by the other modules in this section.
wp_pagebuilder_ui	Initialization code and base widgets (such as dialogs) used by other Page Builder modules.
wp_pagebuilder_controls	Code to add the **Move portlet** menu actions (Move Up/Left/Right/Down).
wp_pagebuilder_debug	Client-side debugging and tracing code, this module is turned off by default.
wp_pagebuilder_data	Data stores used by Page Builder editing tools, and that follow the Dojo read/write API.
wp_pagebuilder_dnd	Support for portlet drag-and-drop in the page layout.
wp_pagebuilder_shelf	The site toolbar code, this module is used for modularized Page Builder themes from 7.0 only.
wp_pagebuilder_csa	Earlier CSA-only Page builder code, this module is used for modularized Page Builder themes from 7.0 only.
wp_pagebuilder_widget_css	Earlier CSA widget CSS, this module is used for modularized Page Builder themes from 7.0 only.
wp_pagebuilder_shelf_base	Contains the base code for the tabs in the theme toolbar, used a prerequisite for other modules that implement tabs in the toolbar.
wp_wcm_modal_dialog	Contains a framework that displays a page in an iframe inside a modal dialog Used as a prerequisite to other modules that use this dialog framework.
wp_managed_pages_support	Contains a JavaScript configuration object that is used by all of the managed pages theme modules.
wp_managed_pages_support_edit	Contains base JavaScript code that is shared by the project menu and preview managed pages modules.
wp_toolbar	A managed pages theme module for the toolbar.
wp_status_bar	The theme status bar that relays information, warning, and error messages to the user.
wp_project_menu	A dojo-less managed pages theme module for the view mode display of project menu.
wp_project_menu_edit	A managed pages theme module for the edit mode function of the project menu.

Table 427. List of Page Builder modules and prerequisites. (continued)

Module	Description
wp_preview	CSS for the managed pages Preview controls seen in page view mode.
wp_preview_menu	A managed pages theme module for the Preview function that is seen in the **More** menu
wp_template_select_dialog	A dialog that is started from the **New Page** dialog. A user can pick a page template and base a newly created page on that template.

"Module capabilities"
Module capabilities enable the use of modules in the theme.

"Dojo classes provided by the Dojo modules" on page 2640
A list of Dojo classes that are provided by each version 1.9 Dojo meta-module.

Module capabilities:

Module capabilities enable the use of modules in the theme.

CF03 This table lists explicitly defined capabilities. Implicit capabilities are not listed. Every module automatically surfaces a default capability, or implicit capability, which is the capability with the exact same name and version as the module.

Table 428. List of module capabilities

Capability Name	Capability Value	Modules in the Capability
dojo	1.9	dojo_19
portal.livetext.action	8.0	wp_liveobject_framework
portal.livetext.hcard	8.0	wp_liveobject_framework
portal.livetext.c2a	8.0	wp_liveobject_framework
portal.livetext.adr	8.0	wp_liveobject_framework
mashups.builder	3.0.0.1	mm_enabler
open_ajax_hub	2.0	mm_open_ajax_hub
mashups.enabler	3.0.0.1	mm_enabler mm_enabler_core
widget_container	2.1	mm_enabler
active_site_analytics	8.0	wp_analytics
content_mapping.picker	8.0	wp_content_mapping_picker
federated_documents.picker	8.0	wp_federated_documents_picker
oneUI	2.1	wp_one_ui_21
oneUI	3.0.1	wp_one_ui_30
oneUI	3.0.3	wp_one_ui_303
CF03 oneUIDijit	3.0.1	wp_one_ui_dijit_30
CF03 oneUIDijit	3.0.3	wp_one_ui_dijit_303

Table 428. List of module capabilities (continued)

Capability Name	Capability Value	Modules in the Capability
cp_tagging_rating	8.0	wp_tagging_rating wp_tagging_rating_opensearch
sametime.proxy	8.5.2	wp_sametime_proxy
sametime.links	8.5.2	wp_sametime_links
portal.search	8.0	wp_search

For extra modules and capabilities, see Content Template Catalog theme extensions and Customizing the CSS styles of social lists.

Dojo classes provided by the Dojo modules:

A list of Dojo classes that are provided by each version 1.9 Dojo meta-module.

dojo dojo.main

dojo_dom
- dojo.window
- dojo.uacss
- dojo.html
- dojo.parser
- dojo.touch

dojo_app
- dojo.cookie
- dojo.back
- dojo.hash
- dojo.i18n
- dojo.io.iframe
- dojo.io.script
- dojo.string
- dojo.cache
- dojo.Stateful
- dojo.AdapterRegistry
- dojo.DeferredList
- dojo.query

dojo_fx
- dojo.fx
- dojo.fx.Toggler
- dojo.fx.easing

dojo_fmt
- dojo.currency
- dojo.number

dojo.text

dojo.colors

dojo.date

dojo.date.locale

dojo.date.stamp

dojo_node_list

dojo.NodeList-fx

dojo.NodeList-html

dojo.NodeList-manipulate

dojo.NodeList-traverse

dojo_data

dojo.data.ItemFileReadStore

dojo.data.ItemFileWriteStore

dojo.data.util.simpleFetch

dojo.data.util.sorter

dojo.data.util.filter

dojo.data.ObjectStore

dojo_dnd_basic

dojo.dnd.common

dojo.dnd.Source

dojo.dnd.AutoSource

dojo.dnd.Target

dojo.dnd.Selector

dojo.dnd.Container

dojo.dnd.Manager

dojo.dnd.Avatar

dojo_dnd_ext

dojo.dnd.move

dojo.dnd.autoscroll

dojo.dnd.Mover

dojo.dnd.Moveable

dojo.dnd.TimedMoveable

dojo_promise

dojo.promise.all

dojo.promise.first

dojo.promise.instrumentation

dojo.promise.Promise

dojo.promise.tracer

dojo_request

dojo.request.default

dojo.request.handlers

dojo.request.iframe

dojo.request.node

dojo.request.notify

dojo.request.registry

dojo.request.script

dojo.request.util

dojo.request.watch

dojo_data_ext

dojo.store.Cache

dojo.store.DataStore

dojo.store.JsonRest

dojo.store.Memory

dojo.store.Observable

dojo.store.util.QueryResults

dojo.store.util.SimpleQueryEngine

dojo_selector_lite

dojo.selector.lite

dijit dijit.dijit

dijit._Templated

dijit.Fieldset

dijit._BidiMixin

dijit_app

dijit.ColorPalette

dijit.ProgressBar

dijit.Toolbar

dijit.ToolbarSeparator

dijit.Tooltip

dijit_menu

dijit.Menu

dijit.MenuBar

dijit.MenuBarItem

dijit.MenuItem

dijit.MenuSeparator

dijit.CheckedMenuItem

dijit.PopupMenuItem

dijit.PopupMenuBarItem

dijit_form

dijit.InlineEditBox

dijit.Calendar

dijit.form.Button

dijit.form.CheckBox

dijit.form.ComboBox

dijit.form.ComboButton

dijit.form.CurrencyTextBox

dijit.form.DateTextBox

dijit.form.DropDownButton

dijit.form.FilteringSelect

dijit.form.Form

dijit.form.HorizontalSlider

dijit.form.HorizontalRule

dijit.form.HorizontalRuleLabels

dijit.form.MappedTextBox

dijit.form.MultiSelect

dijit.form.NumberSpinner

dijit.form.NumberTextBox

dijit.form.RadioButton

dijit.form.RangeBoundTextBox

dijit.form.Select

dijit.form.SimpleTextarea

dijit.form.Slider

dijit.form.TextBox

dijit.form.Textarea

dijit.form.TimeTextBox

dijit.form.ToggleButton

dijit.form.ValidationTextBox

dijit.form.VerticalSlider

dijit.form.VerticalRule

dijit.form.VerticalRuleLabels

dijit_editor

dijit.Editor

dijit._editor._Plugin

dijit._editor.RichText

dijit._editor.html

dijit._editor.range

dijit._editor.selection

dijit_editor_plugins

dijit._editor.plugins.AlwaysShowToolbar

dijit._editor.plugins.EnterKeyHandling

dijit._editor.plugins.FontChoice

dijit._editor.plugins.FullScreen

dijit._editor.plugins.LinkDialog

dijit._editor.plugins.NewPage

dijit._editor.plugins.Print

dijit._editor.plugins.TabIndent

dijit._editor.plugins.TextColor

dijit._editor.plugins.ToggleDir

dijit._editor.plugins.ViewSource

dijit_layout_basic

dijit.layout.ContentPane

dijit.layout.LinkPane

dijit.TitlePane

dijit.Dialog

dijit.TooltipDialog

dijit_layout_ext

dijit.layout.BorderContainer

dijit.layout.AccordionContainer

dijit.layout.AccordionPane

dijit.layout.TabContainer

dijit.layout.LayoutContainer

dijit.layout.ScrollingTabController

dijit.layout.SplitContainer

dijit.layout.StackContainer

dijit.layout.StackController

dijit.layout.TabController

dijit_tree

dijit.Tree

dijit.tree.dndSource

dijit.tree.TreeStoreModel

dijit.tree.ForestStoreModel

dijit-all

dijit.dijit-all

dojox_gfx

dojox.gfx.svg

dojox.gfx.vml

dojox.gfx

dojox_charting

dojox.charting.Chart

dojox_charting_all

dojox.charting.BidiSupport

dojox.charting.Chart

dojox.charting.Chart2D

dojox.charting.Chart3D

dojox.charting.DataChart

dojox.charting.DataSeries

dojox.charting.Element

dojox.charting.Series

dojox.charting.StoreSeries

dojox.charting.Theme

dojox.charting.action2d.Base

dojox.charting.action2d.ChartAction

dojox.charting.action2d.Highlight

dojox.charting.action2d.Magnify

dojox.charting.action2d.MouseIndicator

dojox.charting.action2d.MouseZoomAndPan

dojox.charting.action2d.MoveSlice

dojox.charting.action2d.PlotAction

dojox.charting.action2d.Shake

dojox.charting.action2d.Tooltip

dojox.charting.action2d.TouchIndicator

dojox.charting.action2d.TouchZoomAndPan

dojox.charting.axis2d.Base

dojox.charting.axis2d.common

dojox.charting.axis2d.Default

dojox.charting.axis2d.Invisible

dojox.charting.plot2d.Areas

dojox.charting.plot2d.Bars

dojox.charting.plot2d.Base

dojox.charting.plot2d.Bubble

dojox.charting.plot2d.Candlesticks

dojox.charting.plot2d.ClusteredBars

dojox.charting.plot2d.ClusteredColumns

dojox.charting.plot2d.Columns
dojox.charting.plot2d.common
dojox.charting.plot2d.Default
dojox.charting.plot2d.Grid
dojox.charting.plot2d.Lines
dojox.charting.plot2d.Markers
dojox.charting.plot2d.MarkersOnly
dojox.charting.plot2d.OHLC
dojox.charting.plot2d.Pie
dojox.charting.plot2d.Scatter
dojox.charting.plot2d.Spider
dojox.charting.plot2d.Stacked
dojox.charting.plot2d.StackedAreas
dojox.charting.plot2d.StackedBars
dojox.charting.plot2d.StackedColumns
dojox.charting.plot2d.StackedLines
dojox.charting.plot3d.Bars
dojox.charting.plot3d.Base
dojox.charting.plot3d.Cylinders
dojox.charting.scaler.common
dojox.charting.scaler.linear
dojox.charting.scaler.primitive
dojox.charting.themes.PlotKit.base
dojox.charting.themes.PlotKit.blue
dojox.charting.themes.PlotKit.cyan
dojox.charting.themes.PlotKit.green
dojox.charting.themes.PlotKit.orange
dojox.charting.themes.PlotKit.purple
dojox.charting.themes.PlotKit.red
dojox.charting.themes.Adobebricks
dojox.charting.themes.Algae
dojox.charting.themes.Bahamation
dojox.charting.themes.BlueDusk
dojox.charting.themes.Charged
dojox.charting.themes.Chris
dojox.charting.themes.Claro
dojox.charting.themes.common

dojox.charting.themes.CubanShirts

dojox.charting.themes.Desert

dojox.charting.themes.Distinctive

dojox.charting.themes.Dollar

dojox.charting.themes.Electric

dojox.charting.themes.gradientGenerator

dojox.charting.themes.Grasshopper

dojox.charting.themes.Grasslands

dojox.charting.themes.GreySkies

dojox.charting.themes.Harmony

dojox.charting.themes.IndigoNation

dojox.charting.themes.Ireland

dojox.charting.themes.Julie

dojox.charting.themes.MiamiNice

dojox.charting.themes.Midwest

dojox.charting.themes.Minty

dojox.charting.themes.PrimaryColors

dojox.charting.themes.PurpleRain

dojox.charting.themes.Renkoo

dojox.charting.themes.RoyalPurples

dojox.charting.themes.SageToLime

dojox.charting.themes.Shrooms

dojox.charting.themes.ThreeD

dojox.charting.themes.Tom

dojox.charting.themes.Tufte

dojox.charting.themes.WatersEdge

dojox.charting.themes.Wetland

dojox.charting.widget.BidiSupport

dojox.charting.widget.Chart

dojox.charting.widget.Chart2D

dojox.charting.widget.Legend

dojox.charting.widget.SelectableLegend

dojox.charting.widget.Sparkline

dojox.charting.plot2d.CartesianBase

dojox.charting.plot2d.commonStacked

dojox_gfx3d

dojox.gfx3d

dojox.gfx3d.gradient

dojox.gfx3d.lighting

dojox.gfx3d.matrix

dojox.gfx3d.object

dojox.gfx3d.scheduler

dojox.gfx3d.vector

dojox_fx

dojox.fx

dojox.fx.Shadow

dojox.fx.easing

dojox_layout_basic

dojox.layout.ResizeHandle

dojox.layout.ContentPane

dojox_collections

dojox.collections

dojox.collections.ArrayList

dojox.collections.BinaryTree

dojox.collections.Dictionary

dojox.collections.Queue

dojox.collections.Set

dojox.collections.SortedList

dojox.collections.Stack

dojox_uuid

dojox.uuid

dojox.uuid.Uuid

dojox.uuid.generateRandomUuid

dojox.uuid.generateTimeBasedUuid

dojox_html_basic

dojox.html

dojox.html.metrics

dojox.html.entities

dojox_xml

dojox.data.dom

dojox.xml.parser

dojox_data_basic

dojox.data.XmlStore

dojox.data.CsvStore

dojox_data_all

dojox.data.AndOrReadStore

dojox.data.AndOrWriteStore

dojox.data.AppStore

dojox.data.AtomReadStore

dojox.data.CdfStore

dojox.data.ClientFilter

dojox.data.CouchDBRestStore

dojox.data.css

dojox.data.CssClassStore

dojox.data.CssRuleStore

dojox.data.CsvStore

dojox.data.dom

dojox.data.FileStore

dojox.data.FlickrRestStore

dojox.data.FlickrStore

dojox.data.GoogleFeedStore

dojox.data.GoogleSearchStore

dojox.data.HtmlStore

dojox.data.HtmlTableStore

dojox.data.KeyValueStore

dojox.data.OpenSearchStore

dojox.data.OpmlStore

dojox.data.PersevereStore

dojox.data.PicasaStore

dojox.data.QueryReadStore

dojox.data.RailsStore

dojox.data.S3Store

dojox.data.ServiceStore

dojox.data.SnapLogicStore

dojox.data.WikipediaStore

dojox.data.XmlItem

dojox.data.XmlStore

dojox_grid_all

dojox.grid.cells

dojox.grid.DataGrid

dojox.grid.DataSelection

dojox.grid.LazyTreeGrid

dojox.grid.LazyTreeGridStoreModel

dojox.grid.Selection

dojox.grid.TreeGrid

dojox.grid.TreeSelection

dojox.grid.util

dojox.grid.cells.dijit

dojox.grid.cells.tree

dojox.grid.EnhancedGrid

dojox.grid.enhanced.plugins.AutoScroll

dojox.grid.enhanced.plugins.CellMerge

dojox.grid.enhanced.plugins.Cookie

dojox.grid.enhanced.plugins.Dialog

dojox.grid.enhanced.plugins.DnD

dojox.grid.enhanced.plugins.Exporter

dojox.grid.enhanced.plugins.Filter

dojox.grid.enhanced.plugins.GridSource

dojox.grid.enhanced.plugins.IndirectSelection

dojox.grid.enhanced.plugins.Menu

dojox.grid.enhanced.plugins.NestedSorting

dojox.grid.enhanced.plugins.Pagination

dojox.grid.enhanced.plugins.Printer

dojox.grid.enhanced.plugins.Rearrange

dojox.grid.enhanced.plugins.Search

dojox.grid.enhanced.plugins.Selector

dojox_aspect
dojox.lang.aspect

dojox_string
dojox.string.Builder

dojox.string.tokenize

dojox.string.sprintf

dojox_io
dojox.atom.io.model

dojox.atom.io.Connection

dojox_mobile
dojox.mobile

dojox_mobile_app
dojox.mobile.app

dojox_mobile_compat
dojox.mobile.compat

dojox_mobile_app_compat
dojox.mobile.app.compat

dojox_app
dojox.app.Controller

dojox.app.main

dojox.app.View

dojox.app.ViewBase

dojox.app.controllers.BorderLayout

dojox.app.controllers.History

dojox.app.controllers.HistoryHash

dojox.app.controllers.Layout

dojox.app.controllers.LayoutBase

dojox.app.controllers.Load

dojox.app.controllers.Transition

dojox.app.module.env

dojox.app.module.lifecycle

dojox.app.utils.mvcModel

dojox.app.utils.config

dojox.app.utils.constraints

dojox.app.utils.hash

dojox.app.utils.layout

dojox.app.utils.model

dojox.app.utils.nls

dojox.app.utils.simpleModel

dojox.app.widgets.Container

dojox_dgauges

dojox.dgauges.CircularGauge

dojox.dgauges.CircularRangeIndicator

dojox.dgauges.CircularScale

dojox.dgauges.CircularValueIndicator

dojox.dgauges.GaugeBase

dojox.dgauges.IndicatorBase

dojox.dgauges.LinearScaler

dojox.dgauges.LogScaler

dojox.dgauges.MultiLinerarScaler

dojox.dgauges.RectangularGauge

dojox.dgauges.RectangularRangeIndicator

dojox.dgauges.RectangularScale

dojox.dgauges.RectangularSegmentedRangeIndicator

dojox.dgauges.ScaleBase

dojox.dgauges.ScaleIndicator

dojox.dgauges.TextIndicator

dojox-image

dojox.image.Badge

dojox.image.FlikrBadge

dojox.image.Gallery

dojox.image.Lightbox

dojox.image.LightboxNano

dojox.image.Magnifier

dojox.image.MagnifierLite

dojox.image.SlideShow

dojox.image.ThumbnailPicker

dojox-all

Contains all of the other dojox modules

dojox_layout_ext

dojox.layout.BorderContainer

dojox.layout.Dock

dojox.layout.DragPane

dojox.layout.ExpandoPane

dojox.layout.FloatingPane

dojox.layout.GridContainer

dojox.layout.RadioGroup

dojox.layout.RotatorContainer

dojox.layout.ScrollPane

dojox.layout.TableContainer

dojox.layout.ToggleSplitter

dojox_widget_standby

dojox.widget.standby

Modules that add features to a theme

Theme modules contribute resources, such as JavaScript, CSS and HTML, to a theme or page. The following modules come with the modularized theme and specifically contribute a feature rather than working in the background. The modules can be added to a theme by listing them in the theme's default profile, or to a single page or set of pages.

Portal 8.5 theme

Table 429. Portal 8.5 theme modules and features

Module name	Feature
wp_status_bar	Adds a status bar for displaying informational, warning, and error messages to the users.
wp_layout_windowstates	Allows portlets to be minimized, maximized, and restored from their menus in the skin.
wp_legacy_layouts	Adds responsive layout support for earlier standard portal pages.

Table 429. Portal 8.5 theme modules and features (continued)

Module name	Feature
wp_liveobject_framework	The Live Text Framework allows portlet applications to display text live names and click to action links.

Site toolbar

Table 430. Site toolbar modules and features

Module name	Feature
wp_toolbar_host	Integrates a theme with the toolbar in both view and edit modes, use either wp_toolbar_host or both wp_toolbar_host_view and wp_toolbar_host_edit to add the toolbar to a theme.
wp_toolbar_host_view	Integrates the toolbar into a theme for viewing purposes only.
wp_toolbar_host_edit	Integrates the toolbar into a theme for editing purposes.
wp_toolbar_ghost	Add this module to pages that do not support edit mode to allow interoperability with the toolbar, so if a user navigates from an editable page to one with this module, the toolbar remains open, and the user is able to continue editing.
wp_draft_page_ribbon	Adds `Draft Page` in text that appears along the sides of a page that has a draft in the current project.

Menus

Table 431. Menu modules and features

Module name	Feature
wp_simple_contextmenu_main	Adds support for simple menus, and added to the default profile to enable **Portlet**, **Actions**, **More**, and **Toolbar** navigation menus.
wp_contextmenu_main	Adds support for the **Component Action** menu, which is used for inline editing of content items.
wp_skin_cam	Allows the Component Action Menu to be opened by clicking an icon in the portlet skin.

Tagging and rating

Table 432. Tagging and rating modules and features

Module name	Feature
wp_tagging_rating_light	Provides all lightweight inline tagging and rating widgets.

Table 432. Tagging and rating modules and features (continued)

Module name	Feature
wp_tagging_rating_menu	Adds the tagging and rating menu items in the Actions and portlet menus.
wp_tagging_rating_tagcloud	Provides the widgets in Tag Cloud portlet.

Analytics

Table 433. Analytics modules and features

Module name	Feature
wp_analytics_aggregator	Adds the analytics aggregator
wp_analytics_tags	Adds the **Sites Promotions...** and **Analytics Tags...** options to the Actions and portlet menus, and inserts the microformats for analytics tags and site promotions into the page.
wp_analytics_overlay_reports	Provides analytics overlay reports for portlets and pages.
wp_analytics	Provides all analytics features: aggregator, the analytics tags and site promotion function, and the overlay reports.

Search

Table 434. Search modules and features

Module name	Feature
wp_searchbar	Adds the inline search bar that redirects to the search page to show results.

Web content management

Table 435. Web content management modules and features

Module name	Feature
wcm_inplaceEdit	Turns on inline editing for content items that are displayed with the Web Content Viewer portlet.
wp_oob_sample_styles	Styles for default web content samples.
wp_federated_documents_picker	Allows the insertion of remote content into Web Content Manager elements that contain a rich text field, by using the Insert Link to **Remote Document** button in the authoring portlet.

Social Rendering

Table 436. Social Rendering modules and features

Module name	Feature
wp_social_rendering_85	Provides the social rendering feature.

Connections

Table 437. Connections modules and features

Module name	Feature
wp_ic4_wai_resources	Resources to enable Connections 4 integration with Web Application Integrator.

IBM Sametime

Table 438. Sametime modules and features

Module name	Feature
wp_sametime_proxy	Enables integration with Sametime using Sametime Proxy Server.

iWidget support

Table 439. iWidget support modules and features

Module name	Feature
wp_theme_widget	Allows for the rendering of iWidgets on pages.

Web Dock Application

Table 440. Web Dock Application modules and features

Module name	Feature
wp_webdock	Provides the web dock application.

Accessibility

Table 441. Accessibility modules and features

Module name	Feature
wp_theme_skin_region	Provides accessibility support to the portlet skins.
wp_theme_high_contrast	Adds support for high contrast mode.

Programming

Table 442. Programming modules and features

Module name	Feature
wp_portal	Provides JavaScript configuration objects that contain information about the Portal state to be used by the themes and portlets.
wp_client_main	Client-side JavaScript APIs for browser detection, event attachment, object manipulation, and type detection.
wp_client_ext	Client-side JavaScript APIs for modules and promises.
wp_client_selector	Client-side JavaScript CSS selection engine API.

Table 442. Programming modules and features (continued)

Module name	Feature
wp_worklight	Adds MobileFirst function.
wp_one_ui	Provides CSS for common elements in Portal. For more information, see the documentation in *PortalServer_root*\theme\wp.theme.modules\webapp\installedApps\ThemeModules.ear\ThemeModules.war\modules\oneui\v2.1\lotusOneUIv2.1_Documentation.zip.
wp_one_ui_dijit	An extension of One UI that provides styling for dijits, it acts as a Dojo theme. For more information, see *Themes and theming*.
wp_portlet_css	Styles for IBM Administrative portlets.
getting_started_module	Adds the default simple module.

Related information:

Themes and theming

Adding or removing a module from a profile

To add or remove a module, update the profile that is used to render a page for the theme.

Before you begin

Read the topics on profiles and modules *Working with profiles* and *Writing modules*.

Procedure

1. Open the profile file in the /profiles directory.
2. Make a copy of the profile file and give your copy a unique name.
3. Edit the .json file by adding or removing the specific module ID.
4. Copy the profile that you created to the /profiles directory.
5. Invalidate the resource aggregator cache to integrate your changes. Click the **Administration menu** icon. Then, click **Portal Analysis** > **Theme Analyzer**. Then, click **Utilities** > **Control Center** > **Invalidate cache**. CF06 Auto invalidation recognizes your changes automatically for WebDAV based themes. No further action is required. For more information, see Utilities.

   CF03 "Adding or removing a capability from a portlet"
   To add or remove a capability from a portlet, update the portlet.xml for the portlet, or update the portlet preferences sections for the portlet definition or portlet entity with XML access.

Adding or removing a capability from a portlet:

To add or remove a capability from a portlet, update the portlet.xml for the portlet, or update the portlet preferences sections for the portlet definition or portlet entity with XML access.

Before you begin

Read the topics on Module dependencies in portlets and writing modules.

Procedure

1. Open the `portlet.xml` in your portlet archive.
2. Create a portlet preference section. Or add a preference to the existing section. This example shows a preference that you can add to an existing section.

```
<portlet-preferences>
    <preference>
        <name>capability.1.id</name>
        <value>dojo</value>
        <read-only>true</read-only>
    </preference>
    <preference>
        <name>capability.1.minValue</name>
        <value>1.6</value>
        <read-only>true</read-only>
    </preference>
</portlet-preferences>
```

   For more information, see *Module dependencies in portlets*.
3. Package your `.WAR` file.
4. Redeploy the portlet with the Portal Administrator portlet. Click the **Administration menu** icon. Then, click **Portlet Management** > **Web Modules**.Install or update your portlet there.

ConfigEngine tasks for creating a new profile based on a template and an existing profile in the system

You can extend the theme module framework with a new ConfigEngine task.

You can create a profile from a profile template and an existing profile. The config task reads the existing profile, takes the deferred and non-deferred theme modules from it, and creates a profile in WebDAV based on the template and the theme modules in the existing profile.

You can use these configuration tasks only for theme profile files that are stored in the WebDAV file store.

For more information about the theme module framework and how you can administer it, see the section about *Managing theme capabilities*.

Use the following parameters with the **`create-profile-based-on-template`** configuration tasks.

WasUserid
: The WebSphere Application Server user ID.

WasPassword
: The WebSphere Application Server user ID.

PortalAdminId
: The WebSphere Portal Express administrator user ID.

PortalAdminPwd
: The WebSphere Portal Express administrator password.

ThemeUniqueName
: The unique name of the theme that contains the profile that you want to update.

TemplateProfileFilePath
: The file path to the local template profile that you want the new profile be based on.

TargetProfileFileName
The name of the target profile file that you want to create.

TemplateModuleListProfileFileName
The name of the existing profile to fetch the list of theme modules from.

OverwriteProfile
Can be true or false. Defines whether the profile in WebDAV should be overwritten if it already exists.

Use the following command to create profile with the name `profile_new_deferred.json` in the theme with the unique name `ibm.portal.85Theme`, which is based on the template that exists on your local hard drive in `/opt/WebSphere/profile_new_template.json`. Once it is created in WebDAV, it has the list of modules from the `profile_deferred.json` profile, which already exists. The example assumes that the WebSphere Application Server and WebSphere Portal Express credentials are defined in the `wkplc.properties` file.

```
./ConfigEngine.sh create-profile-based-on-template -DThemeUniqueName=ibm.portal.85Theme -DTemplatePr
```

Customizing the theme

The module framework allows themes to be customized in order to provide flexibility, enhance the user experience, and maximize performance.

"Create a copy of your theme" on page 2659
You need to create your own custom theme by copying the Portal Version 8.5 theme before you start customization. Creating a copy of the theme ensures that your theme has all the required elements for the theme to function and that changes are not overwritten by a fix pack.

"Dynamic content spots" on page 2666
The static template files use dynamic content spots to reference JSP files or other dynamic resources. The dynamic resources are stored in a WAR file.

"Layouts" on page 2677
You can apply ready-use layouts to your portal pages, modify the existing skins, or add your own custom layout to change how your pages display.

"Skins" on page 2682
You can apply ready-use skins to your portal pages, modify the existing skins, or add your own custom skin to change how your pages display.

"Simple menu framework" on page 2688
You can provide menu feeds in JSON format instead of XML format. The operations can then parse the JSON feed without requiring the XML parsing support in the Dojo toolkit.

"Adding a link to the resource permissions administration portlet in a menu" on page 2705
You can add a link to the resource permissions portlet to see or modify role types or inherited access.

"Theme templates (`theme.html`)" on page 2706
You can use static HTML to write portal themes. The static theme template is named `theme.html`.

"Changing the theme logo" on page 2710
You can change the theme logo to customize your portal site and rebrand it to reflect your business.

"Customizing navigation" on page 2712
Use dynamic content spots to determine what is displayed by Top, Primary, and Secondary navigation. Use the `navigation.jsp` file to map properties to the

dynamic content spot IDs in the `theme.html` files. Rendering of the navigation is done with a single JSP file with `<ul>` and `<li>` tags.

"Styles" on page 2718
The WebSphere Portal Express Version 8.5 provides a selection of ready-use styles that you can be applied to your portal pages. You can also modify the existing theme styles or add your own custom style to achieve the look that you want.

"Configuring the portal theme and modules" on page 2722
Themes and modules are configured through theme metadata properties and resource environment provider custom properties.

"Expression language beans for accessing programming models" on page 2728
Expression language (EL) beans are available for accessing WebSphere Programming models. These beans are accessed through the PortalBean represented in the global namespace by `wp`. The beans provide access to IBM WebSphere Portal Express models and associated classes.

"Drag-and-drop" on page 2779
Drag-and-drop uses HTML5 code to facilitate the layout of content.

"Displaying messages in the status bar module" on page 2782
Displaying messages in the status bar module.

"Updating the content menu to open on click instead of on hover" on page 2783
The content menu opens when a user hovers over a portlet containing IBM Web Content Manager items. In combined Cumulative Fix 05, you can set the content menu to open when a user clicks an icon in the portlet skin instead.

"Displaying the draft page ribbon" on page 2784
The draft page ribbon is a strip of text that appears along both sides of a page that has a draft in the current project.

"Adding jQuery to a theme" on page 2786
The jQuery library is a JavaScript library. IBM WebSphere Portal Express includes a jQuery module for the jQuery core so you can use jQuery in a theme.

"Using Sametime with the WebSphere Portal Version 8.5 theme" on page 2789
You can add modules to your profile to use IBM Sametime with the IBM WebSphere Portal Express Version 8.5 theme.

"Tags used by the portal JSPs" on page 2790
Learn about the most commonly used tags in the portal JSPs. Use these tags to modify the appearance and layout of the portal page.

Create a copy of your theme

You need to create your own custom theme by copying the Portal Version 8.5 theme before you start customization. Creating a copy of the theme ensures that your theme has all the required elements for the theme to function and that changes are not overwritten by a fix pack.

Note: Do not modify the Portal Version 8.5 theme directly, because this theme can be updated by service fix packs and override your changes.

"Copy the static resources for your theme" on page 2660
You need to make a unique copy of the Portal 8.5 theme static resources before you start customizing your custom theme.

"Copy the static resources for your skin" on page 2661
You need to make a unique copy of the static resources for your skin.

"Copying the dynamic resources for your theme" on page 2661
You need to make a unique copy of the dynamic resources for your theme.

Make sure that Eclipse,IBM Rational Application Developer or Rational Team Concert with the Java EE developer tools add-on is installed.

"Link the static resources to the dynamic resources for your theme" on page 2662
The dynamic resources for your theme must be linked to the static resources. You must modify the default IBM WebSphere Portal Express theme references to point to the dynamic resources in your theme. First, you must bind your theme to the web application.

"Make your custom skin the default skin" on page 2665
Add your custom skin to the skins list to set it as the default skin for your theme. If it is the only custom skin, it is automatically set as the default skin for your theme.

Copy the static resources for your theme:

You need to make a unique copy of the Portal 8.5 theme static resources before you start customizing your custom theme.

About this task

Important: Enable development mode before you copy the static resources. Set the `resourceaggregation.development.mode` property to true within the WP ConfigService resource environment provider. If you do not enable development mode, be sure to restart your portal server at the end before you use your copied theme on a page.

Procedure

1. Connect your WebDAV client, such as WebDrive or AnyClient, to `http://host:port/wps/mycontenthandler/dav/themelist/`.
2. Copy the `ibm.portal.85Theme` folder to a local disk.
3. Rename the folder to the name of your theme, such as `customTheme`.
4. In the `metadata` folder, edit the `localized_en.properties` file, or whichever file is your default locale, and change the value of the title key to the display name of your theme, such as `Custom Theme`. Save the file. Repeat this step for any of the other locale files for the languages that you plan to support.
5. Edit the `metadata.properties` file and change the `Portal8.5` part of the **`com.ibm.portal.layout.template.href`** value to `customTheme`. Make sure that you have two properties that look like the following example:

   ```
   com.ibm.portal.layout.template.href=dav\:fs-type1/themes/customTheme/layout-templates/2ColumnEqual/
   resourceaggregation.profile=profiles/profile_deferred.json
   ```

6. Save the file.
7. Delete the skins folder from your `customTheme`, which removes the extra copies of the skins that are shipped with IBM WebSphere Portal Express. Your custom skin is created in a later step.

   ```
   customTheme/skins
   ```

8. Copy the entire `customTheme` folder into the `themelist` folder.
9. Double-check the contents of the `customTheme` folder in the `themelist` to ensure that everything is copied correctly. Compare each subfolder in `customTheme` to the corresponding subfolder in `ibm.portal.85Theme` to ensure that it looks like the correct number of files were copied. Recopy any files or subfolders that are missing. In particular, be sure to:
   a. Double-check the contents of the `metadata.properties` file in the `themelist`.

Note: Ensure that the URI is encoded. If there are spaces or other characters in the URI that are not allowed, the theme does not work.

b. Double-check the contents of the `profiles` folder and ensure all the profile files are there.

Copy the static resources for your skin:

You need to make a unique copy of the static resources for your skin.

Procedure

1. Connect your WebDAV client to http://*host*:*port*/wps/mycontenthandler/dav/skinlist/.
2. Copy the `ibm.portal.85Hidden` folder to a local disk.
3. Rename the folder to the name of your skin, such as `customSkin`.
4. In the metadata folder, edit the `localized_en.properties` file, or whichever file is your default locale, and change the value of the title key to the display name of your skin, such as `Custom Skin`. Save the file. Repeat this step for any of the other locale files for the languages that you support.
5. Copy the entire `customSkin` folder into the `skinlist`.
6. Double-check the contents of the `customSkin` folder in the `skinlist` to ensure that everything copied correctly. Compare each subfolder in `customSkin` to the corresponding folder in `ibm.portal.85Hidden` to ensure that it looks like the correct number of files were copied. Recopy any files or subfolders that are missing.

Copying the dynamic resources for your theme:

You need to make a unique copy of the dynamic resources for your theme. Make sure that Eclipse,IBM Rational Application Developer or Rational Team Concert with the Java EE developer tools add-on is installed.

Procedure

1. Switch to the Java EE perspective, and select **File** > **New** > **Dynamic Web Project**.
2. In the Project name field, enter the name of your theme, such as `CustomTheme`.
3. If it is not already selected, select **2.4** for the Dynamic Web Module version.
4. Select **Add project to an EAR** and click **Next to the Web Module page**.
5. On the Web Module page, change **Context Root** to `customTheme`, or whatever you want your context root to be, and click **Finish**.
6. Expand your new CustomTheme project and find and expand the `WebContent` folder.
7. Find the *PortalServer_root*`\theme\wp.theme.themes\default85\installedApps\DefaultTheme85.ear\DefaultTheme85.war\skins` folder on file system and drag it onto the `WebContent` folder. This step copies and imports the skins folder into your dynamic web project.
8. Find the *PortalServer_root*`\theme\wp.theme.themes\default85\installedApps\DefaultTheme85.ear\DefaultTheme85.war\themes` folder on file system and drag it onto the `WebContent` folder. This step copies and imports the themes folder into your dynamic web project.
9. In your CustomTheme project, find the `WEB-INF` folder inside the `WebContent` folder.

10. Find the *PortalServer_root*\theme\wp.theme.themes\default85\installedApps\DefaultTheme85.ear\DefaultTheme85.war\WEB-INF\decorations.xml file on file system and drag it onto the WEB-INF folder. This step copies and imports the file into your dynamic web project.
11. Find the *PortalServer_root*\theme\wp.theme.themes\default85\installedApps\DefaultTheme85.ear\DefaultTheme85.war\WEB-INF\tld folder on file system and drag it onto the WEB-INF folder. This step copies and imports the tld folder into your dynamic web project.
12. If you are developing a theme for a production portal, copy the following plugin.xml file into your dynamic resources. Find the *PortalServer_root*\theme\wp.theme.themes\default85\installedApps\DefaultTheme85.ear\DefaultTheme85.war\WEB-INF\plugin.xml file on file system and drag it onto the WEB-INF folder.
    a. Follow the steps in the *Modify the dynamic resource references for your theme* before you export the EAR and installing on your portal.

    This step copies and imports the file into your dynamic web project.
13. The previous files and folders are the only ones that you need. Do not copy any others from the DefaultTheme85.war file. Find your CustomThemeEAR project, right-click it and select **Export** > **EAR file** from the menu.
14. Click **Browse**, select a folder to export your EAR file to.
15. Click **Save** and click **Finish**.
16. Log on to the WebSphere Integrated Solutions Console and go to **Applications** > **Application Types** > **WebSphere enterprise applications**.
17. Click the **Install** toolbar button.
18. Click **Browse...**, find, and select the EAR file that you exported and click **Next**.
19. Select **Fast Path**, expand **Choose to generate default bindings and mappings**, select **Generate Default Bindings**, and click **Next**.
20. Change any installation option values that you want, or use the default values, and click **Next**.
21. For **Map Modules to Servers**, select the **Custom Theme module** in the table. Then, select the **...,server=WebSphere_Portal** entry in the list, click **Apply**, and click **Next**.
22. Click **Finish**.
23. When the EAR file is done installing, click **Save** directly to the master configuration.
24. Check your CustomTheme EAR in the table of enterprise applications and click the **Start** toolbar button.
25. On the file system, find and expand the *wp_profile_root*\installedApps\<cell>\CustomThemeEAR.ear\CustomTheme.war folder. The unique copies of the dynamic resources for your theme are in the themes and skins folders. When customizing your theme, always change the files at that location and do not modify the ThemeModules.war file.

Link the static resources to the dynamic resources for your theme:

The dynamic resources for your theme must be linked to the static resources. You must modify the default IBM WebSphere Portal Express theme references to point to the dynamic resources in your theme. First, you must bind your theme to the web application.

"Binding your theme to the context root of the web application"
You must bind your theme to the context root of the web application for your theme.

"Modify the dynamic resource references for your theme" on page 2664
You must modify the dynamic resource references to link to the static resources for your theme.

Binding your theme to the context root of the web application:

You must bind your theme to the context root of the web application for your theme.

Procedure

1. In *PortalServer_root*\bin, create the file input.xml with the following contents:

```
<?xml version="1.0" encoding="UTF-8"?>
<request xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
   type="export"
>
<portal action="locate">
<skin action="export" objectid="*" />
<theme action="export" objectid="*" />
</portal>
</request>
```

2. From a command line, change to the *PortalServer_root*\bin directory and run the following **xmlaccess** command to export all skin and theme definitions to a file called output.xml:

```
xmlaccess -user <admin userid> -password <admin password> -url <hostname>:10039/wps/config -in input.xml -out output.xml
```

3. Edit the output.xml file. Search for the title of your theme, such as Custom Theme. Scroll up to the line with the surrounding <theme> tag. It is likely the last <theme> tag in the file. Modify it from:

```
<theme action="update" active="true" context-root="/wps/themeModules" default="false" domain="rel"
  objectid="ZJ_MLSU3F54089F00IP6G7P3F10S5" resourceroot="dynamicSpots">
```

to:

```
<theme action="update" active="true" context-root="/customTheme" default="false" domain="rel"
  objectid="ZJ_MLSU3F54089F00IP6G7P3F10S5" resourceroot="dynamicSpots" uniquename="customTheme">
```

Set the correct context-root and uniquename for your theme.

4. Search for the title of your skin. Scroll up to the line with the surrounding <skin> tag; it is likely the last <skin> tag in the file. Modify it from:

```
<skin action="update" active="true" context-root="/wps/themeModules" default="false" domain="rel"
  objectid="ZK_73OKBB1A088IE0I5O7IP2J0G77" resourceroot="Hidden" type="default">
```

to:

```
<skin action="update" active="true" context-root="/customTheme" default="false" domain="rel"
  objectid="ZK_73OKBB1A088IE0I5O7IP2J0G77" resourceroot="customSkin" type="default" uniquename="customSkin">
```

Set the correct context-root, resourceroot, and uniquename for your skin.

5. Find the <theme> tag for the Portal 8.5 theme. It is likely the first <theme> tag in the file. Find and copy one of the <allowed-skin> tag lines, such as:

```
<allowed-skin skin="ZK_CGAH47L00GJJ40IDC03MS13OS2" update="set"/>
```

Find the <theme> tag for your customTheme theme, it is likely the last <theme> tag in the file. Paste the <allowed-skin> tag line in just before the <parameter> tags. Modify the skin parameter value identifier to be the identifier of your customSkin skin, which can be found in the **objectid** parameter of the <skin> tag of your customSkin skin. It is likely the last <skin> tag in the file, such as:

```
<allowed-skin skin="ZK_N0IEGGC0I08PF0IDQ60O6310N4" update="set"/>
```

6. Find the **com.ibm.portal.layout.template.href** parameter for your customTheme theme. Modify it from:

```
<parameter name="com.ibm.portal.layout.template.href" type="string" update="set"><![CDATA[dav:fs-type1/themes/Portal8.5/layout-templates/2ColumnEqual/]]></parameter>
```

   to:

```
<parameter name="com.ibm.portal.layout.template.href" type="string" update="set"><![CDATA[dav:fs-type1/themes/customTheme/layout-templates/2ColumnEqual/]]></paramete
```

   Set the correct `context-root`, `resourceroot`, and `uniquename` for your layout template.

   Note: Ensure that the URI is encoded. If there are spaces or other characters in the URI that are not allowed, the theme does not work.

7. From the command line, run the following **xmlaccess** command to update the skin and theme definitions according to your change:

```
xmlaccess -user admin userid -password admin password -url hostname:10039/wps/config -in output.xml -out output2.xml
```

8. From the command line, run the following **xmlaccess** command to export all skin and theme definitions again to a file called `output3.xml`.

```
xmlaccess -user admin userid -password admin password -url hostname:10039/wps/config -in input.xml -out output3.xml
```

9. Edit the `output3.xml` file to verify that the result works, and then delete the `input.xml`, `output.xml`, `output2.xml`, and `output3.xml` files.

Modify the dynamic resource references for your theme:

You must modify the dynamic resource references to link to the static resources for your theme.

About this task

Dynamic content spots are defined through a module called `wp_dynamicContentSpots_85`. The module is defined in the `plugin.xml` file, which was copied when you copied your theme.

Procedure

1. Open the `plugin.xml` in the *wp_profile_root*`installedApps\`*cell*`\CustomThemeEAR.ear\CustomTheme.war\WEB-INF\` directory.
2. Edit the IDs and names to have a unique custom name. Use custom plug-in IDs with the prefix of your company name to ensure that your `.WAR` files are invalidated. See *Configuration for resource aggregation* for more information about creating custom plug-in IDs. For example,

```
<?xml version="1.0" encoding="UTF-8"?>
<plugin id="com.yourcompany.customtheme" name="Custom Theme Modules" provider-name="Your Company"

<extension id="wp_dynamicContentSpots_custom" point="com.yourcompany.portal.resourceaggregator.mo
        <module id="wp_dynamicContentSpots_custom">
            <contribution type="dyn-cs">
                <sub-contribution type="markup" ref-id="customtheme_footer">
                    <uri value="res:{war:context-root}/themes/html/dynamicSpots/footer.jsp"/>
                </sub-contribution>
                <sub-contribution type="markup" ref-id="customtheme_crumbTrail">
                    <uri value="mvc:res:{war:context-root}/themes/html/dynamicSpots/crumbTrail.js
                </sub-contribution>
                <sub-contribution type="markup" ref-id="customtheme_topNav">
                    <uri value="mvc:smartphone/tablet@res:{war:context-root}/themes/html/dynamicS
                </sub-contribution>
                <sub-contribution type="markup" ref-id="customtheme_primaryNav">
                    <uri value="mvc:res:{war:context-root}/themes/html/dynamicSpots/navigation.js
                </sub-contribution>
                <sub-contribution type="markup" ref-id="customtheme_secondaryNav">
                    <uri value="mvc:res:{war:context-root}/themes/html/dynamicSpots/navigation.js
                </sub-contribution>
```

```
                    <sub-contribution type="markup" ref-id="customtheme_sideNav">
                        <uri value="mvc:res:{war:context-root}/themes/html/dynamicSpots/sideNaviga
                    </sub-contribution>
                    <sub-contribution type="markup" ref-id="customtheme_mobileNav">
                        <uri value="mvc:smartphone/tablet@res:{war:context-root}/themes/html/dynam
                    </sub-contribution>
                    <sub-contribution type="markup" ref-id="customtheme_commonActions">
                        <uri value="res:{war:context-root}/themes/html/dynamicSpots/commonActions.
                    </sub-contribution>
                    <sub-contribution type="markup" ref-id="customtheme_layout">
                        <uri value="lm:template"/>
                    </sub-contribution>
                    <sub-contribution type="markup" ref-id="85theme_toolbar">
                        <uri value="mc:wp_toolbar85@mvc:dyn-cs%3Aid%3A85toolbar%2Csmartphone%40%2C
                    </sub-contribution>
                    <sub-contribution type="markup" ref-id="customtheme_head">
                        <uri value="res:{war:context-root}/themes/html/dynamicSpots/head.jsp"/>
                    </sub-contribution>
                    <sub-contribution type="markup" ref-id="customtheme_status">
                        <uri value="mc:wp_status_bar@res:{war:context-root}/themes/html/dynamicSpo
                    </sub-contribution>
                </contribution>
            </module>
        </extension>
```

3. Connect your WebDAV client to `http://host:port/wps/mycontenthandler/dav/fs-type1/`.
4. Modify the profiles for your custom theme to include the module in your `plugin.xml`.
   a. From the `themes\customTheme\profiles` folder, copy the profile files.
   b. Remove the `wp_dynamicContentSpots_85` module from the profile.
   c. Add the `wp_dynamicContentSpots_custom` module that you created by modifying the `plugin.xml`.
   d. Copy the profiles back to the profiles folder.
5. From the `themes\customTheme\nls` folder, copy the `theme_en.html` file, or whichever file is your default locale, to the local drive. Repeat for any of the other locale files for the languages that you support.
6. Edit the `theme*.html` files, and find and replace all occurrences of `85theme` with `customTheme`. For example, `dyn-cs:id:85theme_head` becomes `dyn-cs:id:customTheme_head`.
7. Save the files.
8. Copy the files into the `fs-type1` folder.
9. Restart IBM WebSphere Portal Express.

What to do next

Note: You do not need to modify the `theme.html` file in the `themes\customTheme` folder. That file is not used other than to redirect to the appropriate locale file in the `nls` folder. You need to modify it only if you add or remove locales.

Make your custom skin the default skin:

Add your custom skin to the skins list to set it as the default skin for your theme. If it is the only custom skin, it is automatically set as the default skin for your theme.

About this task

Important: If you did not enable development mode before you copied your theme, restart your portal server before you complete the following steps.

Procedure

1. Log on to IBM WebSphere Portal Express.
2. Click the **Administration menu** icon. Then, click **Portal User Interface** > **Themes and Skins**.
3. Select **Custom Theme** in the themes list and click **Edit theme**.
4. Select **Custom Skin** in **Skins for this theme** list and click **Set as Default**.
5. Click **OK**.

What to do next

Your theme is now available for use on your portal. To verify, create a page, edit page properties, and assign your theme to the page.

Note: Your custom theme uses ready-use modules, which are shared across themes and intentionally remain uncopied in the Theme Modules web application. For your theme to operate the Theme Modules web application must remain started in addition to the web application for your theme.

Dynamic content spots

The static template files use dynamic content spots to reference JSP files or other dynamic resources. The dynamic resources are stored in a WAR file.

"Working with dynamic content spots"
You can add dynamic content to your custom theme by using either client-side or server-side logic.

"Creating dynamic content spots" on page 2672
There are two ways you can create dynamic content spots. You can use a module, or a resource environment provider.

"Conditionally disable dynamic content spots" on page 2675
You can control whether a dynamic content spot is rendered into the page or not based on the fact whether a certain module exists on the page or not.

"Modules and dynamic content spots" on page 2675
You can use the modularized framework for dynamic content spots to override spots that were defined through resource environment providers.

"Dynamic content spot debugging" on page 2677
Dynamic Content Spots add dynamic markup into static themes or skins. Sometimes, it is difficult to discern which parts of the page were contributed from which dynamic content spots.

Working with dynamic content spots:

You can add dynamic content to your custom theme by using either client-side or server-side logic.

You can add dynamic content to your custom theme's static content by using dynamic content spots. Dynamic content spots are hook points in your static markup such as the `theme.html` that delegates somewhere else to inject more markup at that spot the page. It can be delegated to any URI known to the system. For instance, it can point to a server-side component such as JSP or servlet.

To abstract the theme from direct links into server-side code, use the `dyn-cs: schema` with a unique identifier for a dynamic content spot. This allows for advanced features such as delegating to different JSPs on different pages. Use the following examples.

Including a named dynamic content spot mapping:

```
<a rel="dynamic-content" href="dyn-cs:id:newDynamicContentSpotName"></a>
```

Including a theme JSP directly:

```
<a rel="dynamic-content"
   href="res:/CustomThemeContext/themes/html/MyTheme/dynamicContent.jsp"></a>
```

Configuring and modifying named dynamic content spots

The dynamic content spot mappings can be defined as part of the module definition by using the `ref-id` attribute on a `dyn-cs` contribution type. Or, they can be defined for the theme and skin templates in a Resource Environment Provider (REP) named WP DynamicContentSpotMappings in the WebSphere Integrated Solutions Console. For more information, see *Creating a dynamic content spot*.

The following tables are an overview of the dynamic content spots that are most frequently used. Use the Theme Analyzer's Contribution Explorer to see all the dynamic content spots provided.

Table 443. Default dynamic content spots for the Portal 8.5 theme template

Name of the content spot	Defined in module	Value	Description
n/a		`co:config`	Starts the combiner data source, which injects the config markup that is identified in the module profile
n/a		`co:head`	Starts the combiner data source, which injects the head markup that is identified in the module profile.

Table 443. Default dynamic content spots for the Portal 8.5 theme template (continued)

Name of the content spot	Defined in module	Value	Description
85theme_asa		`mc:wp_authoring_actionbar@mvc:res:/wps/themeModules`	Provides the site analytics extension
85theme_commonActions	wp_dynamicContentSpots_85	`res:/wps/defaultTheme85/themes/html/dynamicSpots/commonActions.jsp`	Common actions that are on the banner, that is, Actions, Login, Logout
85theme_crumbTrail	wp_dynamicContentSpots_85	`mvc:res:{war:context-root}/themes/html/dynamicSpots/crumbTrail.jsp,smartphone@`	Navigation breadcrumb trail to display the page selection path
85theme_footer	wp_dynamicContentSpots_85	`res:/wps/defaultTheme85/themes/html/dynamicSpots/footer.jsp`	Footer of the page
85theme_head	wp_dynamicContentSpots_85	`res:/wps/defaultTheme85/themes/html/dynamicSpots/head.jsp`	HTML head element that provides the document title, styles, bookmark icon.
85theme_layout	wp_dynamicContentSpots_85	`lm:template`	Layout of the page

Table 443. Default dynamic content spots for the Portal 8.5 theme template (continued)

Name of the content spot	Defined in module	Value	Description
85theme_mobileNav	wp_dynamicContentSpots_85	`mvc:smartphone/tablet@res:/wps/defaultTheme85/th` CF06 `mvc:smartphone/tablet@res:/wps/defaultTheme8`	Navigation used on mobile devices such as smartphones and tablets. CF06 Each level of navigation is lazily loaded as the user clicks.
CF06 85theme_mobileNav_static	wp_dynamicContentSpots_85	`mvc:smartphone/tablet@res:wps/defaultTheme85/themes/html/dynamicSpots/mobileNavigation.jsp`	Navigation that is used on mobile devices such as smartphones and tablets. Each level of navigation is pre-loaded in the page markup.

Table 443. Default dynamic content spots for the Portal 8.5 theme template (continued)

Name of the content spot	Defined in module	Value	Description
85theme_pageModeToggle	wp_dynamicContentSpots_85	zmc:wp_toolbar@res:/wps/defaultTheme85/themes/html/dynamicSpots/pageModeToggle.jsp	Edit Mode and View mode buttons in the theme banner that is seen when the project menu is off.
85theme_primaryNav	wp_dynamicContentSpots_85	mvc:res:{war:context-root}/themes/html/dynamicSpots/navigation.jsp?type=primary&markup=smartphone@,tablet@	Navigation that is in the second level of navigation
85theme_secondaryNav	wp_dynamicContentSpots_85	mvc:res:{war:context-root}/themes/html/dynamicSpots/navigation.jsp?type=secondary&markup=smartphone@,tablet@	Navigation that is in the third level of navigation
85theme_sideNav	wp_dynamicContentSpots_85	mvc:res:/wps/defaultTheme85/themes/html/dynamicSpots/sideNavigation.jsp?startLevel=2&markup=smartphone@,tablet@	Nested side navigation at the third level
85theme_status	wp_dynamicContentSpots_85	mc:wp_status_bar@res:/wps/defaultTheme85/themes/html/dynamicSpots/status.jsp	Area that is located with the page layout that displays status, warning, and error messages to the user

Table 443. Default dynamic content spots for the Portal 8.5 theme template (continued)

Name of the content spot	Defined in module	Value	Description
85theme_topNav	wp_dynamicContentSpots_85	`mvc:smartphone/tablet@res:{war:context-root}/themes/html/dynamicSpots/navigation.jsp?type=top`	Navigation in the theme header, first level of navigation
wp_toolbar_dynspot	wp_toolbar		The tabbed toolbar that is at the beginning of the page while in edit mode
wp_project_menu_dynspot	wp_project_menu		The project drop-down menu in the theme header
wp_search_mobile_dynspot	wp_searchbar		Search input that is displayed on the subbanner
wp_preview_dynspot	wp_preview		**End Preview** button that is at the beginning of the page when you preview the page as a different user
wp_analytics_dynspot	wp_analytics		Provides the site analytics extension

Table 443. Default dynamic content spots for the Portal 8.5 theme template (continued)

Name of the content spot	Defined in module	Value	Description
wp_analytics_head_dynspot	wp_analytics		Provides the site analytics head extension

Table 444. Content spot values available in the skin template

Content spot value	Description
lm:control	Renders the layout control body.
lm:title	Renders the title of the portlet.
lm:description	Renders the description of the portlet.
portlet.link:portlet	Outputs the markup ** to allow page to position itself at a particular portlet markup with a fragment identifier.
wp_analytics_portlet_dynspot Defined in Module: `wp_analytics`.	Provides the site analytics extension for portlets

You can modify the dynamic content spot values two different ways.

- Dynamic content spots that are defined through module `wp_dynamicContentSpots_85` are part of the default theme and is cloned when you create your own theme. Rename the module in your theme, update the reference in the profile, and then you can change the values of the dynamic content spots.
- System dynamic content spots such as `wp_search_dynspot` can be changed by overriding dynamic content spots with the module framework. For example, you can create a module that redefines this dynamic content spot that requires the core module `wp_search`. Your dynamic content spot reference is used as a result.

Creating dynamic content spots:

There are two ways you can create dynamic content spots. You can use a module, or a resource environment provider.

"Creating a dynamic content spot with a module"
In most situations, use the modularized approach to define dynamic content spots. This approach is more flexible than defining them through Resource Environment Providers. You can also switch the dynamic content spot per page and override existing content spot when you define them with a module.

"Creating a dynamic content spot with resource environment providers" on page 2674
You can create a custom dynamic content spot to include in a theme template.

Creating a dynamic content spot with a module:

In most situations, use the modularized approach to define dynamic content spots. This approach is more flexible than defining them through Resource Environment

Providers. You can also switch the dynamic content spot per page and override existing content spot when you define them with a module.

About this task

A dynamic content spot is defined in `theme.html`. You must create a module with the ID `85theme_topNav` and then add that module to a profile. The following example renders the top navigation.

```
<a rel="dynamic-content" href="dyn-cs:id:85theme_topNav"></a>
```

For more information, see *Modules and dynamic content spots*.

Procedure

1. Define the module as a JSON file or as part of a `plugin.xml` file. The dynamic content spot ID is defined within the `ref-id` attribute on the sub contribution.
   - The following example shows the definition of a module by using the JSON format. To define the override module as a JSON file. Add the file in WebDAV at `fs-type1/themes/`*`YourTheme`*`/contributions/`.

     ```
     {
       "modules": [{
         "id" : "topnavoverlay",
         "prereqs": [{
           "id":"wp_dynamicContentSpots_85"
         }],
          "contributions": [{
           "type":"dyn-cs",
           "sub-contributions": [{
             "type":"markup",
             "ref-id":"85theme_topNav",
             "uris": [{
               "value":"res:/your/sample.html"
             }]
           }]
         }]
       }]
     }
     ```

     id This value can be any meaningful ID that is unique to the system.

     ref-id The ID of the dynamic spot to override. In this example, it is `85theme_topNav`.

     value A pointer to the markup for the dynamic spot. In this example, it points to an HTML file relative to the theme location in WebDAV.

     Note: If you want to point to a `.JSP` file, you must define this module within a `plugin.xml` .

     prereqs
     The name of the module that is required or overridden by this module.
   - You can also define the override module in a `plugin.xml` file.

     ```
     <extension point="com.ibm.portal.resourceaggregator.module" id="topnavoverlay" >
        <module id="topnavoverlay">
         <preqreq id="wp_dynamicContentSpots_85"/>
         <contribution type="dyn-cs">
           <sub-contribution type="markup" ref-id="85theme_topNav">
              <uri value="res:/your/sample.jsp" />
     ```

```
      </sub-contribution>
    </contribution>
  </module>
 </extension>
```

extension id
This value can be any meaningful ID that is unique to the system.

module id
This value can be any meaningful ID that is unique to the system.

sub-contribution ref-id
The ID of the dynamic spot to override. In this example, it is `85theme_search`.

uri value
A pointer to the markup for the dynamic spot.

prereqs
The name of the module that is required or overridden by this module.

2. Include the module in a profile. For example, in `newtopnav_profile.json`.

```
{{
  {
   "moduleIDs": [
   "topnavoverlay",
   ...
}}
```

The `newtopnav_profile.json` file must be copied to WebDAV at `fs-type1/themes/`*`YourTheme`*`/profiles/`. For more information, see *Create a module profile*.

3. Apply the profile to a page hierarchy. You can change the profile for the theme or a specific page to define the modules that are loaded. For more information, see *Changing the theme profile*.

Creating a dynamic content spot with resource environment providers:

You can create a custom dynamic content spot to include in a theme template.

Procedure

1. Log on to the WebSphere Integrated Solutions Console.
2. Go to **Resources** > **Resource Environment** > > **Resource environment providers** > **WP DynamicContentSpotMappings**.
3. Select **Custom Properties**.
4. Select **New**.
5. Enter a name to be used as the ID of the mapping. For example `newDynamicContent`.
6. Enter a value that is the URI to the dynamic content to be included by the mapping, similar to the following example:
   `res:/CustomThemeContext/themes/html/MyTheme/dynamicContent.jsp`
7. Select **OK**.
8. Save the changes.
9. Restart the portal.
10. Open the theme template file on WebDAV.

11. Locate where in the template you want to include the dynamic content and add the following line of code:

    ```
    <link rel="dynamic-content" href="dyn-cs:id:<your mappping ID>">
    ```

    Replace <*your mapping ID*> with the name that you gave the mapping, for example **newDynamicContent**.
12. Clear the browser cache and refresh the page that has the theme template that you modified to get the new dynamic content spot to render.

Using parameters

You can pass parameters to a JSP page that is rendering the dynamic content. If the page expects a parameter such as **text**, then you would locate where in the template you want to include the dynamic content and add the following line of code:

```
<link rel="dynamic-content" href="dyn-cs:id:<your mappping ID>+'?text=dynatext'">
```

Conditionally disable dynamic content spots:

You can control whether a dynamic content spot is rendered into the page or not based on the fact whether a certain module exists on the page or not.

When a particular module is turned on, you can choose to have a dynamic content spot display. You can use the mc schema as part of the dynamic content spot URI. The schema is defined in the following example, `mc:module-id@delegate-uri`.

The module-id is the module this URI is tested for. If the module is enabled, the dynamic content spot is rendered. The delegate-uri represents the dynamic content spot URI.

In the Portal 8.5 theme, the status bar component renders error, warning, or informational messages.

In the theme HTML template, the status bar component is rendered with the following dynamic content spot.

```
<a rel="dynamic-content" href="dyn-cs:id:85theme_status"></a>
```

This dynamic content spot maps to the following URI through the system module for the Portal 8.5 theme:

```
'wp_dynamicContentSpots_85': mc:wp_status_bar@res:{war:context-root}/themes/html/dynamicSpots/stat
```

The wp_status_bar module contains the JavaScript resources necessary to display the messages. The markup renders when the module is active.

Modules and dynamic content spots:

You can use the modularized framework for dynamic content spots to override spots that were defined through resource environment providers.

Dynamic content spots and markup contributions are both techniques for adding HTML to a theme, with one important difference. Markup contributions are always placed at the end of the body of the page. But, dynamic content spots render wherever they are placed in the theme HTML template.

When a module requires markup, use a markup contribution if it is not important where the module is in the page, such as with iWidget definitions. If the markup must appear in a particular spot in the theme, then a conditional dynamic spot is used.

Dynamic Content Spots on a page

Portal identifies dynamic content spots after it analyzes the profile of a page and all the modules in the profile that includes its `prereqs`. If any module within the hierarchy contains a dynamic content spot, the spot participates in the page rendering process. If the page's profile does not contain a dynamic content spot's ID that is used within the `theme.html`, nothing is rendered. If Portal identifies a dynamic content spot with a matching ID, it renders the dynamic content spot at the referenced URI.

To use the same dynamic content spot on two different pages, but with different markup, reference different profiles that contain a different module for the dynamic content spot.

In this example, there are two pages, Home and Applications. Home has an inline top navigation, and Applications has a fly-out navigation. The `theme.html` is identical and uses the following dynamic content spot

```
<a rel="dynamic-content" href="dyn-cs:id:85theme_topNav"></a>
```

Create two modules and two profiles to create the two pages with different markup.

Modules:

topNavModule
: Defines a sub contribution with **ref-id** `85theme_topNav` that points to a JSP provided by this module. This JSP renders the inline top navigation.

flyoutNavModule
: Defines a sub contribution with **ref-id** `85theme_topNav` that points to a JSP provided by this module. This JSP renders a fly-out navigation.

Profiles:

HomeProfile, set on Home
: It contains the `topNavModule`. It causes the inline top navigation to be rendered.

ApplicationsProfile, set on Applications
: It contains the `flyoutNavModule`. It causes the inline top navigation to be rendered.

Override dynamic content spots

You can override any dynamic content spot that was defined through resource environment providers. Dynamic content spots in modules overrule the dynamic content spots in resource environment providers. However, you can override dynamic content spots that were defined through modules. To render the dynamic content spots in the correct order, you must create a module that defines a `prereq` on the module that defines the previous dynamic content spot. For more information, see *Writing modules with a plugin.xml file*.

To override the default dynamic content spot for search `wp_search_dynspot`, you must create a module that defines a `prereq` on `wp_searchbar`. This module renders this particular dynamic content spot ID.

Dynamic content spot debugging:

Dynamic Content Spots add dynamic markup into static themes or skins. Sometimes, it is difficult to discern which parts of the page were contributed from which dynamic content spots.

The new Dynamic Content Spot Debugging was introduced to provide a visual indication in the page within your browser. The visual indicator operates two ways.

Default mode

This mode shows the original markup structure and displays the corresponding content sport for markup type when it is in focus. If a user hovers, the markup is highlighted with a background color and a label, which displays the URI of the corresponding spot. If the spot was configured through the `dyn-cs:` scheme the system also resolves the spot to its final URI, which is usually a `.jsp`. The resolved URI is shown as a link. This link does not provide a working URL for you to follow with your browser. You can copy the resolved URI for further use.

Inline mode

With this feature, you can easily spot if you have unnecessary dynamic content spots in your theme or skins. But, it does not preserve the overall design of your page. The advantage of this mode over the default mode is that it displays spots that do not contribute anything visual.

Limitations

The validation report shows errors, warnings, and informational messages about the theme and corresponding artifacts, such as profiles and modules. However, the validation report does not determine whether the profiles that are applied to each page are present in the theme.

When you enable the visualization feature for dynamic content spots, that setting does not persist across a server restart. Also, it applies to the current cluster member. It is not replicated to other cluster members.

Dynamic content spot debugging, development mode, reports, and client tracing do not persist across a cluster or a server restart either.

Layouts

You can apply ready-use layouts to your portal pages, modify the existing skins, or add your own custom layout to change how your pages display.

"Working with layout templates" on page 2678
You can add new layout templates so that they can be assigned to portal pages or update your ready-use layouts to modify the existing skins or change how your pages display.

"Adding a layout to the toolbar" on page 2680
You can add your own custom layout to the theme that can be selected on the site toolbar.

"Default layouts" on page 2681
The IBM WebSphere Portal Express 8.5 theme ships with predefined layouts that can be used as a basis for creating custom layouts.

Working with layout templates:

You can add new layout templates so that they can be assigned to portal pages or update your ready-use layouts to modify the existing skins or change how your pages display.

Layout templates are stored in the WebDAV filestore at the location `/fs-type1/themes/myCustomTheme/layout-templates`. To add a layout template and to assign it to portal pages, create a folder or copy and rename an existing layout template folder in the root folder. Include all CSS files in the theme that are used by the new layout template.

Layout templates control the layout and positioning of the content on a page. The static layout template is called `layout.html`. The author of the layout template defines the HTML fragment markup and CSS for the layout of the page. The HTML fragment uses microformat to specify containers and components, such as portlets and iWidgets to include in the page.

This example defines a two-column layout, with a hidden container for widgets, which participate in the wiring of the page, but are not visible themselves.

```
<div class="hiddenWidgetsDiv">
 <!-- widgets in this container are hidden in the UI by default -->
 <div class="component-container hiddenWidgetsContainer ibmDndRow wpthemeCol12of12 wpthemeFull" name=
 <div style="clear:both"></div>
</div>
<!-- this layout has two equally sized columns -->
<div class="wptheme2Col wpthemeEqual">
 <div class="component-container wpthemeCol wpthemePrimaryContainer ibmDndColumn wpthemeLeft wpthemeC
 <div class="component-container wpthemeCol wpthemeSecondaryContainer ibmDndColumn wpthemeLeft wpther
</div>
```

The meaning of the elements is as follows:

class="component-container"
: The static page parser recognizes the microformat to define containers in the page layout model that can contain components. Components can be portlets or widgets.

name="ibmMainContainer"
: The name attribute on a component container identifies that container uniquely to the page. The static page parser uses the name to correlate containers when it updates the page definition. Using consistent names across layout templates preserves your container contents when you switch between layouts on pages. There must always be a container that is named `ibmMainContainer` for the main content of the page.

name="ibmHiddenWidgets"
: When you create your own layouts, there must always be the one container at the beginning of the page named `ibmHiddenWidgets`.

name="headline"
: If there are containers other than `ibmHiddenWidgets` or `ibmMainContainer`, use `headline` for a header or banner across the page for smooth switching between your layouts and the default layouts.

name="secondary"
Use `secondary` for content that is secondary to the main content, such as a sidebar.

name="tertiary"
Use tertiary for content that supports the secondary content.

name="additional"
Use `additional` for content that supports the tertiary content.

name="footer"
Use `footer` for an item at the end of the page.

class="ibmDndColumn"
class="ibmDndRow"
This class is required for client-side drag-and-drop support. Mark your component container with one of these classes.

class="wpthemeCol"
class="wpthemeRow"
This class provides CSS positioning information. Mark your component container with one of these classes.

class="wpthemeHeadlineContainer"
class="wpthemePrimaryContainer"
class="wpthemeSecondaryContainer"
class="wpthemeTertiaryContainer"
These classes provide CSS sizing information. Mark your component container with one of these classes. These classes are optional. You can use one of your own classes if your container sizing requirements vary from what these provide.

class="wpthemeLeft"
This class provides CSS positioning information. Mark your component container with this class if it is one of multiple containers on the same vertical level and floats to stay on the same level.

class="wpthemeCol1of12"
class="wpthemeCol2of12"
class="wpthemeCol1of5"
class="wpthemeCol3of12"
class="wpthemeCol4of12"
class="wpthemeCol2of5"
class="wpthemeCol5of12"
class="wpthemeCol6of12"
class="wpthemeCol7of12"
class="wpthemeCol3of5"
class="wpthemeCol8of12"
class="wpthemeCol9of12"
class="wpthemeCol4of5"
class="wpthemeCol10of12"
class="wpthemeCol11of12"
class="wpthemeCol5of5"
class="wpthemeCol12of12"
These classes are CSS marker classes. They do not have any styles that are defined for them by default. But, they can be used in CSS selectors to vary the styles of content and portlets that can be dropped in this container. Mark your component container with one of these classes that are based on the width the container has relative to other containers on the same vertical level. These classes are referred to as grid relative width classes.

You can specify that a container takes up any number of width units of the page. See Relative width CSS classes for theme layouts.

class="wpthemeThin"
class="wpthemeNarrow"
class="wpthemeMedium"
class="wpthemeWide"
class="wpthemeFull"

These are CSS marker classes. They do not have any styles that are defined for them by default. But, they can be used in CSS selectors to vary the styles of content and portlets that can be dropped in this container. Mark your component container with one of these classes that are based on the width the container has relative to other containers on the same vertical level. These classes are referred to as semantic relative width classes. See Relative width CSS classes for theme layouts.

If you repeat similar sections in a layout, append 2 through n to the end of any of the names. For example, use `additional2` and `additional3` or use `headline2`, `ibmMainContent2` and `secondary2`. Use this naming convention for better transitions when you switch between layouts.

If you implement a complex layout template that requires its own JavaScript handling, ensure that the required JavaScript components are loaded and initialized by the theme. For best performance, build and minify all JavaScript used into a single, cacheable file that can be loaded one time and cached by the browser.

Adding portlets and iWidgets to layout templates

You can add portlets and iWidgets directly to the layout template definition by using the portlet microformat for static pages, or the iWidget definition specification. You can add portlets or iWidgets inside or outside of containers. The template is applied to the page as a copy. Therefore, every page that uses the template has a new instance of the portlet or iWidget.

Adding a layout to the toolbar:

You can add your own custom layout to the theme that can be selected on the site toolbar.

About this task

To add your own custom layout to the toolbar, first create a folder for your layout in `/fs-type1/themes/myCustomTheme/layout-templates` or copy and rename one of the existing layout folders. In the folder, create or modify your layout template `layout.html` file, and create or modify your `icon.gif` file for the visual representation of your layout as it appears on the toolbar.

These steps add the new layout to the site toolbar.

Procedure

1. Open the `layouts.json` file in WebDAV at `dav:fs-type1/themes/`*`myCustomTheme`*`/system`. If the file does not exist, create the file. The `layouts.json` registers the layout you created in the `layout.html` file with the toolbar and defines it further.
   a. Make sure that the file has the following content:

```
{
localizationPackageName:'com.ibm.bundles',
localizationBundleName:'Shelf', identifier:'label', items:   [
]
}
```

b. For each layout in `dav:fs-type1/themes/myCustomTheme/layout-templates`, add an entry like the following code sample to the JSON array in the file `layouts.json`. For example, here is an entry for a layout that is stored in WebDAV at `dav:fs-type1/themes/myCustomTheme/layout-templates`:

```
{'label':'My Layout','url':'dav:themelist/myCustomTheme/layout-templates/myLayout/','id':'myLayout',
  'thumbnail':ibmCfg.themeConfig.themeRootURI+'/layout-templates/myLayout/icon.gif','help':''}
```

2. Optional: As an alternative to step 1, you can add titles and descriptions of the layout as part of the layout entry.

```
{
    "items": [
    {
        'label': 'Complete',
        'url': ibmCfg.themeConfig.themeWebDAVBaseURI + 'layout-templates/Complete/',
        'id': 'Complete',
        'thumbnail': ibmCfg.themeConfig.themeRootURI + '/layout-templates/Complete/icon.png',
        'help': '',
        "titles": [{
            "value": "Full Page Layout",
            "lang": "en"
        }],
        "descriptions" : [{
            "value": "Full Page Layout Description",
            "lang": "en"
        }]
    },
    ...
    ]
}
```

3. Refresh your browser cache.
4. Reload the page to view the layouts in the **Page** > **Layout** tab of the site toolbar.

Default layouts:

The IBM WebSphere Portal Express 8.5 theme ships with predefined layouts that can be used as a basis for creating custom layouts.

1 column layout
This layout has one main container.

1 row 2 column unequal layout
This layout has one row across the whole page with two unequal columns underneath, one column is larger than the other column.

1 row 3 column equal layout
This layout has one row across the whole page with three equally sized columns underneath.

2 column equal layout
This layout has two equally sized columns.

2 column left layout
This layout has two columns with one column assigned a smaller width.

2 column right layout
This layout has two columns with one column assigned a smaller fixed width.

2 row layout
This layout has two rows that go across the whole page.

3 column center layout
This layout has 3 columns that take up 20%, 60%, and 20% of the space across the page.

3 column equal layout
This layout contains three equally size columns.

Top column 2 column unequal layout
This layout has a column on top with two columns underneath, one column is larger than the other column.

Top column 3 column center layout
This layout has one column on top of three unequal sized columns underneath.

Skins

You can apply ready-use skins to your portal pages, modify the existing skins, or add your own custom skin to change how your pages display.

"Skin templates"
The static skin template `skin.html` is located in the root directory of the skin folder of the WebDAV file store. The `skin.html` file provides the full markup for decoration around a portlet or iWidget. As with theme templates, you can use dynamic content spots to add dynamic elements to the skin template at run time.

"Creating skins" on page 2684
You can create global or theme-based skins to customize a theme. A successful approach to creating custom skins is by copying one of the default skins files, then adding images, JavaScript files, and other custom resources.

"Default skins" on page 2687
The default theme includes these skins that can be used as a basis for creating custom skins: Hidden, Standard, and NoSkin.

Skin templates:

The static skin template `skin.html` is located in the root directory of the skin folder of the WebDAV file store. The `skin.html` file provides the full markup for decoration around a portlet or iWidget. As with theme templates, you can use dynamic content spots to add dynamic elements to the skin template at run time.

The portal provides a base skin template and localized skin templates.

The `skin.html` file is located in the root directory of the skin on WebDAV `\fs-type1\skins\skin-name\`, and there are localized `skin.html` files located within the `nls` directory under the skin `\fs-type1\skins\skin-name\nls\`. Modify the skin template files by using the WebDAV entry point `fs-type1`. When you use this entry point, your changes to the skin template are immediately reflected with a browser refresh.

Root skin template

In a default WebSphere Portal installation, the portal does not render the template file `skin.html` located in the root directory of the theme. Instead, this file links to

the localized templates, and the portal renders the appropriate localized template. The links to the localized templates are at the beginning of the root template. They have the following form:

```
<a rel="alternate" href="nls/skin_locale_code.html" hreflang="locale_code" class="wpthemeDisplayNone"></a>
```

An example of a link to the English template file is as follows:

```
<a rel="alternate" href="nls/skin_en.html" hreflang="en" class="wpthemeDisplayNone"></a>
```

You can place the wpthemeDisplayNone class on these links so that the browser does not expose the anchor element in the user interface. This class makes sure that accessible technologies, such as screen readers, do not encounter these links in the tab order.

If you do not want to use localized skin templates, you can remove these links from the beginning section of the `skin.html` template in the root directory. If you remove the links, the portal renders the root template.

The root skin template also includes Apache Ant scripting in the following form:

```
${bundle_name:bundle_key:character_encoding}
```

The **`character_encoding`** replaces special characters with the escape sequence determined by the specified encoding. The available types of encoding are XML or JSON. You can chain multiple instances of encoding by or as follows:

```
${bundle:key:json:xml} or ${bundle:key:xml:json}
```

You can use the Apache Ant build framework to generate localized templates based on this root template. This framework can be useful if you want to update one template during development and then generate the templates. If you want to use only the root template, replace the Ant scripting with the preferred text that you want to be rendered. You can learn more about the Ant build tool at the Apache Ant Welcome page.

Localized skin templates

In a default portal installation, the theme renders content by using the localized skin templates. These templates are located in the `nls` subdirectory under the skin directory on WebDAV. These files have the locale code appended to the end of the template name, for example `skin_en.html` for English. These templates have translated static text inline within the template.

When you use the localized skin templates and want to view your changes, update the template that the portal renders in the browser. For example, if your preferred language is English, update the `fileskin_en.html` file.

Server-side dynamic content spots

`<a rel="dynamic-content" href="lm:title">`
This inserts the portlet title into the skin.

`<a rel="dynamic-content" href="lm:control">`
This renders the content of the portlet.

Client-side dynamic content spots

Client-side content spots are replaced at page load time or at run time through JavaScript by **`RuntimeSkinModel`** and the live text service.

class='lm-dynamic-icon'

By default, the skins do not contain the lm-dynamic-icon spot. If you want support for a dynamic icon, you need to add an image tag to the titlebar markup in the skin HTML template.

For Example:

```
<img class="lm-dynamic-icon" src=""/>
```

This provides a client-side way for changing the icon for the portlet or iWidget in the skin dynamically. For example, you can add the following code sample to set the icon dynamically. The second parameter sent into the setDynamicContent function is a URL that is the `src` of the icon <img> element.

```
var skinNode = com.ibm.mashups.enabler.runtime.skin.Factory.getRuntimeSkinModel().find("somePortletID");
 skinNode.setDynamicContent(com.ibm.mashups.enabler.runtime.skin.Constants.DYNAMIC_CONTENT_ICON, "http://www.mysite.com/icon.gif");
```

Note: The somePortletID function fetched from the runtime skin model is a string ID of the layout node for which the title or icon is changed. The profile used on the page must include Enabler in view mode as it is required to fetch the skin node and set the dynamic content.

Note: Enabler has been deprecated.

class='lm-dynamic-title'

This content spot changes the title of the skin dynamically at run time.

For example, to set the title, the second parameter is a string that is inserted as the new title.

```
skinNode.setDynamicContent(skinConstants.DYNAMIC_CONTENT_TITLE, "My new title")
```

Where:

`var controlId`: the string ID of the layout node for which the title or icon is changed.

`var runtimeSkinModel`: `com.ibm.mashups.enabler.runtime.skin.Factory.getRuntimeSkinModel()`.

`var skinNode`: `runtimeSkinModel.find(controlId)`.

`var skinConstants`: `com.ibm.mashups.enabler.runtime.skin.Constants`.

Related information:

Apache Ant Welcome page

Creating skins:

You can create global or theme-based skins to customize a theme. A successful approach to creating custom skins is by copying one of the default skins files, then adding images, JavaScript files, and other custom resources.

Do not modify the default skins directly, because this skin can be updated by service fix packs and override your changes.

You can create skins in the following scopes:

Theme-based

You can scope a skin to one particular theme. Use this scope if the skin relies specifically on code within the theme or has a specific function that

is only useful in that particular theme. These skins are located under the root folder of the theme, and create a specific link between the skin and theme.

This is the recommended approach.

Global
The global scope where the skin is not specific for any theme. Global skins are located in the `skins` folder under the WebDAV root.

All skins created in Portal can be access by the WebDAV `skinlist` entry point.

"Create a theme-scoped skin"
You can create a custom skin that is scoped to a specific theme when you copy an existing ready-use choice to use as the base for your new skin.

"Add static resources to a skin" on page 2687
You can add images, URLs, and other static resources to your skin.

Create a theme-scoped skin:

You can create a custom skin that is scoped to a specific theme when you copy an existing ready-use choice to use as the base for your new skin.

You can define new skin resources manually or by using the Manage properties portlet. After the skin is assigned to the theme, you can define the skin and apply it to specific portlets or use it as the theme default.

"Creating new skin resources"
You can begin to create a custom skin that is scoped to a specific theme by creating new skin resources.

"Defining the skin and assigning it to a theme using **Manage properties**" on page 2686
When you create new skin resources, you can define them so that you can assign them to a theme.

"Defining the skin and assigning it to a theme manually" on page 2686
When you create new skin resources, you can define them so that you can assign them to a theme.

Creating new skin resources:

You can begin to create a custom skin that is scoped to a specific theme by creating new skin resources.

Procedure

1. Mount the `/fs-type1/` WebDAV entry point with your WebDAV client:
   `/wps/mycontenthandler/dav/fs-type1/`

   If you changed the `wps` folder, replace it with your `WpsContextRoot` value.
2. Locate your custom theme under the `themes` folder.
3. Under the theme root folder, find the `skins` folder. If it is not there, then create it.
4. Copy a skin that you want to use as the base to your local system. If the `skins` folder did not exist previously and you created it, go to the `skins` folder in the global scope, `dav:fs-type1/skins`. Copy a skin from that location that you want to use as the base.
5. After you copy the base skin folder to your local system, rename the folder.

6. Copy the new folder to the skins folder under your custom theme, dav:fs-type1/themes/*custom_theme*/skins.

Defining the skin and assigning it to a theme using **Manage properties***:*

When you create new skin resources, you can define them so that you can assign them to a theme.

About this task

You can define the skin and assign it to a theme by using **Manage properties** or manually. Learn more about Defining the skin and assigning it to a theme manually.

Procedure

1. In the Theme Manager, select your custom theme and click **Manage properties**.
2. Click the **Skins** tab. Then, click **Add skin**.
3. Enter a title for your new skin. You can also set and modify other skin properties in the **Localization**, **Metadata**, and **Advanced** tabs.
4. In the **Advanced** pane for the skin, ensure that the static content root matches the path to the directory that contains the skin in WebDAV.
5. Click **Done**.

Defining the skin and assigning it to a theme manually:

When you create new skin resources, you can define them so that you can assign them to a theme.

About this task

You can define the skin and assign it to a theme by using **Manage properties** or manually. Learn more about Defining the skin and assigning it to a theme using **Manage properties**.

Procedure

1. Use the following code snippet as a template for the XMLAccess skin definition to import, where *theme_context*, *skin_folder_name*, *skin_unique_name*, *skin_title*, *theme_folder*, *skin_folder_name*, and *theme_unique_name* are the unique values for your theme and skin.

   Note: This template creates a title in English only, but you can add more locales.

```
<?xml version="1.0" encoding="UTF-8"?>
    <request
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" build="wp80" version="8.0"
     xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
     type="update" create-oids="true">
     <portal action="locate">
      <skin action="update" active="true" context-root="theme_context" default="false" domain="re
        resourceroot="skin_folder_name" type="default" objectid="skin_unique_name" uniquename="sk
      <localedata locale="en">
       skin_title
      </localedata>
       <parameter name="com.ibm.portal.skintype" type="string" update="set"><![CDATA[template]]><
       <parameter name="com.ibm.portal.skin.template.file.name.html" type="string" update="set"><
       <parameter name="com.ibm.portal.skin.template.ref"
         type="string" update="set"><![CDATA[dav:fs-type1/themes/theme_folder/skins/skin_folder_n
```

```
  </skin>
  <theme action="update" uniquename="theme_unique_name">
   <allowed-skin skin="skin_unique_name" update="set"/>
  </theme>
 </portal>
</request>
```

2. Import the XMLAccess skin definition by using the command line or the Import XML portlet.

Add static resources to a skin:

You can add images, URLs, and other static resources to your skin.

About this task

When you create a file for the skin on the `/fs-type1/` entry point, there is a link that shows this file through the `/skinlist/` entry point. This link makes it possible to use the path defined to the image as `skinlist` instead of `fs-type1`. You can still use an absolute path to the `/fs-type1/` entry point if you want, such as `<img src="http://server_name:port_number/wps/mycontenthandler/dav/fs-type1/skins/myskin/images/logo.png"/>`.

Procedure

1. Mount the `/fs-type1/` WebDAV entry point with your WebDAV client:
   `/wps/mycontenthandler/dav/fs-type1/`

   If you changed the *wps* folder, replace it with your `WpsContextRoot` value.
2. Add images or other static resources to your skin.
   a. Copy the required static resources, such as images or JavaScript files, into the root directory of your skin in WebDAV.
   b. Organize your files in subdirectories as necessary.
   c. Use server relative or absolute URLs to reference the resources in your `skin.html` file. For example, if you create a folder named `/images` that contains a file named `logo.png`, you can use the following references:

      Server relative URL

      Absolute URL

Default skins:

The default theme includes these skins that can be used as a basis for creating custom skins: Hidden, Standard, and NoSkin.

Hidden skin
Is the default skin. In view mode, this skin does not display the title bar or any decorations. In edit mode, the skin displays the title bar and decorations around the portlet. This skin was created for WCM content that is exposed on external facing websites, where edit mode is not likely to be shown to users so the title bar is hidden. However, when an administrator or user that has access to edit the WCM portlet, they can put a page into edit mode and configure the portlet.

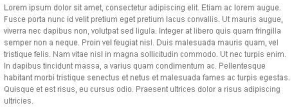
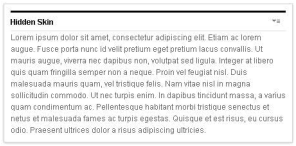
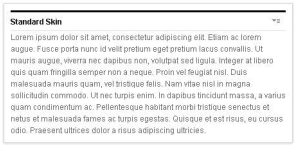
Figure 1. Hidden Skin in view mode. 

Figure 2. Hidden Skin in edit mode. 

Standard skin

Displays the portlet title bar in both edit and view mode. The title bar shows the title of the portlet and the skin action menu. This skin imitates the default skin from past releases.

Figure 3. Standard Skin. 

NoSkin skin

The NoSkin skin does not show the title bar or any decorations around the portlet. This skin is used if you do not need to show the skin actions.

Simple menu framework

You can provide menu feeds in JSON format instead of XML format. The operations can then parse the JSON feed without requiring the XML parsing support in the Dojo toolkit.

"Client-side framework for simple menus"
You can enable simple menus in the client with four modules.

"Server-side framework" on page 2692
A plug-in receives HTTP requests for specific menus from a client and responds with the menu feed in JSON format. The response is marked as content type `application/JSON`.

"Changing menu item visibility" on page 2701
Whether the menu items are visible is controlled in a layered way by client-side JavaScript and markup. The default value for visibility, if not specified, is Boolean true.

"Menu styles" on page 2702
You can vary the appearance of the items in menus that are delivered through the JSON menu framework.

Client-side framework for simple menus:

You can enable simple menus in the client with four modules.

To use the simple menus in your theme, you must add the main module `wp_simple_contextmenu_main` to your theme profile on the current page. All required resources are included on the page.

The simple menu framework includes four modules:

- `wp_simple_contextmenu_main`
  - `wp_simple_contextmenu_js`
  - `wp_simple_contextmenu_css`

- wp_simple_contextmenu_template

Open the menu with the JavaScript function **wptheme.contextMenu.init**. For more information, see the *JsDoc*.

To register a simple menu so that it opens when a user clicks the HTML element, use the following example.

```
<span role="button" aria-haspopup="true" class="wpthemeMenuFocus"
          onclick="if (typeof wptheme != 'undefined') wptheme.contextMenu.init({ 'node': this, mer
          <span class="wpthemeUnderlineText" id="wpContextMenu">My Menu</span>
      </span>
```

This example has the `pageAction` ID, which triggers a request to the server to load a JSON file with the same id. For more information, see Server-side framework.

You can create as many menus as you want in a custom theme.

To force the regeneration of menus, use the JavaScript command:
i$.fireEvent('wptheme/contextMenu/invalidate/all');

When a session timeout occurs, the page refresh returns to the login page.

Module details

The four modules in the simple menu framework provide all the information that you need to render the menus. For flexibility, the modules, except for the JavaScript module, are configured as theme modules. They are included in each theme individually. When you are creating or cloning your own theme, you must copy all the required resources to your theme.

If you copy the default theme, that theme includes all the required resources.

The following list describes all resources that are required in your theme for various modules .

- wp_simple_contextmenu_js (system module)
- wp_simple_contextmenu_main (theme module)
  - meta module, no resources referenced
- wp_simple_contextmenu_css (theme module)
  - theme root directory for static resources/css/wp_simple_contextmenu.css
  - theme root directory for static resources/css/ wp_simple_contextmenu.css.uncompressed.css
  - theme root directory for static resources/css/wp_simple_contextmenuRTL.css
  - theme root directory for static resources/css/ wp_simple_contextmenuRTL.css.uncompressed.css
  - theme root directory for static resources/css/default/contextmenu.css
  - theme root directory for static resources/css/default/ contextmenu.css.uncompressed.css
  - theme root directory for static resources/css/default/ contextmenuCommon.css
  - theme root directory for static resources/css/default/ contextmenuCommon.css.uncompressed.css
  - theme root directory for static resources/css/default/contextmenuRTL.css

  - theme root directory for static resources/css/default/contextmenuRTL.css.uncompressed.css
- wp_simple_contextmenu_template (theme module)
  - theme root directory for static resources/menuDefinitions/templates/simpleMenuTemplate.html

Templates

The menu provides a default template that is embedded into the page markup and contributed through the module `wp_simple_contextmenu_template`. If no inline template or not templateID is passed when initializing the menu, the framework reverts to the default template.

You can define your template with an inline template, which is embedded with the menu markup.

```
<span role="button" aria-haspopup="true" class="wpthemeMenuFocus"
          onclick="if (typeof wptheme != 'undefined') wptheme.contextMenu.init({ 'node': this, menuI
          <span class="wpthemeUnderlineText" id="wpContextMenu">My Menu</span>
          ... <your template markup appears here> ...
      </span>
```

Or, you can use a referenced template through an ID. When calling the init method of the menu, you can pass a templateID as part of the parameters. This ID is resolved as HTML element ID in the DOM and must point to a valid template.

```
<span id="exampleTemplateId" class="wpthemeMenuLeft">
    <div class="wpthemeMenuBorder">
        <div class="wpthemeMenuNotchBorder"></div>
        <!-- define the menu item template inside the "ul" element.  only "css-class", "description"
        <ul class="wpthemeMenuDropDown wpthemeTemplateMenu" role="menu">
            <li class="${css-class}" role="menuitem" tabindex="-1"  ><span class="wpthemeMenuText" >
        </ul>
    </div>
    <!-- Template for loading -->
    <div class="wpthemeMenuLoading wpthemeTemplateLoading">${loading}</div>
    <!-- Template for submenu -->
    <div class="wpthemeAnchorSubmenu wpthemeTemplateSubmenu">
        <div class="wpthemeMenuBorder wpthemeMenuSubmenu">
            <ul id="${submenu-id}" class="wpthemeMenuDropDown" role="menu"><li role="menuitem" tabin
        </div>
    </div>
</span>
```

The template contains 3 sub templates that are identified through classes.

Menu item template
: Defined through the class `wpthemeTemplateMenu`. To generate each individual menu entry, items this DOM node contains are used as a template.

Loading template
: Defined through the class `wpthemeTemplateLoading`. Used to display if the menu content is being loaded.

Submenu template
: Defined through the class `wpthemeTemplateSubmenu`.

The template can also contain a few replacement variables. From within the menu item template, there are three variables.

${css-class}
Replaced with the classes used for this entry.

${description}
Localized description of the menu item.

${title} Localized title of the menu item.

In the loading template, there is one variable.

${loading}
Loading text to be displayed while data is fetched from the server.

In the submenu template, there is one variable.

${submenu-id}
Required to define a unique ID for a newly generated submenu.

Example

This example shows the close handler, another templateId, and alignment.

```
addClass(control.parentNode, VALUE_SELECTED);
            args = {
                "node": control.parentNode,
                "menuId": menuID,
                "jsonQuery": menuQuery,
                "params": {
                    "templateId": "exampleTemplate",
                    "alignment": "right"
                },
                "onClose": function() {
                    removeClass(control.parentNode, VALUE_SELECTED);
                }
            };
            wptheme.contextMenu.init(args);
```

Simple menu extensions

The framework supports extra extensions as part of the module `wp_simple_contextmenu_ext`. The following extensions are available.

actionUrlTarget
You can define the target window within the browser DOM in which a menu action is run. Use this to run a menu action within another iframe of the page for instance. This module is used within the toolbar theme.

badge This extension provides support for displaying badges for each `menuitem` of the `contextmenu`. A badge is a number that is displayed in a graphical box with color. Badges can easily be enabled by setting the metadata `badgeUrl` or `badgeData` on the metadata for the `menuitem`. The feed that is returned from the `badgeUrl` must be of type JSON and must have two JSON elements on the root object.

count The number to be displayed.

level String, which can be either `error`, `warn`, or `info`.

As an alternative to the `badgeURL`, `menuitem` can also contain the `badgeData` element. The `badgeData` element must be a JSON object with the same elements as the feed that is returned from badgeURL. The following is a `badgeURL` example.

```
{
  "type": "StaticMenuitem",
  "id": "myEditFct",
  ...
  "metadata": {
    ...
    "badgeUrl": "?uri=theme-validation:count"
  }
}
```

The following is an example of `badgeData`.

```
{
  "type": "StaticMenuitem",
  "id": "myEditFct",
  ...
  "metadata": {
    ...
    "badgeData": {
      "count": "10",
      "level": "error"
    }
  }
}
```

Server-side framework:

A plug-in receives HTTP requests for specific menus from a client and responds with the menu feed in JSON format. The response is marked as content type `application/JSON`.

The request URL for the menu feed provider is under the `/wps/contenthandler` or `/wps/mycontenthandler` URLs, depending on whether the request is coming from a currently-logged-in user context. The request URL might carry portal URL navigational state, but does require a specific query string to access the menu feed provider. The specific query string to get to the default JSON feed provider is:

```
?uri=menu:<specific menu name>&navID=<navigation node OID or custom unique name>
[&windowID=<portlet window control ID on the page> ]
```

where:

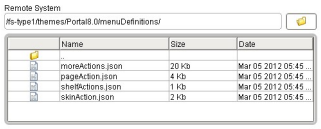
navID This parameter is either the serialized string-format ObjectID or the custom unique name of a portal navigation node (a portal page), which is being displayed when the menu is being requested. This parameter provides a context for the menu feed provider to use when you build the menu, so we know which theme to use, because the theme is tied to the page either explicitly or by inheritance.

windowID
This parameter is optional and is the serialized string-format ObjectID of a portlet window control on the page that is specified by `navID`. This parameter is only necessary for menus that contain portlet-level actions, in other words, `skin menus`. This parameter must be a serialized ObjectID. A custom unique name does not work for the portlet window ID.

WebSphere Portal Express provides three ready-to-use menu definition files with the Portal 8.5 Optimized Theme:

- `pageAction`
- `skinAction`
- `moreActions`

These files exist in WebDAV, under a `menuDefinitions` directory within the root for the theme root. The root is set in theme metadata under the metadata entry name `com.ibm.portal.theme.template.ref`. Other files in that directory are JSON syntax but are not used by the JSON menu framework



New starting in version 8.0.0.1: It is possible to name a menu ID (the *specific menu name* after the `?uri=menu:` in the request) which does not exist. Before 8.0.0.1, this menuID would result in an error message about an invalid or non-existent menu name being requested. In 8.0.0.1 and later, this menuID is treated as though a menu file with that name did exist, with the contents being an empty array ("[ ]"). A menu feed response is then generated that consists only of processing of the dynamic menu contributions or JSON subcontributions for the current theme profile, which are tagged with a `ref-id` matching the menu name on the request.

Important: The order of the items that are dynamically added to a menu is not controlled or guaranteed in any way. Although in testing the order was consistent between server restarts on the same system, it was found to be inconsistent across different operating systems. Especially between IBM JVMs and those not from IBM. If it is necessary to strictly control the order of the menu items, then use the explicit `"type":"ModuleRef"` entry in a menu definition file to explicitly insert the menu item, rather than relying on the dynamic contribution mechanism.

JSON menu definition file syntax

For more information about JSON itself, see the JSON home page.

The JSON syntax of a menu definition file consists of an array of JSON objects, where the array is indicated by outer enclosing opening and closing brackets `[ ]`. Each JSON object is enclosed in braces ({ }) and separated by commas:

```
[
   // optional one-line comment
   {
     JSON object 1 (first menu item)
   },
   /*
     optional multi-line comment
   */
   {
     JSON object 2 (second menu item)
   },
   ...
]
```

You can add comments in this file by using a portal-specific extension to JSON syntax.

Note: The minimum valid JSON menu definition file consists of an opening and closing array bracket ([]). An empty file results in an illegal syntax exception.

The final object in the array does not have a comma between its curly brace and the ending array closing bracket.

The order of the objects in the array within the menu definition file determines the order of appearance of the items in the menu on the client.

Each object within the array is an individual self-contained menu item definition, or a reference to a theme optimization module that can provide more in-line JSON menu definition file markup. This contributed JSON is included in-line as though it was in the menu file itself.

Each object consists of multiple comma-separated JSON members, each of which has a name and a value that are separated by a colon. The member name is always a string in quotation marks. The values can be strings, Booleans, nested objects, or arrays, depending on the member name. Some member names can have only certain values, as defined in the following list.

Each menu item definition must have a `type` entry, which defines that particular menu entry item.

Important: All entry names are case-sensitive.

The following menu item names and values are acceptable values for the type entry.

"type" : "Header"

Defines a label for subsequent entries in the menu. Typically displayed on the client UI as highlighted and outdented, although this menu item is controlled by the style that is applied to the menu items. A header in the menu is typically not clickable.

More title object entries, within the **titles** value, for other languages can be added.

An `itemClass` member can be added for controlling the appearance of the header.

```
{
"type" : "Header",
"titles" : [{"lang":"en","value":"<English text>"}, {"lang":"de", "value" : "German text"},...]
}
```

"type" : "Separator"

Defines a separator between menu items. A separator might show as a space, a line, or other appearance, depending on the style that is applied to the menu.

A separator does not typically need any other members, although an `itemClass` can be added for appearance control.

```
{
"type" : "Separator",
}
```

"type" : "DynamicMenuitem"

Potentially clickable and having an action the `DynamicMenuitem` item has an `id` member whose value is a plug-in name. The menu feed provider uses this ID to retrieve an instance of the named operation, which is then queried for sufficient information to build the menu feed content for this `menuitem`. Several plug-ins are supplied ready-to-use by WebSphere Portal Express for use in the default menu definitions, and these can be reused by custom-written menus and themes.

This plug-in provides its own indicator of whether it is active, including access control permission for the current user, and provides its own

localized title and optionally a description, and an actionHttpMethod value, for the menu feed provider to use when you build the menu for this item. The OperationURI for this operation plug-in becomes the actionUrl in the corresponding menu entry. Other object members are allowed as well, including actionFn, actionHttpMethod, visibilityFn, itemClass, and metadata. A markupID member can be added to create an ID on the html tag for the resulting menu item.

If the isActive() method returns false when called by the menu feed provider code, the menu item is present in the menu feed, but has a `"visibility" : false` boolean member added to the feed. This boolean member indicates to the client-side code to not present this operation to the user, and the client-side code does not include this item in the final rendered menu.

Optionally, a moduleArgs member might also be specified. This member is a string of query parameter-format names and values that are separated by ampersands (&). If present, these arguments are passed to the plug-in when the feed provider accesses the plug-in to build the current menu entry.

```
{
"type" : "DynamicMenuitem",
"id" : "operations.framework.plugin.name"
}
```

"type" : "StaticMenuitem"

Potentially clickable and having an action. This item is typically used to insert a menu item that has a client-side implementation, rather than a server-side implementation.

Allows the menu feed definition file author to completely specify an arbitrary menu item entry. All necessary information must be provided by the menu definition file, because there is no corresponding operation for a `StaticMenuitem`. The **id** parameter is optional and is ignored by the menu feed provider code for a `StaticMenuitem`, although it is not flagged as a syntax error if it is present.

Other optional members might be added as needed.

```
{
"type" : "StaticMenuitem",
 "titles" : [{"lang" : "en", "value" : "My English menu item text"},
        {"lang" : "de", "value" : "Mein menu item text auf Deutsch"},
         ...
        ],
"descriptions" : [{"lang" : "en", "value" : "My English menu item longer description flowing beautiful prose"},
             {"lang" : "de", "value" : "Mein menu item longer description flowing beautiful prose auf Deutsch"},
             ...
             ],
"actionUrl" : "http://www.yourco.com/wps/myportal/some_useful_url",
"actionHttpMethod" : "POST",
"actionFn" : "client_method_to_override_actionUrl",
"metadata" : {
        "navID" : "${navID}",
        "some_name" : "some_value",
        "some_other_name" : "${SubVar_From_Request_Query_Parms}"
        }
"markupId" : "my.item.markupId"
}
```

"type" : "ModuleRef"

A menu definition file has an **id** member whose value is the name of a plug-in. This plug-in must have a contribution of the menu type, and a subcontribution of the JSON type. The menu feed provider uses the value of the **id** member in this JSON object to retrieve a reference to the JSON subcontribution within the menu contribution for that module. This JSON subcontribution must be valid stand-alone JSON menu definition markup,

including the surrounding array opening and closing brackets. The menu feed provider strips the array opening and closing brackets and inserts this contributed markup in-line into the menu feed response as though it is included in the main definition file.

Points by name at a plug-in that provides more JSON format and menu definition file syntax markup, which the feed provider places inline in the menu feed and that replaces the `ModuleRef` entry.

Optionally, a `moduleArgs` member can be added. If present, the value of the `moduleArgs` member is passed as arguments to retrieve the JSON menu definition file markup from the plug-in.

```
{
"type" : "ModuleRef",
"id" : "Theme Optimization Framework module name"
}
```

"type" : "Submenu"

Defines a placeholder in the current menu where a new sublevel menu is attached. This item allows for multiple tiered menus. There is no enforced limit to the nesting level. The submenu is retrieved by an independent additional menu request to the JSON menu feed provider when the submenu entry in the displayed menu is hovered over.

The menu is appended to the one side as screen position dictates. A submenu is fetched in a separate subsequent request by the client.

A submenu names the source of its menu content with either an `id` or a `moduleId` member, and must have a `titles` member to provide the text for the placeholder menu item, and might optionally have a `descriptions` member.

The only difference between these two members is how the next request is processed, when the submenu is expanded and the client code retrieves the expansion menu feed in a new HTTP request:

- For an `id`, the next request is treated as naming a menu definition file where the `id` value is the file name and the extension is `.json`.
- For a `moduleId`, the next request is treated as naming a plug-in that provides the necessary JSON markup. This request is accessed as though a menu definition file specified it.

  ```
  {
  "type" : "ModuleRef",
  "id" : "moduleId_value"
  }
  ```

A `moduleArgs` member might also be added. If present with a `moduleRef` member within a `SubMenu` entry, the value of the `moduleArgs` member is appended to the URL that is built as the ID of the menu item. This value is used as the menu reference from the client for retrieving the menu markup for the cascaded submenu.

```
{
"type" : "SubMenu",
"id" : "name_of_submenu_definition_JSON_file",
 "titles" : [{"lang" : "en", "value" : "My English sub-menu item text"},
      {"lang" : "de", "value" : "Mein sub-menu item text auf Deutsch"},
      ...
      ],
"descriptions" : [{"lang" : "en", "value" : "My English sub-menu item longer description flowing beautiful prose"},
      {"lang" : "de", "value" : "Mein sub-menu item longer description flowing beautiful prose auf Deutsch"},
      ...
      ]
 }
or
 {
```

```
"type" : "SubMenu",
"moduleId" : "name_of_theme_opt_framework_module_which_contributes_submenu_definition_JSON_",
...
}
```

Valid members in a JSON menu definition file

Member name	Value and example syntax	Comments
type	"Header", "Separator", "DynamicMenuitem", "StaticMenuitem", "ModuleRef", Submenu"	Determines the type of menu item that is created by this menu definition file object
`id`	String	For a `DynamicMenuitem` or a `ModuleRef`, the `id` member is required. For a Submenu, either `id` or `moduleId` is required. **DynamicMenuitem** The ID is the name of the plug-in that is accessed by the menu feed provider to get the `actionUrl` (which is the OperationURI of the operation), the localized title and description, and the `isActive` method of the operation, which indicates whether the action is active and also accessible by the current user. The `actionUrl` is sent to the server in a separate request if the user clicks this menu item in the rendered menu. **ModuleRef** The ID is the name of the plug-in that is accessed to retrieve more menu file definition markup. **Submenu** The `id` is present, it is the name of the menu that is requested by the client in an HTTP menu request to create the submenu list of items.
`titles`	Array of objects, each of which has these members: a "lang" defines the language for that title entry, and a "value" contains the string for that language for the title. `"titles" : [ {"lang":"en", "value":"Title in English"}, {"lang":"de", "value":"Title auf Deutsch"}, ... ]`	Used only for header, `StaticMenuitem`, and Submenu entries. The list of languages that are provided covers only the necessary languages that the users of a portal might need. A `DynamicMenuitem` retrieves the title from the corresponding plug-in. The plug-in is required to implement the Localized interface, and provide localized strings for appropriate languages. A Separator has no text that is associated with it. A `ModuleRef` is replaced by other markup.
`descriptions`	Same as titles. `"descriptions" : [ {"lang":"en", "value":"Title in English"}, {"lang":"de", "value":"Title auf Deutsch"}, ... ]`	Optional for all types. If present, this member is interpreted by the default client code as hover help text over the menu item.
`itemClass`	String	Optional for all types. This member is the style class name that is applied to the menu item. If present, this is a class name that is present in the style sheet that is associated with the theme for the menu.

Member name	Value and example syntax	Comments
`enabled`	Boolean true or false	Optional for all types. Defaults to true. This member indicates whether the menu item is clickable on the client side. True equates to active, false equates to not active.
`enableFn`	String	Optional for all types. If present, this member is the name of a JavaScript function to execute on the client to determine whether this menu item is active. The complete menu feed JSON object for this menu item is passed as an argument to the function. The result from this function call overrides the "enabled" setting.
`actionUrl`	String	For a `StaticMenuitem`, either the `actionUrl` or the `actionFn` must be present. If `actionUrl` is present, it might be an absolute URL or a relative URL, or a query string that is appended to the current request URL or tag value. Not required for a `DynamicMenuitem`. Not useful for any other type. Invoking the `actionFn` takes precedence over an `actionUrl` if both are present.
`actionHttpMethod`	String, with normal HTTP values "GET", "POST", "PUT", "DELETE" and other values.	Defaults to "GET". Optional for all object types. If present on a `DynamicMenuitem`, this overrides any action that is provided by the Operation Framework plug-in.
`actionFn`	String	For a `StaticMenuitem`, either the `actionUrl` or the `actionFn` must be present. If `actionFn` is present, this member is the name of a JavaScript function to execute on the client when the menu item is clicked. The complete menu feed JSON object for this menu item is passed as an argument to the function. Invoking the `actionFn` takes precedence over an `actionUrl` if both are present.
`visibilityFn`	String	Optional for all types, but not useful for Separator or `ModuleRef`. For a `DynamicMenuitem` or `StaticMenuitem`, if `visibilityFn` is present, this member is the name of a JavaScript function to execute on the client to determine whether this menu item is active. The complete menu feed JSON object for this menu item is passed as an argument to the function.

Member name	Value and example syntax	Comments
`metadata`	For version 8.0.0.1: Embedded object, where the members can be String values, boolean values, or numeric values, or other nested objects. There is no enforced limit on the nesting level. There are only 2 restrictions on metadata: • No arrays can be used within metadata • No member within a metadata object can have the name "metadata". This results in an IllegalSyntaxException. Prior to version 8.0.0.1, metadata was restricted to only String types with no embedded nesting.	Optional for all types. If present, the members within must observe the restrictions that are listed for the appropriate version. If substitution variables are present in any String value within the metadata object, at any nesting level, in the form `${variableName}`, then these values are substituted from query parameters on the received menu request (`...&variableName=value...`). Multiple replacement variables can be contained in a metadata value string, but only one pass is made through the string. Any replacement variables that are found for which there are no substitutions found in the query parameters from the request, remain unchanged in the metadata that is passed to the client as part of the JSON feed response. For example: `[` `  {` `    ....` `    "metadata" : {` `  "a" : "something",` `  "b" : true,` `  "c" : 123,` `  "thisIsNestedMetadata" : {` `  "a" : "something nested",` `  "d" : "This is a substitution:  ${su` `  }` `    }` `    ...` `  }` `]`
`moduleId`	String	Optional for a Submenu, but only applicable for a Submenu. If present instead of an ID in a Submenu entry, this member is the name of a module like the `id` of a `ModuleRef` that is accessed by the client in a separate direct request if the Submenu item is expanded by the user. The format of the menu item that is built by this is `{"type":"Submenu", "id" : "moduleRef:", ... }`. The client code makes a request to the server by using this ID that is a variation of the normal menu request: `?uri=menu:moduleRef:`. The feed provider code handles this request by starting the named module as though there were a menu feed definition file that contained a `ModuleRef` with that ID.
`moduleArgs`	String, in the format of an HTTP request query parameter set, but without a leading ampersand (the menu feed provider prepends an ampersand). If there are multiple query parameters, insert an ampersand between each parameter after the first. Example: `foo=bar&foo2=bar2&...`	Optional for a Submenu, but only applicable for a Submenu. If present with a moduleId, then the URI built by the menu feed provider for the Submenu entry in the feed looks like `{"type" : "Submenu", "id" : "moduleRef:`*`moduleId_value`*`&`*`moduleArgs_value`*`", ... }`. If present with an ID in the Submenu, the URI looks like the previous example but without the `moduleRef:` prefix for the ID and arguments.

Member name	Value and example syntax	Comments
`markupId`	Arbitrary string, used to create an ID in the html tag that defines the menu item.	Optional. Useful only for `DynamicMenuItem` and `StaticMenuItem` menu entries. Variable substitution is supported for this item, so the `${windowID}` can be used to make unique instances of an ID for each portlet when rendered.

Writing a menu definition file for a modularized theme

You can write a new menu definition file for a new theme by using the server-side feed provider and client-side JavaScript. If the ready-to-use sample theme is copied and altered into a new customized theme, the same JSON menu files can be used, or can be altered as necessary. If any new client-side functions are referenced from menu items, then these new JavaScript features must be created and referenced.

The JSON menu feed provider looks for the JSON menu definition files in a `menuDefinitions` directory in the theme root in WebDAV.

Use the existing menu definitions files as samples, and use tools such as jsonlint to prevalidate the JSON syntax.

The JSON syntax is restrictive for menu definition files except that comments are allowed.

When you are debugging, use the trace string `com.ibm.wps.jsonmenu.*=all` on the server.

Dynamically extended and dynamically constructed menus

Starting with WebSphere Portal 8.0.0.1, it is possible to have menu items added dynamically to a menu definition file. This feature can be used to extend an existing "static" menu definition file, such as the files that exist in the sample theme within the `menuDefinitions` folder in WebDAV. It can also be used to dynamically construct a menu without a static menu file that exists at all.

Recall that in the request URI for a menu, the query parameter starts with `"?uri=menu:`*`menu name`*`"`. This menu name is treated as a file name by the menu framework, and we search for the file in the "base location" for resources for that theme as given by the theme metadata property `com.ibm.portal.theme.template.ref`. Within the menu definition, an entry of `"type":"ModuleRef"` can reference a theme optimization module, which contains a `menu` contribution, and within that a JSON subcontribution. This JSON subcontribution has a URL, which points to the actual JSON code. The JSON retrieved from that subcontribution URL is substituted into the menu feed in place of the `"type":"ModuleRef"` entry from the file.

Starting with WebSphere Portal 8.0.0.1, this JSON subcontribution can also be tagged with a `ref-id` qualifier. The value of this qualifier is a String. At the end of processing the menu definition file, the menu framework adds one more step. It searches through all the JSON subcontributions that are found in theme modules from the profile for the current page, where the page is given by the **`navID`** parameter on the menu request. If any of the `ref-id` tags from those subcontributions match the requested menu name that is being processed, then the

JSON code for those subcontributions is dynamically added to the end of the menu feed, exactly as though there were an entry `"type":"ModuleRef"` that specifically pointed to that module.

This menu framework allows for dynamic extension of the menu contents with new menu items by creating new theme modules, or at least new subcontributions, and updating the profile, rather than having to update the menu definitions files.

In addition, this feature can be used to dynamically construct a menu without creating a menu definition file at all. If the incoming menu request contains a menu name that does not exist, the menu framework now treats that as though it pointed at a menu definition file with only the minimum menu syntax for an empty array, `"[ ]"`. The menu is then created by using only the JSON subcontributions from the modules that are named in the profile for the current page where the `ref-id` qualifier matches the requested menu name, even though it names a file that does not exist.

Important: The order of the items that are dynamically added to a menu is not controlled or guaranteed in any way. Although in testing the order was consistent between server restarts on the same system, it was found to be inconsistent across different operating systems. Especially between IBM JVMs and those not from IBM. If it is necessary to strictly control the order of the menu items, then use the explicit `"type":"ModuleRef"` entry in a menu definition file to explicitly insert the menu item, rather than relying on the dynamic contribution mechanism.

For more information about constructing theme optimization modules that include the JSON subcontribution, see Adding a menu item with a module.

Controlling the cache lifetime of a JSON menu feed

In WebSphere Portal Express version 8.0.0.1 and later, the JSON menu framework accepts one or more WP ConfigService properties of the form `"jsonmenu.cache.time.<menu name>"` where the value is the cache time in seconds. This value must be greater than or equal to 0. A value of `0` indicates that the menu is not cached.

The `<menu name>` values from these WP ConfigService settings are matched against the value from the `"?uri=menu:<menu name>"` query parameter on the incoming request for a menu. The comparison of the menu names from the WP ConfigService to the received request for a menu is case-sensitive.

Related information:

JSON home page

Changing menu item visibility:

Whether the menu items are visible is controlled in a layered way by client-side JavaScript and markup. The default value for visibility, if not specified, is Boolean true.

Procedure

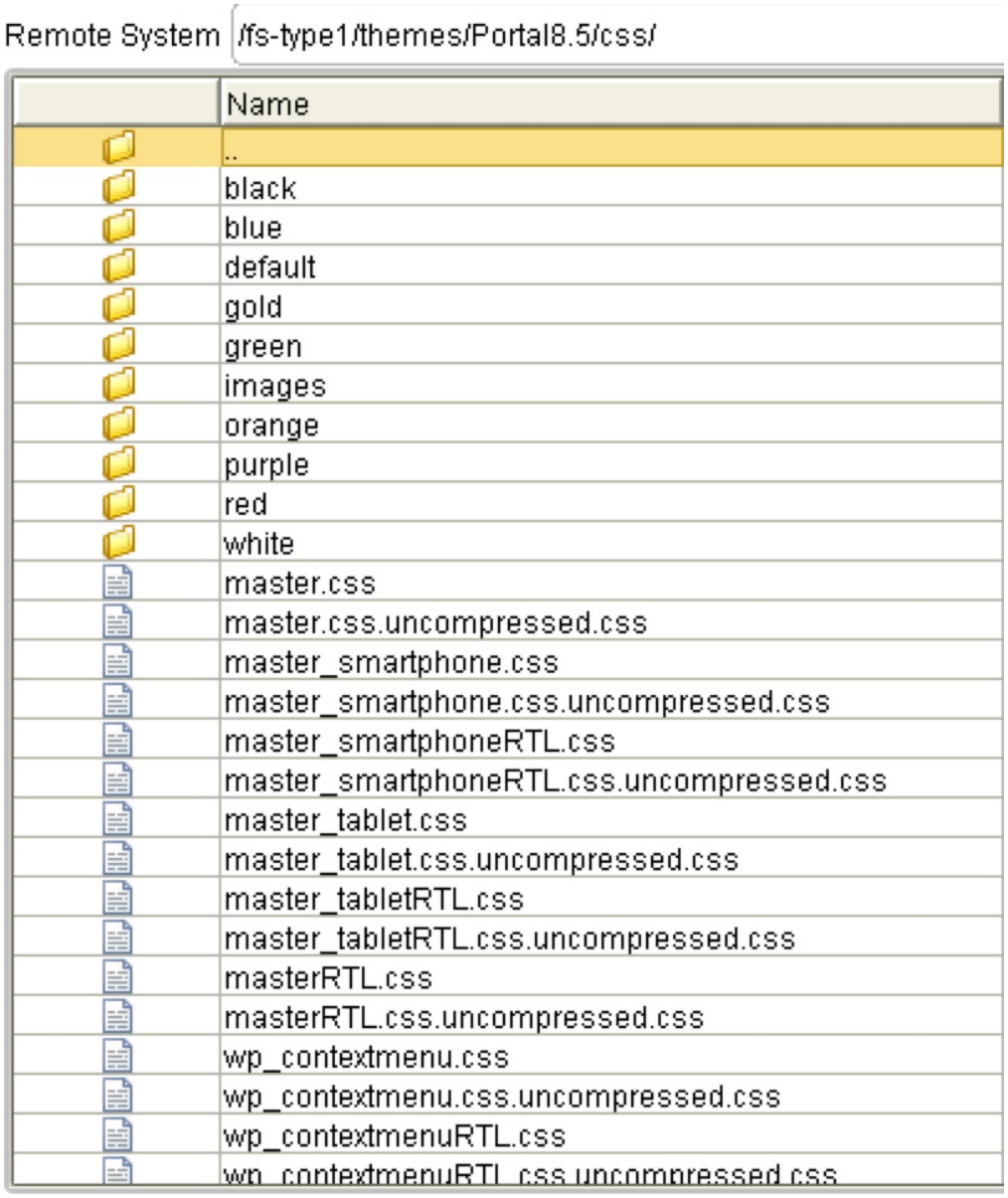
1. In the menu feed from the server is a default value whether a menu item is visible. The client-side code controls whether a menu item is visible in the feed. If a menu item includes a visibilityFn member, that function is invoked client-side to control the visibility.

2. Mark a menu item with a visibility boolean member, in the case where the server-side feed provider code has reached the conclusion that the menu item should not be visible to the client ("visibility" : false). The "visibility" : false member is added to the menu feed when a DynamicMenuitem references a plug-in whose isActive() method returns false when invoked by the menu feed provider code. The isActive() method on a plug-in can return false if, for example, the user does not currently have access control permissions sufficient to invoke the operation on the current page or portlet or other object, or if operation is inconsistent with the current state of the object (if known).

Menu styles:

You can vary the appearance of the items in menus that are delivered through the JSON menu framework.

The appearance of the items in the menus that are rendered by the JSON menu framework are determined by style names that open on the rendered markup for the menu items. These styles are defined in Cascading Style Sheet (CSS) files that are part of the theme. The default theme has these style sheets in WebDAV in the `themes/Portal8.5/css` directory.



There are two ways that the appearance of the menu items can be changed.

- Keep the same style names but change the definition of the named styles in the css files.
- Change the style names that are applied to the menu items and provide definitions and implementations of the new styles.

The first method keeps the same style names on the menu items, but redefine what appearance results from that style. When you creat your custom theme, take copies of the existing default theme css files as samples and alter them to achieve the appearance you want.

For the second approach, the **JSON** menu definition syntax allows the specification of a style name on the menu items, including headers and separators, using the itemClass member in a menu item definition. By adding explicit entries for the style names on the menu items, those overriding style names are used in the resulting menu markup. The itemClass member is only useful on menu items of

type StaticMenuItem, DynamicMenuItem, Header, Separator, and SubMenu. On a SubMenu, the style is applied only to the SubMenu entry `anchor link` itself, and not to the menu items that actually make up that SubMenu.

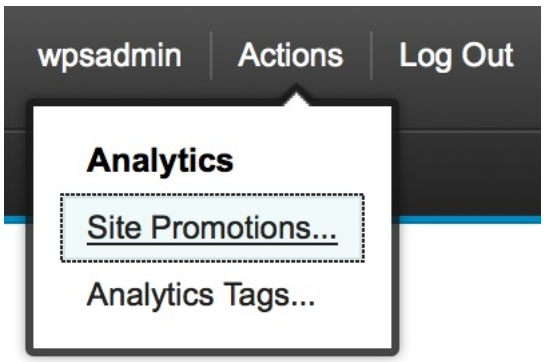
These CSS files define the menus.

`contextmenuCommon.css`
Contains common styles are valid for left-to-right (LTR) and right-to-left (RTL) structured pages.

`contextmenu.css`
The menu styles that are used on LTR structured pages.

`contextmenuRTL.css`
The menu styles that are used on RTL structured pages.

Default menu styles



Style class definitions

wpthemeMenuShow
By default all menus are hidden. This class is used to control visibility. If set on the node the menu is made visible.

wpthemeMenuLeft
Defines a menu that is left-oriented and opens to the right.

wpthemeMenuRight
Defines a menu that is right-oriented and opens to the left.

wpthemeMenuBorder
Defines the main `div` that shows the border and contains all menu items.

wpthemeMenuNotchBorder
Defines the arrow that points to the button that opened the menu.

wpthemeMenuDropDown
Inner element within the menu that contains a list of menu items.

wpthemeMenuitem
Describes a single menu item.

wpthemeMenuDisabled
Describes a disabled menu item.

wpthemeMenuLoading
Describes the text that is displayed when the data is being loaded.

wpthemeAnchorSubmenu
Used to register the submenu within the markup.

wpthemeTemplateMenu
Defines the template for an individual menu item.

wpthemeTemplateLoading
Defines the template for the loading text.

wpthemeTemplateSubmenu
Defines the template for the submenu.

wpthemeMenuError
Used for display when an error occurs.

The menu JavaScript is in `dav:fs-type1/themes/Portal8.5/js/contextmenu.js`.

The `wptheme.contextmenu.css` JSON object defines constants that are used to apply CSS classes. The constants are by default that is set to:

```
contextMenu: {
...
css: {
disabled: "wpthemeMenuDisabled",
show: "wpthemeMenuShow",
error: "wpthemeMenuError",
menuTemplate: "wpthemeTemplateMenu",
submenuTemplate: "wpthemeTemplateSubmenu",
loadingTemplate: "wpthemeTemplateLoading"
},
....
}
```

Adding a link to the resource permissions administration portlet in a menu

You can add a link to the resource permissions portlet to see or modify role types or inherited access.

About this task

To view or modify role types or inherited access, you must use the **Resource Permissions** portlet. Click the **Administration menu** icon. Then, click **Access** > **Resource Permissions**.

Procedure

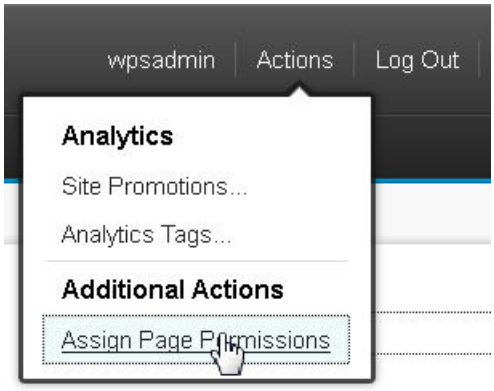
1. Connect your WebDAV client, to `http://`*`host`*`:`*`port`*`/wps/mycontenthandler/dav/fs-type1/`.
2. Go to `/themes/`*`YourCustomTheme`*`/menuDefinitions` and open the menu file to which you want to add the new permissions entry. By default, the `pageAction.json` file corresponds to the **Actions** menu.
3. Add this menu entry to the JSON array: If you make this the last entry, add a comma before the following example.

```
{
   "type":"DynamicMenuitem",
   "id":"ibm.portal.operations.assignPagePermissions",
   "titles": [
      {
         "value":"Assign Page Permissions",
         "lang":"en"
      }
   ]
}
```

4. Restart the server to apply your changes.

Results

The following screen shot shows the modified **Actions** menu. The link in the menu opens a resource page permissions portlet that affects the page that the user is currently viewing.



Theme templates (`theme.html`)

You can use static HTML to write portal themes. The static theme template is named `theme.html`.

A `theme.html` file is in the root directory of the theme on WebDAV `\fs-type1\themes\`*`theme-name`*`\`, and there are `theme.html` files in the `nls` directory under the theme `\fs-type1\themes\`*`theme-name`*`\nls\`. The `theme.html` includes the full HTML structure of the page, including `<html>`, `<head>`, and `<body>` sections. It can include both static and dynamic content.

Notes:

- The folder `nls` contains a file that is named `theme.html` without a locale that is associated with it. This file is not used. You can ignore it.
- Remember to modify the theme template files by using the WebDAV entry point `fs-type1`. When you use this entry point, your changes to the theme template are immediately reflected upon a browser refresh.

Root theme template

In a default WebSphere Portal Express installation, the portal does not render the template file `theme.html` in the root directory of the theme. Instead, this file links to the templates, and the portal renders the appropriate template. The links to the templates are in the `<head>` section of the root template. They have the following form:

```
<link rel="alternate" href="nls/theme_locale_code.html" hreflang="locale_code">
```

An example of a link to the English template file is as follows:

```
<link rel="alternate" href="nls/theme_en.html" hreflang="en">
```

If you do not want to use theme templates, you can remove these links from the `theme.html` template in the root directory. If you do this, the portal renders this root template.

This theme template also includes Apache Ant scripting in the following form:

```
${bundle_name:bundle_key:character_encoding}
```

The `character encoding` replaces special characters with the escape sequence determined by the specified encoding. The available types of encoding are `xml` or `json`. You can chain multiple instances of encoding as follows:

`${bundle:key:json:xml} or ${bundle:key:xml:json}`

You can use the Apache Ant build framework to generate templates that are based on this root template. This can be useful if you want to update one template during development and then generate the templates by using the Ant build process. If you want to use only the root template, replace the Ant scripting with the preferred text that you want to be rendered. You can learn more about the Ant build tool at Apache Ant.

Theme templates

In a default WebSphere Portal Express installation, the theme architecture renders content by using the theme templates. These templates are in the `nls` subdirectory under the theme directory on WebDAV. These files have the locale code that is appended to the end of the template name, for example `theme_en.html` for English. These templates translate static text inline within the template.

When you use the theme templates and want to view your changes, update the template that the portal renders in the browser. For example, if your preferred language is English, update the file `theme_en.html`.

Adding static content to the `theme.html`

You can add static content to the `theme.html` in the following ways:

Adding content directly:
You can add static content, such as HTML markup and images directly to the file `theme.html`.

Adding content from WebDAV:
You can add content that is in WebDAV relative to the `theme.html` file with a relative URL reference.

Adding content by relative URLs:
You can use relative URLs to reference static content in the `/common-resources/` folder in the WebDAV file store. If the relative path does not successfully resolve to a file within the theme folder, the portal uses the folder `/common-resources/` as a fallback location to locate the resource. This way the theme can reference common resources and still preserve the ability to override a file in that folder with a resource of the same name in the theme folder.

Adding server-side dynamic content to the `theme.html`

Dynamic content changes per user, or per page, or per some other server state. Therefore, you cannot define it statically in the theme file. Instead, you insert it into the response at run time. To do this, you edit the `theme.html` and identify these dynamic content spots. Then at run time, a server-side theme parser identifies and resolves dynamic content spots, and streams their output into the final response to the browser.

The format of a dynamic content spot is as follows:

```
<a rel="dynamic-content" href="{path to dynamiccontent}"></a>
```

rel="dynamic-content"
The theme template parser recognizes the element `rel="dynamic-content"`. It resolves the `href` attribute and inserts its output into the response.

href The `href` can point to any URI that is resolved by the resource addressability framework.

Examples:

1. The following example is a special content spot that renders the referenced layout template and content of the current page:
   ```
   <a rel="dynamic-content" href="lm:template"></a>
   ```
2. The following example includes a dynamic content spot from the mapping that is specified through the module `dyn-cs` contribution type, in the `WP_DynamicContentSpotMappings` Resource Environment Provider or theme metadata.
   ```
   <a rel="dynamic-content" href="dyn-cs:customSpot"></a>
   ```
3. The following example includes the output of a theme JSP with a resolver POC URL:
   ```
   <a rel="dynamic-content" href="res:/customContext/themes/html/customTheme/customSpot.jsp"></a>
   ```

Changing the theme template location

You can change the location of the theme template. For this purpose, the theme contains a metadata parameter that stores the location of the theme template `theme.html`. The parameter name is `com.ibm.portal.theme.template.ref`. If required, you can point it to an external location. For example, you can specify a POC URI or the URL to an external server. You do not need to store the theme template on WebDAV. In a default portal installation, the metadata parameter for the theme is as follows:

```
<parameter name="com.ibm.portal.theme.template.ref" type="string" update="set">
<![CDATA[dav:fs-type1/themes/theme_folder/]]>
</parameter>
```

"Setting Inherited Theme Templates"
You can set a theme template to be used on a page and it automatically sets the template for all child pages that are associated with that page. The inherited metadata can be used when you want every page under a specific page to have the same theme template.

"Renaming Theme Templates" on page 2709
You can rename the default filename of the theme template from `theme.html`. In the ready-to-use theme renaming the `theme.html` file is used to provide a different theme view for the administrative pages.

Setting Inherited Theme Templates:

You can set a theme template to be used on a page and it automatically sets the template for all child pages that are associated with that page. The inherited metadata can be used when you want every page under a specific page to have the same theme template.

About this task

This inheritance is introduced through a new metadata key which is added to a page. The new metadata is
`com.ibm.portal.theme.inherited.template.file.name.html=`*`myfile`*`.html`.

Note: The *myfile*.html file, which corresponds to the theme template that you are setting, must already be in WebDAV in the directory /mycontenthandler/fs-type1/themes/Portal8.5/.

You can set the theme template by using WebDAV or managed pages. After your custom theme template is set, you can see it displayed on the page and all child pages corresponding to that parent page.

Procedure

1. Set the theme template through the **Edit Page Properties** dialog from the **Overview** tab.
   a. Go to the page where you would like to set the new metadata.
   b. Click **Edit Mode**.
   c. In the **Overview** section, hover over the name of the page and click the **Edit**.
   d. Click the **Advanced** tab and enter the following information in the **Metadata** section:

      Key com.ibm.portal.theme.inherited.template.file.name.html.

      Value Enter the HTML file name that corresponds to the theme template you are trying to set.
   e. Click **Add** to set the new metadata for the page.
   f. Click **Save**.
2. Set the theme template from the managed pages administrative portlet.
   a. In the IBM WebSphere Portal Express administrative section, click the **Manage Pages** tab and go to the page where you want to set the metadata.
   b. Select the **Edit Page Properties** option for the page you want to edit.
   c. In **Advanced options**, select **I want to set parameters** and enter the following information:

      New parameter
      com.ibm.portal.theme.inherited.template.file.name.html.

      New value
      Enter the HTML file name that corresponds to the theme template you are trying to set.
   d. Click **Add** to set the new metadata for the page.
   e. Click **Save**.

Renaming Theme Templates:

You can rename the default filename of the theme template from theme.html. In the ready-to-use theme renaming the theme.html file is used to provide a different theme view for the administrative pages.

About this task

This change can be done selectively for individual pages by setting a page metadata called com.ibm.portal.theme.template.file.name.html. This setting is not inherited along the navigation hierarchy.

To set the theme template on a page, you must use the XML configuration interface (XMLAccess).

Procedure

1. Export the page by using XMLAccess or by following the instructions in Exporting pages and labels.
2. Edit the resulting XML file and add the following parameter to the content-node, replacing `my_theme.html` with your theme template file name.

   ```
   <parameter name="com.ibm.portal.theme.template.file.name.html" type="string" update="set"><![CDAT
   ```
3. Save the XML file.
4. Import the page using XMLAccess or the Import XML portlet.

Changing the theme logo

You can change the theme logo to customize your portal site and rebrand it to reflect your business.

About this task

The default logo that appears in the banner is a blank placeholder image. The following block of code displays the blank placeholder image:

```
<span class="wpthemeBranding">
     <img alt="IBM Logo" src="/wps/themeModules/themes/html/dynamicSpots/icons/blank.gif">
     <span class="wpthemeAltText">IBM Logo</span>
</span>
```

This logo is provided to the theme with a CSS style class so it can be modified or overridden by packaged styles on the customization shelf. It also updates as new styles are selected. You do not have to modify the theme template.

You can change the theme logo in `.wpthemeLogo in fs-type1:themes\ Your_custom_theme \css\default\default_view.css and fs-type1:themes\ Your_custom_theme \css\default\default_view.css.uncompressed.css`:

Procedure

1. Remove the **display** attribute from the style class.
2. Define a new path to your image for the **background-image** attribute.
3. Modify the height and width.

You can also change your header text to display your logo next to the navigation.

4. Apply the **white** style on the customization shelf.
5. In `fs-type1:themes\Your_custom_theme\nls\theme_locale.html`, enter the following block of code, where *IBM WebSphere Portal* is the name of your portal site:

   ```
   <div class="wpthemeLogo wpthemeLeft">
        <span class="wpthemeAltText">IBM WebSphere Portal</span>
   </div>
   ```

   Note: There is no HTML image element to display the logo. It is provided to the theme through a CSS style class just like the first area of branding. You do not have to modify the theme template.

What to do next

You can also create your own custom style to use with your custom theme defined in `fs-type1:themes\Your_custom_theme\system\styles.json`. If you do that, you can override these styles in the custom style css file, such as `fs-type1:themes\ Your_custom_theme \css\Your_custom_style \Your_custom_style .css`.

"Changing the logo action"
You can change the default action of a logo to take users to different pages in your portal.

Changing the logo action:

You can change the default action of a logo to take users to different pages in your portal.

The default logo action takes users to your portal home page. The logo can complete three other actions:

- The logo can take users to the first child page.
- The logo can take users to another portal page with the unique ID of that page.
- The logo can take users to another portal page with the vanity URL of that page.

Taking users to the first child page

Remove the default code and replace it with the following code :

```
<%-- Logo --%>
 <li>
  <span class="wpthemeBranding">
  <portal-core:lazy-set var="showHiddenPages" elExpression=="wp.publicRenderParam['{http:/
  <portal-logic:if deviceClass="smartphone/tablet">
   <c:set var="isMobile" value="true"/>
  </portal-logic:if>
     <c:set var="homeNodeFound" value="false"/>
  <c:forEach var="node" items="${wp.navigationModel.children[selectionPath[1]]}" varStatus
         <c:set var="isHiddenPage" value="${node.metadata['com.ibm.portal.Hidden'] || (is
         <c:if test="${!homeNodeFound && (!isHiddenPage || showHiddenPages)}">
   <c:set var="nodeID" value="${wp.identification[node]}"/>
           <a class="wpthemeBrandingLink" href="?uri=nm:oid:${nodeID}" alt="<portal-fmt:o
   <img src="
      </a>
      <span class="wpthemeAltText"><portal-fmt:text key="theme.ibmLogo" bundle="nls.commor
         <c:set var="homeNodeFound" value="true"/>
    </c:if>
  </c:forEach>
  </span>
 </li>
```

Taking users to another portal page with its unique ID

Remove the following line of default code:

```
<a class="wpthemeBrandingLink" href="?uri=nm:oid:${nodeID}" alt="<portal-fmt:out>${node.ti
```

Replace it with the following code snippet, where the value of *contentNode* is replaced with the unique name of the page:

```
<portal-navigation:urlGeneration contentNode="ibm.portal.Home.Welcome" >
           <a class="wpthemeBrandingLink" href="<% wpsURL.write(escapeXmlWriter); %>" alt
   </portal-navigation:urlGeneration>
```

Taking users to another portal page with its vanity URL

Remove the following line of default code:

```
<a class="wpthemeBrandingLink" href="?uri=nm:oid:${nodeID}" alt="<portal-fmt:out>${node.ti
```

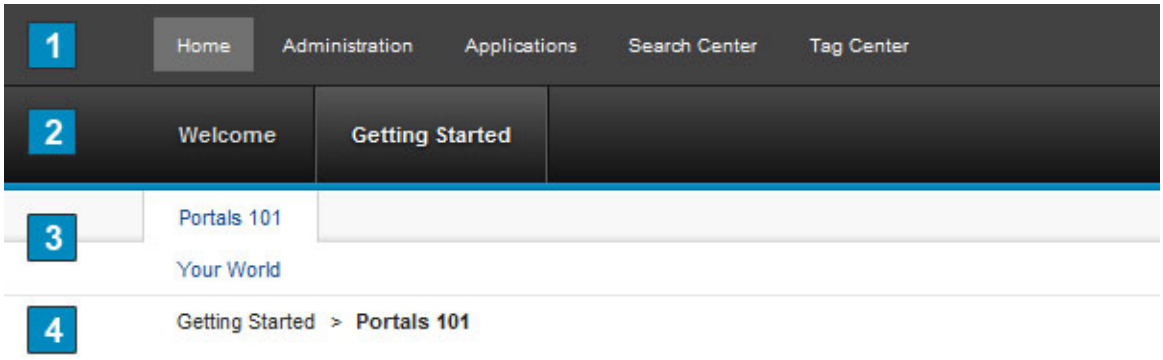
Replace it with the following code snippet, where *host*, *port*, and *contextroot* are substituted with your actual values and where *Home* is replaced with the vanity URL of your page:

```
<a class="wpthemeBrandingLink" href="http://host:port/contextroot/vanityurl/Home" alt="">
```

Customizing navigation

Use dynamic content spots to determine what is displayed by Top, Primary, and Secondary navigation. Use the `navigation.jsp` file to map properties to the dynamic content spot IDs in the `theme.html` files. Rendering of the navigation is done with a single JSP file with `<ul>` and `<li>` tags.

These are the levels of navigation that is provided in a theme:



Top - Item 1

Displays links for the pages directly under Content Root, such as Home, Administration, and Applications. Use the `dyn-cs:id:85theme_topNav` to display Top navigation:

```
<div class="wpthemeHeader">
   ...
   <a rel="dynamic-content" href="dyn-cs:id:85theme_topNav"></a>
   ...
</div>
```

Primary - Item 2

Displays links to the child pages of the currently selected top page, such as Getting Started and Features for Home. Use the `dyn-cs:id:85theme_primaryNav` to display Primary navigation:

```
<div class="wpthemeBanner">
   ...
   <a rel="dynamic-content" href="dyn-cs:id:85theme_primaryNav"></a>
   ...
</div>
```

Secondary - Item 3

Displays links to the child pages of the currently selected primary page. Use the `dyn-cs:id:85theme_secondaryNav` to display Primary navigation:

```
<div class="wpthemeSecondaryBanner">
   ...
   <a rel="dynamic-content" href="dyn-cs:id:85theme_secondaryNav"></a>
   ...
</div>
```

Breadcrumb - Item 4

Displays the position of the current web page within the website and the logical path back to the highest level of the site framework. The breadcrumb trail starts at the content root and goes down to the currently selected static page.

Side Displays links for the child and grandchild pages of the currently selected top page. By default, this template is applied to the Administration section of your portal.

```
<div class="wpthemeSideNavigation wpthemeLeft" role="navigation">
   ...
   <a rel="dynamic-content" href="dyn-cs:id:85theme_sideNav"></a>
   ...
</div>
```

Mobile

Displays links for the child and grandchild pages of the currently selected top page, on mobile devices only, such as smartphones and tablets. CF06 Each level of navigation is lazily loaded onto the page as the user clicks.

```
<div class="wpthemeBanner">
      ...
      <a rel="dynamic-content" href="dyn-cs:id:85theme_mobileNav"></a>
      ...
</div>
```

CF06 To turn off lazy loading of mobile navigation, replace the 85theme_mobileNav dynamic content spot with the 85theme_mobileNav_static dynamic content spot. CF06

```
<div class="wpthemeBanner">      ...
          <a rel="dynamic-content"
    href="dyn-cs:id:85theme_mobileNav_static"></a>            ...
    </div>
```

Navigation content spots

The content spot IDs map to **wp_dynamicContentSpots_85** module subcontributions in the *PortalServer_root*\theme\wp.theme.themes\default85\installedApps\DefaultTheme85.ear\DefaultTheme85.war\WEB-INF\plugin.xml file.

Table 445. Content spot property names.

Name	Value
85theme_topNav	mvc:smartphone/tablet@res:{war:context-root}/themes/html/dynamicSpots/navigation.jsp?type=top
85theme_primaryNav	mvc:res:{war:context-root}/themes/html/dynamicSpots/navigation.jsp?type=primary,smartphone@,tablet@
85theme_secondaryNav	mvc:res:{war:context-root}/themes/html/dynamicSpots/navigation.jsp?type=secondary,smartphone@,tablet@
85theme_sideNav	mvc:res:{war:context-root}/themes/html/dynamicSpots/sideNavigation.jsp?startLevel=2,smartphone@,tablet@
85theme_mobileNav	mvc:smartphone/tablet@res:{war:context-root}/themes/htm CF06 mvc:smartphone/tablet@res:{war:context-root}/themes

The subcontribution URI value indicates which JSP is loaded in the spot. For the three horizontal navigation spots, the same navigation.jsp file is used with a different parameter passed to the JSP. The navigation.jsp file is in the *PortalServer_root*\theme\wp.theme.themes\default85\installedApps\DefaultTheme85.ear\DefaultTheme85.war\themes\html\dynamicSpots folder. The parameter is a key of type with three possible values:

top Causes the navigation JSP to render the top navigation, starting at level 0 in the page navigation, which is the highest level.

primary
Causes the navigation JSP to render the primary navigation, starting at level 1 in the page navigation.

secondary
Causes the navigation JSP to render the secondary navigation, starting at level 2 in the page navigation.

In the `theme.html` files for your theme, you can remove the default navigation dynamic content spots. You can then replace the dynamic content spots with your own mappings that point to your own jsp implementation. For example, you can replace the three top navigation levels with a single top navigation and a single side navigation.

"Creating a dynamic content spot for navigation"
Create a dynamic content spot mapping to customize the theme for Top, Primary, and Secondary navigation. Change the *yourTheme* value to the name of your theme.

"Adding a level of navigation" on page 2715
You can increase levels of navigation by adding a fourth level of navigation.

"Removing a level of navigation" on page 2715
You can reduce your levels of navigation to two levels.

"Side navigation" on page 2715
The Portal 8.5 theme includes a side navigation template that can be applied to render pages at the secondary level in a list with the main content. By default, this template is applied to the Administration section of your portal.

Creating a dynamic content spot for navigation:

Create a dynamic content spot mapping to customize the theme for Top, Primary, and Secondary navigation. Change the *yourTheme* value to the name of your theme.

Procedure

1. Optional: For primary navigation, add a subcontribution to the **wp_dynamicContentSpots_85** module in your theme's `plugin.xml` file. The `plugin.xml` file is in the *wp_profile_root*`\installedApps\`*`cell`*`\`*`YourTheme`*`.ear\`*`YourTheme`*`.war\WEB-INF` folder.
   a. Give your subcontribution the `ref-id` *`yourTheme`*`_primaryNav`, assuming you are changing the primary level.
   b. Enter the URI value `mvc:res:{war:context-root}/themes/html/dynamicSpots/navigation.jsp?type=primary,smartphone@,tablet@`. For example,

```
<sub-contribution type="markup" ref-id="yourTheme_primaryNav">
<uri value="mvc:res:{war:context-root}/themes/html/dynamicSpots/navigation.jsp?type=primary,sm
</sub-contribution>
```

2. Modify the `theme.html` files for your theme and change the dynamic content spot ID for the primary navigation from `85theme_primaryNav` to *`yourTheme`*`_primaryNav`.
3. Define the *`yourTheme`*`PrimaryNav` style class in one of the `.css` files for your theme that gets loaded by one of the modules for your theme.
4. If you are in development mode, restart the application for your theme. Otherwise, restart the portal server.

Adding a level of navigation:

You can increase levels of navigation by adding a fourth level of navigation.

About this task

Note: Change the *yourTheme* value to the actual name of your theme.

Procedure

1. Add a subcontribution to the **wp_dynamicContentSpots_85** module in your theme's `plugin.xml` file. The `plugin.xml` file is in the *wp_profile_root*`\installedApps\`*cell*`\`*YourTheme*`.ear\`*YourTheme*`.war\WEB-INF` folder.
   a. Give your subcontribution the `ref-id` *yourTheme*`_tertiaryNav`.
   b. Enter the URI value `mvc:res:{war:context-root}/themes/html/dynamicSpots/navigation.jsp?type=tertiary,smartphone@,tablet@`.

   ```
   <sub-contribution type="markup" ref-id="yourTheme_tertiaryNav">
   <uri value="mvc:res:{war:context-root}/themes/html/dynamicSpots/navigation.jsp?type=tertiar
   </sub-contribution>
   ```
2. Modify the `theme.html` files for your theme and add a fourth, or tertiary, navigation dynamic content spot under the third, or secondary, navigation dynamic content spot, which is under the secondary banner block:

   ```
   <a rel="dynamic-content" href="dyn-cs:id:yourTheme_tertiaryNav"></a>
   ```
3. Define the *yourTheme*`TertiaryNav` style class in one of the CSS files for your theme that gets loaded by one of the modules for your theme.
4. If you are in development mode, restart the web application for your theme. Otherwise, restart the portal server.

Removing a level of navigation:

You can reduce your levels of navigation to two levels.

Procedure

Modify the `theme.html` files for your theme and delete the tertiary, or secondary, dynamic content spot:

```
<a rel="dynamic-content" href="dyn-cs:id:85theme_secondaryNav"></a>
```

Side navigation:

The Portal 8.5 theme includes a side navigation template that can be applied to render pages at the secondary level in a list with the main content. By default, this template is applied to the Administration section of your portal.

The side navigation is composed of three parts:

Theme template
The theme template includes the markup required to render a dynamic content spot next to the main content and sets a class on the body element that can be used to scope styles to this template.

Location of localized templates: `dav:fs-type1\themes\Portal8.5\nls\sidenav\theme_sidenav.html`

Dynamic Content Spot
The dynamic content spot recursively loops over the navigation hierarchy to render the pages. The dynamic content spot is a JSP that uses expression

logic to access the navigation and provides easy to read code. You can customize the start level parameter of the dynamic content spot to render different sets of pages.

Location: *PortalServer_root*\theme\wp.theme.themes\default85\installedApps\DefaultTheme85.ear\DefaultTheme85.war\themes\html\dynamicSpots\sideNavigation.jsp

Styles The CSS is built into the theme master layer and out of box customizations were not included for each style. The default style is applied to the Administration pages.

Location of uncompressed files:

dav:fs-type1\themes\Portal8.5\css\default\sidenav.css

dav:fs-type1\themes\Portal8.5\css\default\sidenavRTL.css

Location of compressed files:

dav:fs-type1\themes\Portal8.5\css\master.css

dav:fs-type1\themes\Portal8.5\css\masterRTL.css

"Set side navigation for a page"
You can set the side navigation template for a specific page.

"Set side navigation as theme default"
You can set the side navigation template as the theme default by changing the ready-to-use template to become the default theme.html.

"Customizing side navigation" on page 2717
You can customize your side navigation by scoping styles specifically to the side navigation template or changing the width of the side navigation area or main content area.

Set side navigation for a page:

You can set the side navigation template for a specific page.

Procedure

1. Log on to your portal.
2. Export the page where you want to apply the side navigation template.

   Note: You can export the page using XMLAccess or by using the Manage Pages portlet.
3. Add the page metadata key com.ibm.portal.theme.template.file.name.html with a value <![CDATA[theme_sidenav.html]]> to the exported XML file. For example:

```
<content-node action="update" content-parentref="wp.example.parent" active="true" objectid="wp.example.id" uniquename="wp.example">
  ...
  <parameter name="com.ibm.portal.theme.template.file.name.html" type="string" update="set"><![CDATA[theme_sidenav.html]]></parameter>
  ...
</content-node>
```

4. Save the new page XML file.
5. Import the XML file using the command line or the Import XML portlet.
6. You can now navigate to the page and it is now rendered as a side navigation.

Set side navigation as theme default:

You can set the side navigation template as the theme default by changing the ready-to-use template to become the default theme.html.

Procedure

1. Connect to the file store WebDAV entry point: http://<server>:<port>/wps/mycontenthandler/dav/fs-type1/
2. Navigate to the theme folder with the current default theme template, for example `dav:fs-type1/themes/Portal8.5/`.
3. You can rename the current `theme.html` to something else or you can delete it.
4. Navigate to the theme folder with the side navigation theme template, most likely the same folder as the old default `theme.html` file.
5. Rename the side navigation template from `theme_sidenav.html` to `theme.html`.

Results

After renaming the template to theme.html, it will now be used as the default template. Inside the `theme.html` file are links to the localized templates, which are named `theme_sidenav_`*`locale`*`.html`. There is no need to rename these templates, but if you want to make the template naming uniform, you can change the link elements as well as the localized template file names.

Customizing side navigation:

You can customize your side navigation by scoping styles specifically to the side navigation template or changing the width of the side navigation area or main content area.

Scoping styles to the side navigation template

You can scope styles specifically to the side navigation template so you do not need to include different styles between templates. By default, the template applies the CSS class wpthemeSplitView to the HTML body element. This class can be used in the CSS to scope styles and is demonstrated in the `sideNav.css`.

Example:

```
.wpthemeSplitView .someStyle { color: #000; }
```

This style demonstrates that it applies only to the element with someStyle class when the side navigation template is applied.

Change the width of the navigation and main content

You can change the width of the side navigation area or main content area by modifying the styles. The applicable styles are:

```
.wpthemeSplitView .wpthemeFrame {
min-width: 1225px;
}

.wpthemeSplitView .wpthemeMainContent > div {
width: 1135px;
}

.wpthemeSplitView .wpthemeLayoutContainers {
width: 850px;
}

.wpthemeSideNavigation {
width: 275px;
margin: 10px 10px 10px 0;
}
```

Change the layout container width

The ready-to-use layout templates have specific side navigation styles applied. If you change the width of the main content, you need to adapt the layouts. These styles are also in the `sideNav.css` file. The applicable styles are:

```
.wpthemeSplitView .wptheme1Col .wpthemeCol {width:850px;}
.wpthemeSplitView .wptheme2Col .wpthemeCol {width:400px;}
.wpthemeSplitView .wptheme3Col .wpthemeCol {width:260px;}
.wpthemeSplitView .wptheme2Col.wpthemeUnequal .wpthemePrimaryContainer {width:545px;}
.wpthemeSplitView .wptheme2Col.wpthemeUnequal .wpthemeSecondaryContainer {width:260px;}
.wpthemeSplitView .wptheme3Col.wpthemeUnequal .wpthemePrimaryContainer {width:434px;}
.wpthemeSplitView .wptheme3Col.wpthemeUnequal .wpthemeSecondaryContainer,
.wpthemeSplitView .wptheme3Col.wpthemeUnequal .wpthemeTertiaryContainer {width:175px;}
.wpthemeSplitView .wpthemeTopCol .wpthemeHeadlineContainer {width: 830px; margin-right: 15px; margin-bottom: 20px;}
.wpthemeSplitView .wpthemeRow {margin: 0 0 20px 20px;}
.wpthemeSplitView #layoutContainers .layoutRow .layoutColumn .component-control { width: 850px; }
```

Styles

The WebSphere Portal Express Version 8.5 provides a selection of ready-use styles that you can be applied to your portal pages. You can also modify the existing theme styles or add your own custom style to achieve the look that you want.

"Creating a theme style"
You can add your own custom style to the theme that can be selected on the site toolbar.

"Updating your custom style" on page 2719
After you create a theme style, use this procedure to apply a customized look and feel to the theme.

"Applying alternate styles" on page 2722
You can apply alternate styles to your portal pages using the IBM WebSphere Portal Express theme that you can apply to your portal pages.

Creating a theme style:

You can add your own custom style to the theme that can be selected on the site toolbar.

About this task

To add your own custom style to the theme, create a Cascading Style Sheet (CSS) file in WebDAV. This new file includes CSS class definitions that override the ones that are provided by the default CSS layer, which is located here: `dav:fs-type1/themes/Portal8.5/css/master.css`.

The ready-use alternate styles can be used as guide while creating your own alternate files in folders as peers to the default CSS layer. The alternate styles can be found at this location: `dav:fs-type1/themes/Portal8.5/css/`.

These steps add a new style to the Styles tab of the site toolbar.

Procedure

1. On your local system, create a folder and place a new CSS file in it that includes the class overrides for the alternate style. For example, `./custom/custom.css`.
2. Connect to the `fs-type1` WebDAV entry point, http://*server:port*/wps/mycontenthandler/dav/fs-type1/.
3. Copy the new folder to this location in WebDAV: `dav:fs-type1/themes/Portal8.5/css/`.

4. Open the file in WebDAV at `dav:fs-type1/themes/Portal8.5/system/styles.json`.
5. Register the style by adding an entry to the items array in the following format:

```
      {'label':'display_name_for_the_style',
   'id':'unique_name_for_style',
'url':'path_to_the_stylesheet_relative_to_theme_folder_in_webdav',
'thumbnail':'path_to_the_thumbnail_image',
'help':'short_description_for_the_style'}
```

   The following code is an example of the items array:

```
      {'label':'change_style_white',
   'id':'white.css',
'url':'css/white/white.css',
'thumbnail':ibmCfg.themeConfig.themeRootURI+'/css/white/icon.gif',
'help':'The white style.'}
```

   The display name shows in the site toolbar. Make the path relative to the folder in WebDAV. The JSON object in the file `styles.json` contains two attributes that are used for globalizing the style display strings `localizationPackageName` and `localizationBundleName`. These objects are used by the `dojo.i18n` file to provide localized strings by creating bundles with `dojo.i18n.getLocalization("localizationPackageName", "localizationBundleName")`. If you choose to globalize the display name of your new style, add a key to the bundle and replace the label value in the JSON with the key name. The site toolbar automatically looks up the display name in the bundle using the key.

Updating your custom style:

After you create a theme style, use this procedure to apply a customized look and feel to the theme.

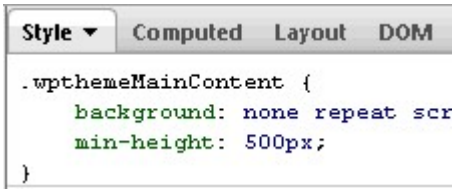
About this task

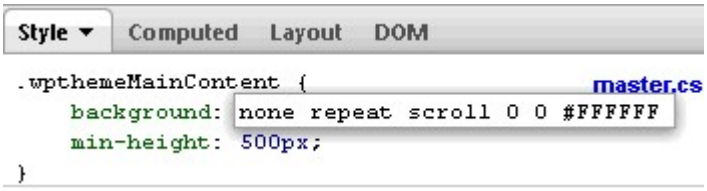
Do not edit the IBM WebSphere Portal Express 8.5 theme CSS style sheets directly, because these changes might be lost during a fix pack upgrade. Instead, create the CSS class in a new style sheet belonging to your custom theme.

Procedure

1. Apply the custom style to a page.
   a. On the page that you want to apply the style, turn on **Edit Mode**.
   b. Click the **Page** > **Styles** on the toolbar.
   c. Click the custom style.
2. Write the new styles.
   a. Open a portal page that has your theme and custom style applied.
   b. Use a tool, like Firebug, to select and inspect the style rules that you want to change.
   c. After determining the styles that you want to override in the theme, copy the overrides to the custom style sheet on WebDAV.
3. Apply the custom style sheet to your entire page structure.
   a. Figure out the root page for the area on which you want to apply the custom style. To apply to the entire site, select the **Content Root**.
   b. Add metadata on the root page with a key of `colorPalette` and a value pointing to your custom style sheet in relation to the theme directory on WebDAV. For example: `css/custom/custom.css`.

4. Using Firebug to inspect the CSS file.
   a. Open the Firebug plug-in in a Mozilla Firefox browser.
   b. Click the element selection icon  in the Firebug menu bar.
   c. Click the area on the page that you want to change.
   d. Check that the correct area is selected in the HTML tab. 

   e. Look at the **Style** tab to inspect all the CSS files that are applied to the HTML element. 

   f. Edit the CSS file in the **Style** tab until the HTML element looks how you want it to. To edit an attribute click the text.
   

   g. After you determine that the new value applied has the appearance that you want, copy the CSS file that you changed in the **Style** tab into the custom style sheet.

"Style definitions"
Customize your theme with these definitions.

Related information:

Get Firebug

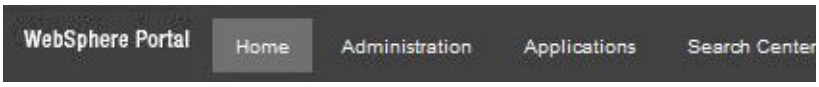
Style definitions:

Customize your theme with these definitions.

.wpthemeHeader
The container element for the theme header that by default displays the top-level navigation and page mode toggle button.

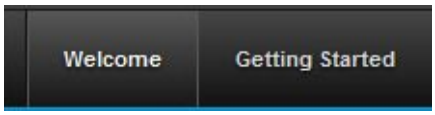
.wpthemeHeaderNav *
This style definition controls the look and feel for the top-level navigation located in the theme header.

.wpthemeLogo
This style definition controls the logo, next to the top-level navigation. By default this style is disabled.

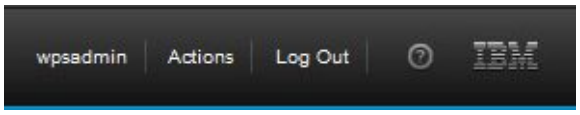


.wpthemeBanner
The container element for the theme banner that by default displays the primary navigation and common actions.

.wpthemePrimaryNav *
The style controls the look and feel for the primary level of navigation located in the theme banner.



.wpthemeCommonActions *
The style controls the look and feel for the common actions that by default displays the profile management link, actions menu, and login and logout links.



.wpthemeBanner.wpthemeBranding *
The logo displayed by default in the common actions.

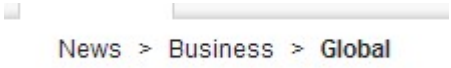
.wpthemeSecondaryBanner
The container element for the secondary banner that, by default,displays the secondary navigation and search form.

.wpthemeSecondaryNav *
The style controls the look and feel for the secondary level of navigation located in the theme banner.

.wpthemeSearch *
The style controls the look and feel for the search form.

. wpthemeMainContent
The container element for the main content of the page. By default, it displays the navigation breadcrumb trail, status bar, and layout.

.wpthemeCrumbTrail *
The style definitions with this scope control the navigation breadcrumb trail displayed before the main content.



.wpthemeFooter
The container element for the footer of the theme.

Applying alternate styles:

You can apply alternate styles to your portal pages using the IBM WebSphere Portal Express theme that you can apply to your portal pages.

Procedure

1. Log on to your portal as an administrator and select the page that you want to change the style.
2. Turn on **Edit mode**.
3. Click **Page** > **Styles** on the toolbar.
4. Select a style for your portal page. When you click a style, that style is applied to the page.

Configuring the portal theme and modules

Themes and modules are configured through theme metadata properties and resource environment provider custom properties.

"Changing theme metadata"
The first step of theme configuration is through theme metadata properties. Changes to the metadata are specific to a single theme, and the entries and values, therefore, can vary from theme to theme.

"Changing resource environment provider custom properties" on page 2723
The second order of theme configuration is through resource environment provider (REP) custom properties in the **WP GlobalThemeConfig** REP. Changes to the REP custom properties apply across all themes, and the values, therefore, cannot vary from theme to theme.

"Adding resource environment provider properties" on page 2724
Resource environment provider (REP) custom properties are global in scope, but you can use a consistent naming convention for a theme or module that applies across multiple themes.

CF06 "Configuration for resource aggregation" on page 2725
Set the following properties in the WP ConfigService Resource Environment Provider to configure the resource aggregator.

"Using your configuration properties" on page 2725
Create a JavaScript object for your theme on the client side. Theme metadata properties must be loaded dynamically. If you must load a property statically, use a resource environment provider custom property instead.

"Configuration settings for capability filters" on page 2727
Set the following properties in the **Wp ConfigService** Resource Environment Provider to enable or disable the various capability filters.

Changing theme metadata:

The first step of theme configuration is through theme metadata properties. Changes to the metadata are specific to a single theme, and the entries and values, therefore, can vary from theme to theme.

About this task

The theme metadata is located in the folder for your theme in `themelist`.

Procedure

1. Connect your WebDAV client to http://*localhost*:*port*/wps/mycontenthandler/dav/themelist/

2. From the folder for your theme, copy the `metadata.properties` file to local drive.
3. Edit the local copy of the file and modify the property values as needed according to the following table.
4. Copy the local copy of the `metadata.properties` file back into the folder for your theme in `themelist`.

Results

Table 446. Metadata properties and values.

Property	Default Value	Description
`com.ibm.portal.layout.template.href`	`dav/:fs-type1/themes/Portal8.5/layout-templates/2ColumnEqual/`	Change this value to say which layout template is used by default on pages in your theme. After initially creating your theme by cloning the IBM WebSphere Portal Express 8.5 theme, be sure to replace `Portal8.5` in the path with the name of the folder for your theme.
`resourceaggregation.profile`	`profiles/profile_deferred.json`	Change this value to say which `profile .json` file (list of modules) to load. The path value is relative to the folder for your theme in `fs-type1`.
`com.ibm.portal.theme.template.ref`	`dav/:fs-type1/themes/Portal8.5/`	Change this value to say in which folder your custom theme's templates are located. Be sure to replace `Portal8.5` with the name of your custom theme folder.

Changing resource environment provider custom properties:

The second order of theme configuration is through resource environment provider (REP) custom properties in the **`WP GlobalThemeConfig`** REP. Changes to the REP custom properties apply across all themes, and the values, therefore, cannot vary from theme to theme.

About this task

The REP custom properties are in the WebSphere Integrated Solutions Console.

Procedure

1. Access the WebSphere Integrated Solutions Console.
2. Go to **Resources** > **Resource Environment** > **Resource Environment Providers** > **WP GlobalThemeConfig**.
3. To view and edit the custom properties for this resource environment provider, click the **Custom properties** link.
4. Modify the property values as needed according to the following table. For each property to modify:
   a. Click the **property name** link.
   b. Modify the value in the **Value** field.
   c. Click **OK**.
5. Save and persist the changes to the master configuration.
6. Restart the portal server.

Results

Resource environment providers use the following properties and values.

`user.displaynameattribute`

Change this value if your identity manager user hierarchy uses a different attribute for the user's full display name. Default value: `cn`

`resources.theme.loadingImage`

The path to the loading image relative to the theme base URI. Default value:`css/images/loading.gif`

Adding resource environment provider properties:

Resource environment provider (REP) custom properties are global in scope, but you can use a consistent naming convention for a theme or module that applies across multiple themes.

About this task

For properties that are used only by a single theme, prefix with *yourTheme*. For those properties that are used only by a single module, prefix with *yourModule*. Modules do not belong to any particular theme and can be used by any theme. It is a good practice to put your modules in one web app and each theme in its own separate web app.

Replace occurrences in italics with the actual names of your items.

Procedure

1. Access the WebSphere Integrated Solutions Console.
2. Go to **Resources** > **Resource Environment** > **Resource Environment Providers**.
3. Select **Node=<node>, Server=WebSphere_Portal**.
4. Click **New**.
5. Enter *YourPrefix* `ThemesConfig` in the **Name** field.
6. Click **OK**.
7. Find your REP in the table and click ***YourPrefix*** **ThemesConfig**.
8. Click **Custom properties**.
9. Create a global and a theme property:
   a. Click **New**.
   b. Enter `modules.contextRoot` in the **Name** field. Because it is global in scope, it does not use any particular prefix.
   c. Enter the context root of your web application that contains your modules in the **Value** field, such as */*`yourprefix`*/modules*.
   d. Click **OK**.
   e. Click **New**.
   f. Enter *`yourTheme`*`.contextRoot` in the Name field. Because it is scoped to your theme, it uses the *yourTheme*. prefix.
   g. Enter the context root of your web application that contains your theme in the **Value** field, such as */*`yourprefix`*/*`yourtheme`.
   h. Click **OK**.
10. Optional: Create example properties that add REP properties for your theme or module:
    a. Click **New**.
    b. Enter *`yourTheme.yourRepProperty`* in the Name field. Because it is scoped to your theme, use the *yourTheme*. prefix.

c. Enter *yourvalue* in the **Value** field.
d. Click **OK**.
e. Click **New**.
f. Enter *yourModule.yourRepProperty* in the Name field. Because it is scoped to your module, use the *yourModule.* prefix.
g. Enter *yourvalue* in the **Value** field.
h. Click **OK**.

11. Save and persist the changes to the master configuration.
12. Restart the portal server.

Configuration for resource aggregation:

Set the following properties in the WP ConfigService Resource Environment Provider to configure the resource aggregator.

Configure theme invalidation auto recognition

This setting defines a regular expression that is used by the server to recognize updates in WebDAV. When the regular expression finds a match, it invalidates the theme caches automatically. This feature does not work on `.WAR` file-based themes. However, it significantly reduces the need for you to invalidate the cache manually.

In the WP ConfigService Resource Environment Provider, modify the `resourceaggregation.auto.invalidate.regex` property. By default, the property is set to `.*\.json|.*/modules/.*`.

For example, updating the file `dav:fs-type1/themes/Portal8.5/contributions/theme.json` through any WebDAV client, is recognized from the default pattern and causes an automatic theme invalidation.

Configure system module invalidation

This setting defines a regular expression that is used by the server to invalidate system modules that are defined in `plugin.xml` files during the theme invalidation process. The regular expression skips all built-in `.WAR` files. If your`.WAR` file is not invalidated, its plug-in ID might start with `com.ibm` or `wp`. Use custom plug-in IDs with the prefix of your company name to ensure that your `.WAR` files are invalidated.

In the WP ConfigService Resource Environment Provider, modify the `resourceaggregation.cache-refresh.skipPluginRegEx` property. By default, the property is set to `^(com\.ibm|wp\.|wcm\.|jquery).*`.

Using your configuration properties:

Create a JavaScript object for your theme on the client side. Theme metadata properties must be loaded dynamically. If you must load a property statically, use a resource environment provider custom property instead.

About this task

Configuration properties can be loaded either statically, `type="config_static"`, or dynamically, `type="config_dynamic"`. You can use one or the other or both

depending on your needs. Static is intended for property values that do not commonly change after they are loaded initially, and is better for performance because the values are cached. Dynamic is intended for property values that change more frequently.

All configuration properties that your module must reference on the client side are merged together into a single convenient global config object. They are merged whether they are theme metadata properties or resource environment provider custom properties. You can then easily reference any property within the *yourcoCfg*.themesConfig object, such as *yourcoCfg*.themesConfig.*yourTheme_yourRepProperty*.

The theme and the modules that are provided with WebSphere Portal Express merge their configuration properties into the ibmCfg global variable. You can run this variable in a browser console and inspect what the property names and values are. Changes to values in the **WP GlobalThemeConfig** REP are global to all themes. You can override certain values on the client side if you need the values to be different in only your themes that include your module. You can do so with syntax similar to the following example in one of the config*.jsp files for your module:

```
i$.merge({<%--
  --%>ibmCfg: {<%--
  --%>themeConfig: {<%--
  --%>loadingImage: "css/images/yourloading.gif"<%--
  --%>}}});
```

If your module requires the module,**wp_portal** then the portal configuration loads before the configuration for your module, ensuring that your override is merged in last.

Replace occurrences of *your** in italics with the actual names of your items.

Procedure

1. Create an extension point (module definition) in the plugin.xml file for your theme with subcontributions for the configuration of the module.

```
<extension point="com.ibm.portal.resourceaggregator.module" id="yourprefix_yourmodule_config">
 <module id="yourprefix_yourmodule">
  <prereq id="wp_portal"/>
  <contribution type="config">
   <sub-contribution type="config_static">
    <uri value="res:{war:context-root}/yourmodule/markup/config_global.jsp" />
   </sub-contribution>
   <sub-contribution type="config_dynamic">
    <uri value="res:{war:context-root}/yourmodule/markup/config.jsp" />
   </sub-contribution>
  </contribution>
 </module>
</extension>
```

2. Add your **moduleID** to the profile file for your theme so that your module loads.
   a. Connect your WebDAV client to http://*localhost*:*port*/wps/mycontenthandler/dav/fs-type1/.
   b. From the profiles folder within the folder for your theme, copy the profile_deferred.json file to a local drive.
   c. Rename the profile_deferred.json file to create a custom profile.
   d. Modify the local copy of the file and add your **moduleID** at the end of the list:

      ,
      "yourprefix_yourmodule"
   e. Copy your custom profile JSON file into the profiles folder in the folder for your theme in fs-type1 on the portal server.

3. Create the `config_global.jsp` file, `config.jsp` file, or both in your modules web application in the *\yourmodule*\markup folder. The following example of `config.jsp` file loads dynamically. A `config_global.jsp` file (static) would look similar, only without any theme metadata properties in it.

```
<%@ page session="false" contentType="text/javascript;charset=ISO-8859-1" buffer="none" %>
<%@ page trimDirectiveWhitespaces="true" %>
<%@ taglib uri="http://www.ibm.com/xmlns/prod/websphere/portal/v8.5/portal-fmt" prefix="portal-fmt" %>
i$.merge({<%--
    --%>yourcoCfg: {<%--
     --%>themesConfig: {<%--
      --%>modules_contextRoot: "<portal-fmt:out escape="json">${wp.rep["REP.YourPrefix ThemesConfig"]["modules.contextRoot"]}</portal-fmt:out>",<%--
      --%>yourTheme_contextRoot: "<portal-fmt:out escape="json">${wp.rep["REP.YourPrefix ThemesConfig"]["yourTheme.contextRoot"]}</portal-fmt:out>",<%--
      --%>yourTheme_yourRepProperty: "<portal-fmt:out escape="json">${wp.rep["REP.YourPrefix ThemesConfig"]["yourTheme.yourRepProperty"]}</portal-fmt:out>",<%--
      --%>yourTheme_yourMetadataProperty: "<portal-fmt:out escape="json">${wp.themeConfig["yourTheme.yourMetadataProperty"]}</portal-fmt:out>"<%--
      --%>yourModule_yourRepProperty: "<portal-fmt:out escape="json">${wp.rep["REP.YourPrefix ThemesConfig"]["yourModule.yourRepProperty"]}</portal-fmt:out>",<%--
--%>}}}});
```

   If the *yourcoCfg* global variable and **themesConfig** child property do not exist, the merge automatically creates them. You can also mix in your properties with any other properties that might already be present if the *yourcoCfg* object exists.
4. If you have JSP reloading turned on in your web applications, then restart the web application for your modules and the web application for your theme. Otherwise, restart the portal server.
5. Restart your portal server.

Configuration settings for capability filters:

Set the following properties in the **Wp ConfigService** Resource Environment Provider to enable or disable the various capability filters.

Enable or disable the runtime filter for the local rendering use case

```
Property : resourceaggregation.enableRuntimePortletCapabilitiesFilter
Possible Values:
true: Enables the runtime filter
false: Disables the runtime filter
Default: true
```

Enable or disable the buildtime filter for the local rendering use case

```
Property : resourceaggregation.enableBuildtimePortletCapabilitiesFilter
Possible Values:
true: Enables the buildtime filter
false: Disables the buildtime filter
Default: true
```

Set the themes to which the filters are applied in the local rendering case

```
Property : resourceaggregation.themeObjectIDs
Possible Values:
Comma separated list of unique names of themes for which the filters should apply.
Default: ibm.portal.80Theme, ibm.portal.85Theme
```

Note: This setting does not apply for the remote rendering use case (WSRP).

Note: CF03 This property is no longer supported in combined cumulative fix 03. The filters are applied for custom themes by default, and you must now disable it for custom themes, if you do not want them filtered. In previous versions, the opposite was true. To disable the filter for any theme, add the metadata `THEME_METADATA_DISABLE_THEME_FILTER` to your theme with the value of `true`. For more information about how to set metadata, see *Change the auto-loading of portlet capabilities*.

CF03

Disable theme filters in local rendering

Theme filters check whether a portlet can be rendered based on the current rendering context or not. This particular filter checks all `capability.*.id` preferences and validates them against the theme capabilities. If a capability is not available, the portlet is either not rendered or is not allowed to be added to the page. For more information, see *Module dependencies in portlets*.

This feature is enabled by default with combined cumulative fix 03 and can be disabled by adding the following theme metadata to your theme.

```
resourceaggregation.disableThemeFilter = true
```

The easiest way to set a theme metadata is through the themelist entry point. Modify the `metadata.properties`. You can also use XMLaccess to update your theme.

Enable or disable the runtime filter for the remote rendering use case (WSRP)

```
Property : resourceaggregation.enableRuntimePortletCapabilitiesFilterForWSRP
Possible Values:
true: Enables the runtime filter for WSRP
false: Disables the runtime filter for WSRP
Default: true
```

Enable or disable the buildtime filter

```
Property : resourceaggregation.enableBuildtimePortletCapabilitiesFilter
Possible Values:
true: Enables the buildtime filter
false: Disables the buildtime filter
Default: true
```

Disable JSON file filter

```
Property: resourceaggregation.load.json.only
Possible values:
true: Filters files other than json files when creating WebDav contribution and profile directories.
false: Does not filter any files in the WebDav contribution and profile directories.
```

Expression language beans for accessing programming models

Expression language (EL) beans are available for accessing WebSphere Programming models. These beans are accessed through the PortalBean represented in the global namespace by `wp`. The beans provide access to IBM WebSphere Portal Express models and associated classes.

PortalBean

The PortalBean provides access to EL Beans that represent some of the WebSphere Portal Express models. You can use these models to access resource information in your JSPs.

You can access the PortalBean (wp) with the following items. Append the following beans to `wp.` to create a bean in your JSP. For example, `wp.themeList.current` gets the current element and returns "Theme" on page 2762.

wp.ac Provides read access to the current access control permissions for a resource.

AccessControl
Provides read access to the current access control node for a resource.

wp.analyticsTagList
Attaches marketing information to Portal resources such as portal pages, portlets, or web content items. Reads the analytics tags that are associated with an identifiable portal resource.

AnalyticsTag
The AnalyticsTag expression bean represents a single analytics tag.

wp.clientProfile
Provides access to the client profile.

wp.identification
Provides access to the identification services serialize and deserialize.

wp.languageList
Provides access to the list of languages that are defined in portal.

Language
Provides access to the interface that represents a portal language.

wp.layoutModel
Provides access to the tree model representation of the layout of a page.

LayoutModel
Provides access to the tree model representation object of the layout of a page.

LayoutContainer
Provides access to the interface that represents a container in a LayoutModel.

LayoutControl
Provides access to the interface that represents a control in a LayoutModel.

wp.localizedDescription
Provides access to the description of the currently rendered page.

Example:

```
${wp.localizedDescription}
```

Returns: Description for the current page description; it is never null.

wp.localizedTitle
Provides access to the title of the currently rendered page.

Example:

```
${wp.localizedTitle}
```

Returns: Title for the current page title; it is never null.

wp.metadata
Provides access to the aggregated metadata of a node. The metadata that can be provided by individual nodes of the content model are combined according to the hierarchy that the content model exposes for these nodes. Values set on the node itself take precedence over values set for its parents.

Example:

```
${wp.metadata[wp.selectionModel.selected]['com.ibm.portal.layout.template.ref']}
```

Returns: Metadata, never null.

CF05 **wp.moduleList**

The module list bean accesses modules that are part of the modularized theme architecture. You can query individual modules and their attributes and identify of the currently used modules within the scope of the currently selected and rendered page and theme.

CF05 **Module**

Represents one individual module.

CF05 **CurrentModuleList**

The current module list always represents the list of modules within the scope of the current request, which is the currently selected page and theme. This module iterates through all modules, locate individual modules and query their capabilities. This is especially useful for portlet developers to check whether certain capabilities are available on the page or not.

CF05 **Module**

Represents one individual module.

CF05 **ModuleCapabilitiesList**

Represents the aggregated capabilities of a list of modules. This is usually the non-deferred or deferred capabilities of the currently selected page and depends on how this object was fetched. It is useful for portlet developers to check whether certain capabilities are available on the page or not.

CF05 **ModuleCapability**

Represents one capability as defined by a module.

wp.navigationModel

Provides access to the navigation model.

NavigationNode

Provides access to a navigation node in a navigation model.

ContentNode

Provides access to a content node. This interface offers a way to obtain the type of the content node.

wp.publicRenderParam

Public render parameters can be used by portlets to share context information. They are addressed with qualified names. The `wp.publicRenderParam` expression bean can be used within a theme or theme module to read the first String value of a public render parameter.

wp.publicRenderParamValues

The `wp.publicRenderParamValues` expression bean can be used within a theme or theme module to read the values of a public render parameter. The public render parameter is read in the context of the currently selected page that is determined internally.

wp.rep

Provides access to the set of configuration entries for the specified resource environment provider.

Properties

Object representing a set of properties mad available as part of the resource environment provider.

wp.selectionModel
Provides access to a selection model for a navigation model.

NavigationNode
Provides access to a navigation node in a navigation model.

ContentNode
Provides access to a content node. This interface offers a way to obtain the type of the content node.

wp.themeConfig
Encapsulates the theme configuration parameter lookup process.

wp.themeList
Provides access to all themes in the system and the currently selected theme.

"Theme" on page 2762
Provides access to the attributes of one theme.

ProfileList
Provides access to all profiles in a theme and the currently selected profile.

Profile
Provides access to the attributes of one profile.

wp.title
Provides access to the title of the currently rendered page or the title information set by a portlet.

Example:

```
${wp.title}
```

Returns: Title in the current locale as a string.

wp.user
Provides access to the active user.

Common beans

Common beans
The following beans are returned by many different beans and are used flexibly.

Description
Provides access to the description of certain objects such as Navigation, Content, Theme, Profile, and others.

Metadata
Provides access to the metadata of certain objects such as Navigation, Content, Theme, Profile, and others.

Title Provides access to the title of certain objects such as Navigation, Content, Theme, Profile, and others.

CF05 **UrlGeneration**
Creates a portal URL you can control with attributes.

CF05 **UrlGenerationPage**
Extends UrlGeneration on a page. All the UrlGeneration attributes are available in addition to these attributes.

CF05 **UrlGenerationPortlet**
Extends UrlGeneration in a portlet. All the UrlGeneration and UrlGenerationPage attributes are available in addition to these attributes.

Other Beans

These beans cannot be accessed through the PortalBean. To access the uiNavigationModel Bean, you must use the uiNavigationModel tag and define the variable name that you want the bean to be available under.

CF05 **uiNavigationModel**
By default, the uiNavigationModel lists the visible pages as part of its iterator. When the **Show hidden pages** option is selected in the toolbar, it also lists the hidden pages. There is a mobile hidden flag for pages. The model also helps you specify a mobile test device class expression, which is used to evaluate if the system is rendered as part of a mobile request.

CF05 **uiNavigationNode**
Provides access to a navigation node in a navigation model.

wp.ac - Access control:

Provides read access to the current access control permissions for a resource.

Example: `${wp.ac[node].hasPermission['editor']}`

Attributes:

get(id) Gets an individual access control node.

Example:

```
${wp.ac[wp.selectionModel.selected].selected}
```

Parameters:

id String or Identifiable object to look up the access control objects, must not be null.

Returns: AccessControl; it can be null if it is not found.

AccessControl:

Provides read access to the current access control permissions for a resource.

Attributes:

hasAddChildPermission
Determine whether the page is a non-private page and the current user has contributor, editor, manager, or admin permission for the specified page.

Example:

```
${wp.ac[wp.selectionModel.selected].hasAddChildPermission}
```

Parameters: none

Returns: Boolean; true if the check is successful, otherwise false.

hasEditSharedPermission
Determines whether the page is a non-private page and the current user has editor, manager, or admin permission for the specified page.

Example:

```
${wp.ac[wp.selectionModel.selected].hasEditSharedPermission}
```

Parameters: none

Returns: Boolean; true if the check is successful, otherwise false.

hasPersonalizePermission

Determines whether the specified page is a private page and the current user has PRIVILEGED_USER permission.

Example:

```
${wp.ac[wp.selectionModel.selected].hasPersonalizePermission}
```

Parameters: none

Returns: Boolean; true if the check is successful, otherwise false.

hasPermission

Determines whether the resource has a specific permission. Attribute is the case in-sensitive name of the `com.ibm.portal.ac.data.RoleType`.

Example:

```
${wp.ac[wp.selectionModel.selected].hasPermission['Administrator']}
${wp.ac[wp.selectionModel.selected].hasPermission['ManageR']}
${wp.ac[wp.selectionModel.selected].hasPermission['editor']}
```

Parameters: none

Returns: Boolean; true if the check is successful, otherwise false.

isPrivate

Determines whether the specified page is private.

Example:

```
 ${wp.ac[wp.selectionModel.selected].isPrivate}
```

Parameters: none

Returns: Boolean; true if the page is private, false if it is not.

wp.analyticsTagList:

Attaches marketing information to Portal resources such as portal pages, portlets, or web content items. Reads the analytics tags that are associated with an identifiable portal resource.

For more information, see “Analytics tags and site promotions” on page 1724

Attributes:

get(objectid)

Determines the list of analytics tags for the portal resource with the given objectID.

Example:

```
<c:forEach items="${wp.analyticsTagList[wp.identification[wp.selectionModel.selected]]}" v
     <span class="asa.tag.<c:out value='${current.name}'/>"><c:out value='${current.value}
</c:forEach>
```

Parameters:

objectid
An ObjectID, or Identifiable object to locate the resource; must not be null.

Returns: A list containing AnalyticsTag objects. You can use the forEach JSTL tag to iterate through the returned list.

AnalyticsTag:

The AnalyticsTag expression bean represents a single analytics tag.

Attributes:

name

Returns the name of the analytics tag.

Example:

```
<c:forEach items="${wp.analyticsTagList[wp.identification[wp.selectionModel.selected]]}" var
    <span class="asa.tag.<c:out value='${current.name}'/>"><c:out value='${current.value}' />
</c:forEach>
```

Parameters: none

Returns: A string representing the name of the analytics tag. It is never null.

value

Returns the value of the analytics tag.

Example:

```
<c:forEach items="${wp.analyticsTagList[wp.identification[wp.selectionModel.selected]]}" var
    <span class="asa.tag.<c:out value='${current.name}'/>"><c:out value='${current.value}' />
</c:forEach>
```

Parameters: none

Returns: A string representing the value of the analytics tag. It is never null.

wp.clientProfile:

Provides access to the client profile.

Attributes:

get(attribute)
Provides access to the defined attributes associated with the client/CCPP profile.

Example:

```
${wp.clientProfile['DeviceClass']}
```

The following example shows how to access the device class information for the client profile:

```
<c:set var="deviceClass" scope="request" value="${wp.clientProfile['DeviceClass']}" />
<c:choose>
<c:when test="${deviceClass == 'desktop'}">
<jsp:forward page="/jsp/html/desktop/View.jsp"/>
</c:when>
<c:when test="${deviceClass == 'tablet'}">
<jsp:forward page="/jsp/html/tablet/View.jsp"/>
</c:when>
```

```
<c:otherwise>
<jsp:forward page="/jsp/html/View.jsp"/>
</c:otherwise>
</choose>
```

Parameters:

attribute
String; the name of the client attribute you want to retrieve.

Returns: String; the value of the client profile's attribute name. Can be null.

wp.identification:

Provides access to the identification services to serialize and deserialize objectIDs.

Attributes:

get(object)
Serializes the objectID into a string.

Example:

```
${wp.identification[object]}
```

Parameters:

object
Identifiable object to be serialized.

Returns: String, a serialized representation of an object. It is null if it is not a valid object.

get(string)
Deserializes a string into an objectID.

Example:

```
${wp.identification[string]}
```

Parameters:

string
String to look up the identification objects, must not be null.

Returns: objectID; it can be null if the string is invalid or the object is not found.

wp.languageList:

The wp.languageList provides access to the list of languages that are defined in portal.

Attributes:

get(id) Get an individual language object.

Example:

```
${wp.languageList[id]}
```

Parameters:

id Identifiable object to look up the Language objects, must not be null.

Returns: Language object; It can be null if not found.

iterator

Iterates through languages.

Example:

```
<c:forEach var="node" items="${wp.languageList}">
</c:forEach>
```

Parameters: None

Returns: An iterator with Language objects; it is never null.

Language:

Provides access to the interface that represents a portal language.

Attributes:

created

Returns the creation date of the language.

Example:

```
<c:forEach var="node" items="${wp.languageList}">
     ${node.created}
</c:forEach>
```

Parameters: none

Returns: Date, the creation date object of the resource. It is never null.

description

The description of the language.

Example:

```
<c:forEach var="node" items="${wp.languageList}">
     ${node.description}
</c:forEach>
```

Parameters: none

Returns: Description object for the language node; it is never null. You can use the value of the title object to retrieve the description in current locale.

lastModified

Returns the last modification date of the language.

Example:

```
<c:forEach var="node" items="${wp.languageList}">
    ${node.lastModified}
</c:forEach>
```

Parameters: none

Returns: Date, the last modified date object of the resource. It is never null.

locale Returns the locale of the language.

Example:

```
<c:forEach var="node" items="${wp.languageList}">
     ${node.locale}
</c:forEach>
```

Parameters: none

Returns: Locale, the locale object of the resource. It is never null.

objectID

Returns the obect identifier for the current language.

Example:

```
<c:forEach var="node" items="${wp.languageList}">
     ${node.objectID}
</c:forEach>
```

Parameters: none

Returns: objectID associated with this language.

title The title of this language.

Example:

```
<c:forEach var="node" items="${wp.languageList}">
     ${node.title}
</c:forEach>s
```

Parameters: none

Returns: Title object for the language node; it is never null. You can use the value of the title object to retrieve the title in current locale.

wp.layoutModel:

The wp.layoutModel provides access to the tree model representation of the layout of a page.

Attributes:

children(node)
Returns an iterator of child nodes.

Example:

```
<c:forEach var="node" items="${wp.layoutModel.children[wp.identification[wp.selectionModel
 ${node}<br>
</c:forEach>
```

Parameters:

node
Identifiable, or NavigationNode object to look up the children; it must not be null.

Returns: An iterator with NavigationNode objects; it is never null.

get(id) Get the layout model for a navigation node or navigation node ID.

Example:
The following example displays getting the layout model for the currently selected node from the selection model. The second line retrieves the children of the root mode of the layout model.

```
<c:set var="layoutmodel" value="${wp.layoutModel[wp.selectionModel.selected]}"/>
<c:set var="containers" value="${layoutmodel.children[layoutmodel.root]}"/>
```

Parameters:

id String or identifiable object of the navigation object; must not be null.

Returns: LayoutModel for the navigation node. Can be null.

hasChildren
Determines whether the specified NavigationNode has associated nodes.

Example:

```
${wp.layoutModel.hasChildren[wp.selectionModel.selected]}
```

Parameters:

node
Identifiable, or NavigationNode object to look up the children; it must not be null.

Returns: Boolean; true if the node has children. Otherwise, it is false.

parent

Access to the parent of a NavigationNode.

Example:

```
${wp.layoutModel.parent[wp.selectionModel.selected]}
```

Parameters:

node
Identifiable or NavigationNode object to look up the children; it must not be null.

Returns: LayoutNode; the parent node for the node if there is a parent. Otherwise, it is null.

path(node)
Provides access to the path information for the node. The path represents the hierarchy from the root to the give node as a list. It is like a breadcrumb.

Example:

```
<c:forEach var="node" items="${wp.layoutModel.path[wp.selectionModel.selected]}">
 &lt;- ${node}
</c:forEach>
```

Parameters:

node
Identifiable or NavigationNode object to look up the children; it must not be null.

Returns: a list of LayoutNodes representing the path from the root to the node.

root Returns the root node of the layout model.

Example:

```
${wp.layoutModel.root}
```

Parameters: none

Returns: NavigationNode; it is never null.

LayoutModel:

Provides access to the tree and list model representation object of the layout of a page.

Many of the following attributes return LayoutNode. However, LayoutContainer and LayoutControl are types of LayoutNode, and when you iterate, you can encounter either type.

Attributes:

children
Returns the children of the specified node.

Example:

```
${wp.layoutModel[wp.selectionModel.selected].children[node]}
```

Parameters:

node
This value represents an entry in a model.

Returns: Children of the specified node.

get(id) Gets an individual layout model.

Example:

```
${wp.layoutModel[id]}
```

Parameters:

id String or Identifiable object to look up the layout objects. It must not be null.

Returns: LayoutNode; it can be null if not found.

hasChildren
Determines whether the current specified LayoutNode has associated nodes.

Example:

```
${wp.layoutModel[wp.selectionModel.selected].hasChildren[node]}
```

Parameters:

node
LayoutNode; node to check for child nodes.

Returns: Boolean; true if the layout node has children. Otherwise, it is false.

iterator
Iterates through layout nodes.

Example:

```
<c:forEach var="node" items="${wp.layoutModel[wp.selectionModel.selected]}">
</c:forEach>
```

Parameters: None

Returns: an iterator with LayoutNode objects; it is never null.

parent

Access to the parent of a node.

Example:

```
${wp.layoutModel[wp.selectionModel.selected].parent[node]}
```

Parameters:

node
LayoutNode; the current node.

Returns: LayoutNode; the parent node for the node if there is a parent. Otherwise, it is null.

path Provides access to the path information for the node. The path represents the hierarchy from the root to the give node as a list. It is like a breadcrumb.

Example:

```
${wp.layoutModel[wp.selectionModel.selected].path[node]}
```

Parameters:

node
LayoutNode; the current node.

Returns: a list of LayoutNodes representing the path from the root to a node.

root Returns the root node of the layout model.

Example:

```
${wp.layoutModel[wp.selectionModel.selected].root}
```

Parameters: none

Returns: LayoutNode; it is never null.

LayoutContainer:

Provides access to the interface that represents a container in a LayoutModel.

Attributes:

description
The description of the layout container.

Example:

```
${wp.layoutModel[wp.selectionModel.selected].root.description}
```

Parameters: none

Returns: Description object for the layout container; it is never null. You can use the value of the title object to retrieve the description in current locale.

metadata
Metadata that is associated with this layout container.

Example:

```
${wp.layoutModel[wp.selectionModel.selected].root.metadata}
```

Parameters: none

Returns: Metadata, never null.

objectID
Returns the obect identifier for the current layour container.

Example:

```
${wp.layoutModel[wp.selectionModel.selected].root.objectID}
```

Parameters: none

Returns: Object Identifier for the current control. The following displays an example of getting the layout model for the currently selected node from the selection model. The second line retrieves the children of the root node of the layout model. The children of the root node is represented as LayoutContainer.

```
<c:set var="layoutmodel" value="${wp.layoutModel[wp.selectionModel.selected]}"/>
<c:set var="containers" value="${layoutmodel.children[layoutmodel.root]}"/>
```

title The title of this container.

Example:

```
${wp.layoutModel[wp.selectionModel.selected].root.title}
```

Parameters: none

Returns: Title associated with the current layout container object.

LayoutControl:

Provides access to the interface that represents a control in a LayoutModel.

Attributes:

description
The description of the current control.

Example:

```
${wp.layoutModel[wp.selectionModel.selected].root.description}
```

Parameters: none

Returns: Description object for the current control; it is never null. You can use the value of the title object to retrieve the description in current locale.

metadata
The metadata map of the current control.

Example:

```
${wp.layoutModel[wp.selectionModel.selected].root.metadata}
```

Parameters: none

Returns: Metadata, never null.

objectID
Description

Example:

```
${wp.layoutModel[wp.selectionModel.selected].root.objectID}
```

Parameters:

Returns: Object identifier for the current control.

title The title of the current control.

Example:

```
${wp.layoutModel[wp.selectionModel.selected].root.title}
```

Parameters: none

Returns: Title object for the current control; it is never null. You can use the value of the title object to retrieve the title in current locale.

wp.metadata:

Provides access to the aggregated metadata of a node. The metadata that can be provided by individual nodes of the content model are combined according to the hierarchy that the content model exposes for these nodes. Values set on the node itself take precedence over values set for its parents.

Attributes:

get(id) Gets the aggregated metadata map of the content node that is specified.

Example:

```
${wp.metadata[wp.selectionModel.selected]}
```

Parameters:

id String or Identifiable object to look up the content node; it must not be null.

Returns: Metadata; It can be null if not found.

wp.moduleList:

The module list bean accesses modules that are part of the modularized theme architecture. You can query individual modules and their attributes and identify of the currently used modules within the scope of the currently selected and rendered page and theme.

Attributes:

current

Returns the module list for the currently selected page and theme.

Example:

```
<c:forEach var="node" items="${wp.moduleList.current}">
    ${node.name}/${node.version},
</c:forEach>
```

Parameters: none

Returns: CurrentModuleList; Never null.

get(moduleKey)

Returns the module for the module key. The module key must be in the format *<module name>* or *<module name>/<module version>*. The version is optional. If no version is used, then the module with the highest version is selected.

Example:

```
${wp.moduleList["wp_portal"]}
${wp.moduleList["wp_portal/0.0"]}
```

Parameters:

moduleKey

String representing the module. Must be in the format *<module name>* or *<module name>/<module version>*.

Returns: Module. It can be null if it is not found.

Module:

Represents one individual module

Attributes:

name Returns the name of the module.

Example:

```
<c:forEach var="node" items="${wp.moduleList.current}">
    ${node.name}/${node.version},
</c:forEach>
```

Parameters: none

Returns: String; never null.

version

Returns the version of the module.

Example:

```
<c:forEach var="node" items="${wp.moduleList.current}">
    ${node.name}/${node.version},
</c:forEach>
```

Parameters: none

Returns: String; never null.

CurrentModuleList:

The current module list always represents the list of modules within the scope of the current request, which is the currently selected page and theme. This module iterates through all modules, locates individual modules and queries their capabilities. It is useful for portlet developers to check whether certain capabilities are available on the page or not.

Attributes:

iterator

Iterates through modules.

Example:

```
<c:forEach var="node" items="${wp.moduleList.current}">
    ${node.name}/${node.version},
</c:forEach>
```

Parameters: None

Returns: An iterator with Module objects; it is never null.

capabilities

Returns the aggregated non-deferred capabilities for all modules that are part of the currently selected page and theme.

Example:

```
<c:forEach var="node" items="${wp.moduleList.current.capabilities}">
    ${node.name}/${node.value},
</c:forEach>
```

Parameters: none

Returns: ModuleCapabilitiesList; Never null.

deferredCapabilities

Returns the aggregated deferred capabilities for all modules that are part of the currently selected page and theme.

Example:

```
<c:forEach var="node" items="${wp.moduleList.current.deferredCapabilities}">
    ${node.name}/${node.value},
</c:forEach>
```

Parameters: none

Returns: ModuleCapabilitiesList; Never null.

get(moduleKey)

Returns the module for the module key. The module key must be in the format *<module name>* or *<module name>/<module version>*. The version is optional. If no version is used, then the module with the highest version is selected.

Example:

```
${wp.moduleList.current["wp_portal"]}
${wp.selectionModel.selected.moduleList["wp_portal"]}
```

Parameters:

moduleKey

String representing the module. Must be in the format *<module name>* or *<module name>/<module version>*.

Returns: Module. Can be null if it is not found.

Module:

Represents one individual module

Attributes:

name Returns the name of the module.

Example:

```
<c:forEach var="node" items="${wp.moduleList.current}">
    ${node.name}/${node.version},
</c:forEach>
```

Parameters: none

Returns: String; never null.

version

Returns the version of the module.

Example:

```
<c:forEach var="node" items="${wp.moduleList.current}">
    ${node.name}/${node.version},
</c:forEach>
```

Parameters: none

Returns: String; never null.

ModuleCapabilitiesList:

Represents the aggregated capabilities of a list of modules. This is usually the non-deferred or deferred capabilities of the currently selected page and depends on how this object was fetched. This is especially useful for portlet developers to check if certain capabilities are available on the page or not.

Attributes:

get(key)

Returns the module for a key.

Example:

```
${wp.moduleList.current.capabilities['oneUI'].value
${!empty wp.moduleList.current.capabilities['oneUI']}
```

Parameters:

key

String, the name of the capability.

Returns: ModuleCapability. May be null if it is not found.

iterator

Iterates through capabilities.

Example:

```
<c:forEach var="node" items="${wp.moduleList.current.capabilities}">
    ${node.name}/${node.value},
</c:forEach>
```

Parameters: None

Returns: An iterator with ModuleCapability objects; it is never null.

ModuleCapability:

Represents one capability as defined by a module.

Attributes:

name Returns the name of the capability.

Example:

```
${wp.moduleList.current.capabilities['oneUI'].name}
```

Parameters: none

Returns: String; never null.

version

Returns the version of the capability.

Example:

```
${wp.moduleList.current.capabilities['oneUI'].value}
```

Parameters: none

Returns: String; never null.

wp.navigationModel:

The wp.navigationModel provides access to the navigation model. The navigation model is a tree representation of all pages.

Attributes:

children(node)

Returns an iterator of child nodes.

Example:

```
<c:forEach var="node" items="${wp.navigationModel.children[wp.identification[wp.selectionM
 ${node}<br>
</c:forEach>
```

Parameters:

node

Identifiable, or NavigationNode object to look up the children; it must not be null.

Returns: An iterator with NavigationNode objects; it is never null.

get(id) Get an individual navigation node.

Example:

```
${wp.navigationModel[id]}
```

Parameters:

id String or Identifiable object to look up the navigation object; it must not be null.

Returns: NavigationNode; it can be null if not found.

hasChildren

Determines whether the specified NavigationNode has associated nodes.

Example:

```
${wp.navigationModel.hasChildren[wp.selectionModel.selected]}
```

Parameters:

node

Identifiable, or NavigationNode object to look up the children; it must not be null.

Returns: Boolean; true if the node has children. Otherwise, it is false.

parent

Access to the parent of a NavigationNode.

Example:

```
${wp.navigationModel.parent[wp.selectionModel.selected]}
```

Parameters:

node

Identifiable or NavigationNode object to look up the children; it must not be null.

Returns: NavigationNode; the parent node for the node if there is a parent. Otherwise, it is null.

path(node)

Provides access to the path information for the node. The path represents the hierarchy from the root to the give node as a list. It is like a breadcrumb.

Example:

```
<c:forEach var="node" items="${wp.navigationModel.path[wp.selectionModel.selected]}">
 &lt;- ${node}
</c:forEach>
```

Parameters:

node

Identifiable or NavigationNode object to look up the children; it must not be null.

Returns: a list of NavigationNodes representing the path from the root to the node.

root Returns the root node of the navigation model.

Example:

```
${wp.navigationModel.root}
```

Parameters: none

Returns: NavigationNode; it is never null.

NavigationNode:

Represents a navigation node in a navigation model.

Attributes:

contentNode
Returns the content node that is associated with the navigation node.

Example:

```
${wp.selectionModel.selected.contentNode}
${wp.navigationModel.selected.contentNode}
```

Parameters: none

Returns: ContentNode. It is never null.

description
The description of the navigation node.

Example:

```
${wp.selectionModel.selected.description}
${wp.navigationModel.selected.description}
```

Parameters: none

Returns: Description object for the navigation node; it is never null. You can use the value of the title object to retrieve the description in current locale.

isPrivate
Determines whether the current node is private.

Example:

```
${wp.selectionModel.selected.isPrivate}
${wp.navigationModel.selected.isPrivate}
```

Parameters: none

Returns: Boolean, true if the page is private, otherwise false.

metadata
The metadata map of this navigation node.

Example:

```
${wp.selectionModel.selected.metadata['com.ibm.portal.Hidden']}
${wp.navigationModel.selected.metadata['com.ibm.portal.Hidden']}
```

Parameters: none

Returns: Metadata, never null.

CF05 **moduleList**
Returns the module list for the currently selected page and theme.

Example:

```
<c:forEach var="node"items="${wp.selectionModel.selected.moduleList}">
    ${node.name}/${node.version},
</c:forEach>
<c:forEach var="node"items="${wp.navigationModel.selected.moduleList}">
    ${node.name}/${node.version},
</c:forEach>
```

Parameters: none

Returns: CurrentModuleList, never null.

objectID

Returns the ObjectID associated with this navigation node.

Example:

```
${wp.selectionModel.selected.objectID}
${wp.navigationModel.selected.objectID}
```

Parameters: none

Returns: ObjectID. Never null.

profileRef

Returns the profile reference for the page. If it is empty or null, the default theme profile reference is used.

Example:

```
${wp.selectionModel.selected.profileRef}
${wp.navigationModel.selected.profileRef}
```

Parameters: none

Returns: String, the profile reference within the theme. It can be null if you are using a non-modularized theme.

projectID

Returns the project identifier that is associated with this navigation node, or null of no project is associated.

Example:

```
${wp.selectionModel.selected.projectID}
${wp.navigationModel.selected.projectID}
```

Parameters: none

Returns: String representing the associated project. It can be null if no project is associated.

themeID

Returns the set theme ID for the page. If it is not set for the page, this function returns the inherited theme or default system theme.

Example:

```
${wp.selectionModel.selected.themeID}
${wp.navigationModel.selected.themeID}
```

Parameters: none

Returns: ObjectID of the referenced theme. Never null.

title The title of this navigation node.

Example:

```
${wp.selectionModel.selected.title}
${wp.navigationModel.selected.title}
```

Parameters: none

Returns: Title associated with the current object.

CF05 **url**

Short hand for urlGeneration that returns a string and cannot be manipulated any further.

Example:

```
${wp.selectionModel.selected.url}
${wp.navigationModel.selected.url}
```

Parameters: none

Returns: String; the URL pointing to this page.

CF05 **urlGeneration**

Creates a portal URL you can control with attributes. The URL attributes can be set by using further methods on the UrlGeneration object as shown in the examples section.

Example:

```
${wp.selectionModel.selected.urlGeneration}
${wp.navigationModel.selected.urlGeneration}
```

More examples:

```
<c:set var="node" value="${wp.selectionModel.selected}"/>
<a href="${node.url}">Simple URL, no modifications possible</a>
<a href="${node.urlGeneration}">Simple URL</a>
<a href="${node.urlGeneration.keepNavigationalState}">With NavState</a>
<a href="${node.urlGeneration.noNavigationalState}">Without NavState</a>
<a href="${node.urlGeneration.setThemeTemplate('Plain')}">With ThemeTemplate</a>
<a href="${node.urlGeneration.forcePublic}">Public Link</a>
<a href="${node.urlGeneration.secure}">Secure Link</a>
<a href="${node.urlGeneration.setLocale('de')}">In Deutsch</a>
<a href="${node.urlGeneration.setParam('a','b')}">With Params</a>
<a href="${node.urlGeneration.setParam('a','b').setParam('c','d').forcePublic.setLocale('c
Complex URL</a>
<a href="${node.urlGeneration.logout}">Logout</a>
<a href="${node.urlGeneration.login}">Login</a>
<a href="${node.urlGeneration.normalize}">Normalized URL</a>
<a href="${node.urlGeneration.allowRelativeURL}">Relative URL</a>
<a href="${node.urlGeneration.disallowRelativeURL}">Disallow Relative URL</a>
<a href="${node.urlGeneration.forceAbsolute}">Absolute URL</a>
```

```
<c:set var="node" value="${wp.navigationModel.selected}"/>
<a href="${node.url}">Simple URL, no modifications possible</a>
<a href="${node.urlGeneration}">Simple URL</a>
<a href="${node.urlGeneration.keepNavigationalState}">With NavState</a>
<a href="${node.urlGeneration.noNavigationalState}">Without NavState</a>
<a href="${node.urlGeneration.setThemeTemplate('Plain')}">With ThemeTemplate</a>
<a href="${node.urlGeneration.forcePublic}">Public Link</a>
<a href="${node.urlGeneration.secure}">Secure Link</a>
<a href="${node.urlGeneration.setLocale('de')}">In Deutsch</a>
<a href="${node.urlGeneration.setParam('a','b')}">With Params</a>
<a href="${node.urlGeneration.setParam('a','b').setParam('c','d').forcePublic.setLocale('c
Complex URL</a>
<a href="${node.urlGeneration.logout}">Logout</a>
<a href="${node.urlGeneration.login}">Login</a>
<a href="${node.urlGeneration.normalize}">Normalized URL</a>
<a href="${node.urlGeneration.allowRelativeURL}">Relative URL</a>
<a href="${node.urlGeneration.disallowRelativeURL}">Disallow Relative URL</a>
<a href="${node.urlGeneration.forceAbsolute}">Absolute URL</a>
```

Parameters: none

Returns: UrlGenerationPage; the URL object pointing to this page

ContentNode:

Provides access to a content node. This interface offers a way to obtain the type of the content node.

Attributes:

contentNodeType

Returns the type of this content node as `com.ibm.portal.content.ContentNodeType`.

Example:

```
${wp.navigationModel.selected.contentNode.contentNodeType}
${wp.selectionModel.selected.contentNode.contentNodeType}
```

The following example displays how to check whether the currently selected node is a label:

```
<c:if test="${wp.navigationModel.selected.contentNode.contentNodeType == 'LABEL'}">
<c:if test="${wp.selectionModel.selected.contentNode.contentNodeType == 'LABEL'}">
```

Parameters: none

Returns: `com.ibm.portal.content.ContentNodeType – COMPOSITION, EXTERNALURL, LABEL, PAGE, STATICPAGE`

description

The description of the content node.

Example:

```
${wp.navigationModel.selected.contentNode.description}
${wp.selectionModel.selected.contentNode.description}
```

Parameters: none

Returns: ContentNode object for the navigation node; it is never null. You can use the value of the title object to retrieve the description in current locale.

metadata

The metadata map of this content node.

Example:

```
${wp.navigationModel.selected.contentNode.metadata['com.ibm.portal.Hidden']}
${wp.selectionModel.selected.contentNode.metadata['com.ibm.portal.Hidden']}
```

Parameters: none

Returns: Metadata, never null.

CF05 **moduleList**

Returns the module list for the currently selected page and theme.

Example:

```
<c:forEach var="node" items="${wp.navigationModel.selected.contentNode.moduleList}">
 ${node.name}/${node.version},
</c:forEach>
<c:forEach var="node" items="${wp.selectionModel.selected.contentNode.moduleList}">
 ${node.name}/${node.version},
</c:forEach>
```

Parameters: none

Returns: CurrentModuleList, never null.

objectID

Returns the ObjectID associated with this content node.

Example:

```
${wp.navigationModel.selected.contentNode.objectID}
${wp.selectionModel.selected.contentNode.objectID}
```

Parameters: none

Returns: ObjectID. Never null.

profileRef

Returns the profile reference for the page. If it is empty or null, the default theme profile reference is used.

Example:

```
${wp.navigationModel.selected.contentNode.profileRef}
${wp.selectionModel.selected.contentNode.profileRef}
```

Parameters: none

Returns: String, which represents the reference to a profile that exists within the theme. It can be null if you are using a non-modularized theme.

themeID

Returns the set theme ID for the page. If it is not set for the page, this function returns the inherited theme or default system theme.

Example:

```
${wp.navigationModel.selected.contentNode.themeID}
${wp.selectionModel.selected.contentNode.themeID}
```

Parameters: none

Returns: ObjectID of the referenced theme. Never null.

title The title of this content node.

Example:

```
${wp.navigationModel.selected.contentNode.title}
${wp.selectionModel.selected.contentNode.title}
```

Parameters: none

Returns: Title associated with the current object.

CF05 **url**

Short hand for urlGeneration that returns a string and cannot be manipulated any further.

Example:

```
${wp.navigationModel.selected.url}
${wp.selectionModel.selected.url}
```

Parameters: none

Returns: String; the URL pointing to this page.

CF05 **urlGeneration**

Creates a portal URL you can control with attributes. The URL attributes can be set by using further methods on the UrlGeneration object as shown in the examples section.

Example:

```
${wp.navigationModel.selected.urlGeneration}
```

```
${wp.selectionModel.selected.urlGeneration}
```

More examples:

```
<c:set var="node" value="${wp.navigationModel.selected}"/>
<a href="${node.url}">Simple URL, no modifications possible</a>
<a href="${node.urlGeneration}">Simple URL</a>
<a href="${node.urlGeneration.keepNavigationalState}">With NavState</a>
<a href="${node.urlGeneration.noNavigationalState}">Without NavState</a>
<a href="${node.urlGeneration.setThemeTemplate('Plain')}">With ThemeTemplate</a>
<a href="${node.urlGeneration.forcePublic}">Public Link</a>
<a href="${node.urlGeneration.secure}">Secure Link</a>
<a href="${node.urlGeneration.setLocale('de')}">In Deutsch</a>
<a href="${node.urlGeneration.setParam('a','b')}">With Params</a>
<a href="${node.urlGeneration.setParam('a','b').setParam('c','d').forcePublic.setLocale('de')
Complex URL</a>
<a href="${node.urlGeneration.logout}">Logout</a>
<a href="${node.urlGeneration.login}">Login</a>
<a href="${node.urlGeneration.normalize}">Normalized URL</a>
<a href="${node.urlGeneration.allowRelativeURL}">Relative URL</a>
<a href="${node.urlGeneration.disallowRelativeURL}">Disallow Relative URL</a>
<a href="${node.urlGeneration.forceAbsolute}">Absolute URL</a>
```

```
<c:set var="node" value="${wp.selectionModel.selected}"/>
<a href="${node.url}">Simple URL, no modifications possible</a>
<a href="${node.urlGeneration}">Simple URL</a>
<a href="${node.urlGeneration.keepNavigationalState}">With NavState</a>
<a href="${node.urlGeneration.noNavigationalState}">Without NavState</a>
<a href="${node.urlGeneration.setThemeTemplate('Plain')}">With ThemeTemplate</a>
<a href="${node.urlGeneration.forcePublic}">Public Link</a>
<a href="${node.urlGeneration.secure}">Secure Link</a>
<a href="${node.urlGeneration.setLocale('de')}">In Deutsch</a>
<a href="${node.urlGeneration.setParam('a','b')}">With Params</a>
<a href="${node.urlGeneration.setParam('a','b').setParam('c','d').forcePublic.setLocale('de')
Complex URL</a>
<a href="${node.urlGeneration.logout}">Logout</a>
<a href="${node.urlGeneration.login}">Login</a>
<a href="${node.urlGeneration.normalize}">Normalized URL</a>
<a href="${node.urlGeneration.allowRelativeURL}">Relative URL</a>
<a href="${node.urlGeneration.disallowRelativeURL}">Disallow Relative URL</a>
<a href="${node.urlGeneration.forceAbsolute}">Absolute URL</a>
```

Parameters: none

Returns: UrlGenerationPage; the URL object pointing to this page

wp.publicRenderParam:

Public render parameters can be used by portlets to share context information. They are addressed with qualified names. The `wp.publicRenderParam` expression bean can be used within a theme or theme module to read the first String value of a public render parameter.

The public render parameter is read in the context of the currently selected page that is determined internally. If your public render parameter has multiple values, you must use the wp.publicRenderParamValues expression bean instead.

Attributes:

get(qname)

Reads the first string value of the public render parameter that is identified by the qualified name.

Example:

```
<c:set var="editModeQName" value="{http://www.ibm.com/xmlns/prod/websphere/portal/publicpa
<c:set var="isEditModeActive" value="${wp.publicRenderParam[editModeQName]}" />
<c:if test="${isEditModeActive}">
    ...
</c:if>
```

Parameters:

qname
The serialized qualified name of the public render parameter. Its format is {<namespaceURI>}<localname>. This format corresponds with the serialization format defined by the javax.xml.namespace.QName class. To avoid issues with the JSP expression parser, store the qualified name in a separate JSP variable. The qname parameter is mandatory. It must not be null.

Returns: A String representing the first value of the addressed public render parameter or null.

wp.publicRenderParamValues:

The wp.publicRenderParamValues expression bean can be used within a theme or theme module to read the values of a public render parameter. The public render parameter is read in the context of the currently selected page that is determined internally.

Attributes:

get(qname)

Reads the first string value of the public render parameter that is identified by the qualified name.

Example:

```
<c:set var="pageModesQName" value="{http://www.ibm.com/xmlns/prod/websphere/portal/publicp
<c:set var="pageModes" value="${wp.publicRenderParamValues[pageModesQName]}" />
<c:forEach var="mode" items="${pageModes}">
    ...
</c:forEach>
```

Parameters:

qname
The serialized qualified name of the public render parameter. Its format is {<namespaceURI>}<localname>. This format corresponds with the serialization format defined by the javax.xml.namespace.QName class. To avoid issues with the JSP expression parser, store the qualified name in a separate JSP variable. The **qname** parameter is mandatory. It must not be null.

Returns: A String array that contains the values of the addressed public render parameter or null. You can use the forEach JSTL tag to loop over the returned array.

wp.rep - Resource Environment Provider:

The wp.rep provides access to the set of configuration entries for the specified resource environment provider.

Attributes:

get(provider key)

Retrieves the property that is defined in the Resource Environment Provider.

Example:

```
${wp.rep['resourceEnvironmentProvider']['property']}
```

To retrieve the Portal defined Dojo context root.

```
${wp.rep['WP GlobalThemeConfig']['resources.js.dojo.contextRoot']}
```

Parameters: String; specifies the provider key. .

Returns: Properties; it can be null if not found.

Properties:

Object representing a set of properties made available as part of the resource environment provider.

Attributes:

get(key)

Retrieves the property that is defined on the Resource Environment Provider.

Example:

```
${wp.rep['resourceEnvironmentProvider']['property']}
```

To retrieve the Portal defined Dojo context root.

```
${wp.rep['WP GlobalThemeConfig']['resources.js.dojo.contextRoot']}
```

Parameters: String; specifies the property key.

Returns: String; specifies the property value. It can be null if not found.

iterator

Retrieves all properties that are stored in the resource environment provider.

Example:

```
<c:forEach var="node" items="${wp.rep['WP GlobalThemeConfig']}">
${node.key}  =>  ${node.value}<br>
</c:forEach>
```

Parameters: None

wp.selectionModel:

The NavigationSelectionModel provides access to a selection model for a navigation model. The selection model is the navigation model from the perspective of the currently selected page, from which you can get to the selected page's ancestors and descendents.

Attributes:

children(node)

Returns an iterator of child nodes.

Example:

```
<c:forEach var="node" items="${wp.selectionModel.children[wp.identification[wp.selectionMode
 ${node}<br>
</c:forEach>
```

Parameters:

node
Identifiable, or NavigationNode object to look up the children; it must not be null.

Returns: An iterator with NavigationNode objects; it is never null.

hasChildren
Determines whether the specified NavigationNode has associated nodes.

Example:

```
${wp.selectionModel.hasChildren[wp.selectionModel.selected]}
```

Parameters:

node
Identifiable, or NavigationNode object to look up the children; it must not be null.

Returns: Boolean; true if the node has children. Otherwise, it is false.

parent

Access to the parent of a NavigationNode.

Example:

```
${wp.selectionModel.parent[wp.selectionModel.selected]}
```

Parameters:

node
Identifiable or NavigationNode object to look up the children; it must not be null.

Returns: LayoutNode; the parent node for the node if there is a parent. Otherwise, it is null.

path(node)
Provides access to the path information for the node. The path represents the hierarchy from the root to the give node as a list. It is like a breadcrumb.

Example:

```
<c:forEach var="node" items="${wp.selectionModel.path[wp.selectionModel.selected]}">
 &lt;- ${node}
</c:forEach>
```

Parameters:

node
Identifiable or NavigationNode object to look up the children; it must not be null.

Returns: a list of LayoutNodes representing the path from the root to the node.

root Returns the root node of the layout model.

Example:

```
${wp.selectionModel.root}
```

Parameters: none

Returns: NavigationNode; it is never null.

selected

Returns the currently rendered page.

Example:

```
${wp.selectionModel.selected}
```

Parameters: none

Returns: NavigationNode.

NavigationNode:

Represents a navigation node in a navigation model.

Attributes:

contentNode

Returns the content node that is associated with the navigation node.

Example:

```
${wp.selectionModel.selected.contentNode}
${wp.navigationModel.selected.contentNode}
```

Parameters: none

Returns: ContentNode. It is never null.

description

The description of the navigation node.

Example:

```
${wp.selectionModel.selected.description}
${wp.navigationModel.selected.description}
```

Parameters: none

Returns: Description object for the navigation node; it is never null. You can use the value of the title object to retrieve the description in current locale.

isPrivate

Determines whether the current node is private.

Example:

```
${wp.selectionModel.selected.isPrivate}
${wp.navigationModel.selected.isPrivate}
```

Parameters: none

Returns: Boolean, true if the page is private, otherwise false.

metadata

The metadata map of this navigation node.

Example:

```
${wp.selectionModel.selected.metadata['com.ibm.portal.Hidden']}
${wp.navigationModel.selected.metadata['com.ibm.portal.Hidden']}
```

Parameters: none

Returns: Metadata, never null.

CF05 **moduleList**

Returns the module list for the currently selected page and theme.

Example:

```
<c:forEach var="node"items="${wp.selectionModel.selected.moduleList}">
    ${node.name}/${node.version},
</c:forEach>
```

```
<c:forEach var="node"items="${wp.navigationModel.selected.moduleList}">
    ${node.name}/${node.version},
</c:forEach>
```

Parameters: none

Returns: CurrentModuleList, never null.

objectID

Returns the ObjectID associated with this navigation node.

Example:

```
${wp.selectionModel.selected.objectID}
```

```
${wp.navigationModel.selected.objectID}
```

Parameters: none

Returns: ObjectID. Never null.

profileRef

Returns the profile reference for the page. If it is empty or null, the default theme profile reference is used.

Example:

```
${wp.selectionModel.selected.profileRef}
```

```
${wp.navigationModel.selected.profileRef}
```

Parameters: none

Returns: String, the profile reference within the theme. It can be null if you are using a non-modularized theme.

projectID

Returns the project identifier that is associated with this navigation node, or null of no project is associated.

Example:

```
${wp.selectionModel.selected.projectID}
```

```
${wp.navigationModel.selected.projectID}
```

Parameters: none

Returns: String representing the associated project. It can be null if no project is associated.

themeID

Returns the set theme ID for the page. If it is not set for the page, this function returns the inherited theme or default system theme.

Example:

```
${wp.selectionModel.selected.themeID}
```

```
${wp.navigationModel.selected.themeID}
```

Parameters: none

Returns: ObjectID of the referenced theme. Never null.

title The title of this navigation node.

Example:

```
${wp.selectionModel.selected.title}
```

```
${wp.navigationModel.selected.title}
```

Parameters: none

Returns: Title associated with the current object.

CF05 **url**

Short hand for urlGeneration that returns a string and cannot be manipulated any further.

Example:

```
${wp.selectionModel.selected.url}
${wp.navigationModel.selected.url}
```

Parameters: none

Returns: String; the URL pointing to this page.

CF05 **urlGeneration**

Creates a portal URL you can control with attributes. The URL attributes can be set by using further methods on the UrlGeneration object as shown in the examples section.

Example:

```
${wp.selectionModel.selected.urlGeneration}
${wp.navigationModel.selected.urlGeneration}
```

More examples:

```
<c:set var="node" value="${wp.selectionModel.selected}"/>
<a href="${node.url}">Simple URL, no modifications possible</a>
<a href="${node.urlGeneration}">Simple URL</a>
<a href="${node.urlGeneration.keepNavigationalState}">With NavState</a>
<a href="${node.urlGeneration.noNavigationalState}">Without NavState</a>
<a href="${node.urlGeneration.setThemeTemplate('Plain')}">With ThemeTemplate</a>
<a href="${node.urlGeneration.forcePublic}">Public Link</a>
<a href="${node.urlGeneration.secure}">Secure Link</a>
<a href="${node.urlGeneration.setLocale('de')}">In Deutsch</a>
<a href="${node.urlGeneration.setParam('a','b')}">With Params</a>
<a href="${node.urlGeneration.setParam('a','b').setParam('c','d').forcePublic.setLocale('de')
Complex URL</a>
<a href="${node.urlGeneration.logout}">Logout</a>
<a href="${node.urlGeneration.login}">Login</a>
<a href="${node.urlGeneration.normalize}">Normalized URL</a>
<a href="${node.urlGeneration.allowRelativeURL}">Relative URL</a>
<a href="${node.urlGeneration.disallowRelativeURL}">Disallow Relative URL</a>
<a href="${node.urlGeneration.forceAbsolute}">Absolute URL</a>
```

```
<c:set var="node" value="${wp.navigationModel.selected}"/>
<a href="${node.url}">Simple URL, no modifications possible</a>
<a href="${node.urlGeneration}">Simple URL</a>
<a href="${node.urlGeneration.keepNavigationalState}">With NavState</a>
<a href="${node.urlGeneration.noNavigationalState}">Without NavState</a>
<a href="${node.urlGeneration.setThemeTemplate('Plain')}">With ThemeTemplate</a>
<a href="${node.urlGeneration.forcePublic}">Public Link</a>
<a href="${node.urlGeneration.secure}">Secure Link</a>
<a href="${node.urlGeneration.setLocale('de')}">In Deutsch</a>
<a href="${node.urlGeneration.setParam('a','b')}">With Params</a>
<a href="${node.urlGeneration.setParam('a','b').setParam('c','d').forcePublic.setLocale('de')
Complex URL</a>
<a href="${node.urlGeneration.logout}">Logout</a>
<a href="${node.urlGeneration.login}">Login</a>
<a href="${node.urlGeneration.normalize}">Normalized URL</a>
<a href="${node.urlGeneration.allowRelativeURL}">Relative URL</a>
<a href="${node.urlGeneration.disallowRelativeURL}">Disallow Relative URL</a>
<a href="${node.urlGeneration.forceAbsolute}">Absolute URL</a>
```

Parameters: none

Returns: UrlGenerationPage; the URL object pointing to this page

ContentNode:

Provides access to a content node. This interface offers a way to obtain the type of the content node.

Attributes:

contentNodeType

Returns the type of this content node as `com.ibm.portal.content.ContentNodeType`.

Example:

```
${wp.navigationModel.selected.contentNode.contentNodeType}
${wp.selectionModel.selected.contentNode.contentNodeType}
```

The following example displays how to check whether the currently selected node is a label:

```
<c:if test="${wp.navigationModel.selected.contentNode.contentNodeType == 'LABEL'}">
<c:if test="${wp.selectionModel.selected.contentNode.contentNodeType == 'LABEL'}">
```

Parameters: none

Returns: `com.ibm.portal.content.ContentNodeType – COMPOSITION, EXTERNALURL, LABEL, PAGE, STATICPAGE`

description

The description of the content node.

Example:

```
${wp.navigationModel.selected.contentNode.description}
${wp.selectionModel.selected.contentNode.description}
```

Parameters: none

Returns: ContentNode object for the navigation node; it is never null. You can use the value of the title object to retrieve the description in current locale.

metadata

The metadata map of this content node.

Example:

```
${wp.navigationModel.selected.contentNode.metadata['com.ibm.portal.Hidden']}
${wp.selectionModel.selected.contentNode.metadata['com.ibm.portal.Hidden']}
```

Parameters: none

Returns: Metadata, never null.

CF05 **moduleList**

Returns the module list for the currently selected page and theme.

Example:

```
<c:forEach var="node" items="${wp.navigationModel.selected.contentNode.moduleList}">
 ${node.name}/${node.version},
</c:forEach>
<c:forEach var="node" items="${wp.selectionModel.selected.contentNode.moduleList}">
 ${node.name}/${node.version},
</c:forEach>
```

Parameters: none

Returns: CurrentModuleList, never null.

objectID

Returns the ObjectID associated with this content node.

Example:

```
${wp.navigationModel.selected.contentNode.objectID}
${wp.selectionModel.selected.contentNode.objectID}
```

Parameters: none

Returns: ObjectID. Never null.

profileRef

Returns the profile reference for the page. If it is empty or null, the default theme profile reference is used.

Example:

```
${wp.navigationModel.selected.contentNode.profileRef}
${wp.selectionModel.selected.contentNode.profileRef}
```

Parameters: none

Returns: String, which represents the reference to a profile that exists within the theme. It can be null if you are using a non-modularized theme.

themeID

Returns the set theme ID for the page. If it is not set for the page, this function returns the inherited theme or default system theme.

Example:

```
${wp.navigationModel.selected.contentNode.themeID}
${wp.selectionModel.selected.contentNode.themeID}
```

Parameters: none

Returns: ObjectID of the referenced theme. Never null.

title The title of this content node.

Example:

```
${wp.navigationModel.selected.contentNode.title}
${wp.selectionModel.selected.contentNode.title}
```

Parameters: none

Returns: Title associated with the current object.

CF05 **url**

Short hand for urlGeneration that returns a string and cannot be manipulated any further.

Example:

```
${wp.navigationModel.selected.url}
${wp.selectionModel.selected.url}
```

Parameters: none

Returns: String; the URL pointing to this page.

CF05 **urlGeneration**

Creates a portal URL you can control with attributes. The URL attributes can be set by using further methods on the UrlGeneration object as shown in the examples section.

Example:

```
${wp.navigationModel.selected.urlGeneration}
${wp.selectionModel.selected.urlGeneration}
```

More examples:

```
<c:set var="node" value="${wp.navigationModel.selected}"/>
<a href="${node.url}">Simple URL, no modifications possible</a>
<a href="${node.urlGeneration}">Simple URL</a>
<a href="${node.urlGeneration.keepNavigationalState}">With NavState</a>
<a href="${node.urlGeneration.noNavigationalState}">Without NavState</a>
<a href="${node.urlGeneration.setThemeTemplate('Plain')}">With ThemeTemplate</a>
<a href="${node.urlGeneration.forcePublic}">Public Link</a>
<a href="${node.urlGeneration.secure}">Secure Link</a>
<a href="${node.urlGeneration.setLocale('de')}">In Deutsch</a>
<a href="${node.urlGeneration.setParam('a','b')}">With Params</a>
<a href="${node.urlGeneration.setParam('a','b').setParam('c','d').forcePublic.setLocale('c
Complex URL</a>
<a href="${node.urlGeneration.logout}">Logout</a>
<a href="${node.urlGeneration.login}">Login</a>
<a href="${node.urlGeneration.normalize}">Normalized URL</a>
<a href="${node.urlGeneration.allowRelativeURL}">Relative URL</a>
<a href="${node.urlGeneration.disallowRelativeURL}">Disallow Relative URL</a>
<a href="${node.urlGeneration.forceAbsolute}">Absolute URL</a>
```

```
<c:set var="node" value="${wp.selectionModel.selected}"/>
<a href="${node.url}">Simple URL, no modifications possible</a>
<a href="${node.urlGeneration}">Simple URL</a>
<a href="${node.urlGeneration.keepNavigationalState}">With NavState</a>
<a href="${node.urlGeneration.noNavigationalState}">Without NavState</a>
<a href="${node.urlGeneration.setThemeTemplate('Plain')}">With ThemeTemplate</a>
<a href="${node.urlGeneration.forcePublic}">Public Link</a>
<a href="${node.urlGeneration.secure}">Secure Link</a>
<a href="${node.urlGeneration.setLocale('de')}">In Deutsch</a>
<a href="${node.urlGeneration.setParam('a','b')}">With Params</a>
<a href="${node.urlGeneration.setParam('a','b').setParam('c','d').forcePublic.setLocale('c
Complex URL</a>
<a href="${node.urlGeneration.logout}">Logout</a>
<a href="${node.urlGeneration.login}">Login</a>
<a href="${node.urlGeneration.normalize}">Normalized URL</a>
<a href="${node.urlGeneration.allowRelativeURL}">Relative URL</a>
<a href="${node.urlGeneration.disallowRelativeURL}">Disallow Relative URL</a>
<a href="${node.urlGeneration.forceAbsolute}">Absolute URL</a>
```

Parameters: none

Returns: UrlGenerationPage; the URL object pointing to this page

wp.themeConfig:

The wp.themeConfig encapsulates the theme **configuration** parameter lookup process.

Attributes:

get(key)

The following list shows the lookup order:

1. Theme metadata
2. The provided Resource Environment Provider.

Example:

```
<c:set var="themeModuleContextRoot" value="${wp.themeConfig['resources.modules.ibm.contextRo
```

Parameters:

key
String, the property key to look up.

Returns: String, the property value. It can be null.

wp.themeList:

wp.themeList provides access to list of theme objects.

Provides access to the model for a list of installed portal themes. Elements of this model are implemented by Theme objects.

Attributes:

current
Gets the current theme.

Example:

```
${wp.themeList.current}
```

Parameters: none

Returns: Theme, never null.

Exception: `current` requires a valid main portal request. If the required context is not included, an exception is thrown.

get(id) Get an individual theme.

Example:

```
${wp.themeList[id]}
```

Parameters:

id String or Identifiable object to look up the theme objects; it must not be null.

Returns: Theme; it can be null if not found.

iterator
Iterates through themes.

Example:

```
<c:forEach var="node" items="${wp.themeList}">
</c:forEach>
```

Parameters: none

Returns: An iterator with Theme objects; it is never null.

Theme:

Provides access to the attributes of one theme.

Attributes:

description
The description of the theme.

Example:

```
${wp.themeList.current.description}
```

Parameters: none

Returns: Description object for the theme node; it is never null. You can use the value of the title object to retrieve the description in current locale.

metadata
The metadata map of this theme.

Example:

```
${wp.themeList.current.metadata['com.ibm.portal.Hidden']}
```

Parameters: none

Returns: Metadata, never null.

profiles
Represents the profile list of this theme.

Example:

```
${wp.themeList.current.profiles}
```

Parameters: none

Returns: ProfileList, it is never null.

themeTemplateURI
Represents the URI pointing to the theme template that is configured for the current page that is rendered.

Example:

```
<%@ taglib uri="http://www.ibm.com/xmlns/prod/websphere/portal/v8.5/resolver" prefix="r" %
<r:dataSource uri='spa:${wp.identification[${wp.selectionModel.selected}]}' escape="none">
    <r:param name="themeURI" value="${wp.themeList.current.themeTemplateURI}"/>
    <r:param name="mime-type" value="text/html"/>
</r:dataSource>
```

Parameters: none

Returns: The URI pointing to the theme template that is configured for the current page that is rendered.

Exception: This attribute works for the current theme and current page that is rendered only.

title The title of this theme.

Example:

```
${wp.themeList.current.title}
```

Parameters: none

Returns: Title object for the theme node; it is never null. You can use the value of the title object to retrieve the title in current locale.

ProfileList:

Provides access to all profiles in a theme and the currently selected profile.

Attributes:

current
Gets the current profile.

Example:

```
${wp.themeList.current.profiles.current}
```

Parameters: none

Returns: Profile; it is never null.

Exception: `current` requires a valid main portal request. If the required context is not included, an exception is thrown.

iterator

Iterates through profiles.

Example:

```
<c:forEach var="profile" items="${wp.themeList.current.profiles}" varStatus="profileStatus">
     ${profile.title} / ${profile.relativePath}<br/>
</c:forEach>
```

Parameters: None

Returns: An iterator with Profile objects; it is never null.

Profile:

Provides access to the attributes of one profile.

Attributes:

description

The description of the profile.

Example:

```
${wp.themeList.current.profiles.current.description}
```

Parameters: none

Returns: Description object for the profile node; it is never null. You can use the value of the title object to retrieve the description in current locale.

metadata

The metadata map of this profile.

Example:

```
${wp.themeList.current.profiles.current.metadata['com.ibm.portal.Hidden']}
```

Parameters: none

Returns: Metadata, never null.

relativePath

The relative path of this profile.

Example:

```
${wp.themeList.current.profiles.current.relativePath}
```

Parameters: none

Returns: The URI of this profile as a string.

title The title of this profile.

Example:

```
${wp.themeList.current.profiles.current.title}
```

Parameters: none

Returns: Title object for the profile node; it is never null. You can use the value of the title object to retrieve the title in current locale.

uri The URI of this profile.

Example:

```
${wp.themeList.current.profiles.current.uri}
```

Parameters: none

Returns: The URI of this profile as a string.

wp.user - User:

The wp.user provides access to the active user.

Attributes:

get(profileAttribute)
Specifies the active profileAttribute

Example:

```
${wp.user['cn']}
```

Parameters:

profileAttribute
String; the attribute's name.

Returns: The value of the attribute from the user's profile.

objectID
Returns the object identifier for the current user.

Example:

```
${wp.identification[wp.user.objectID]}
```

Parameters: none

Returns: The object ID of the user.

Common beans:

The following beans are returned by many different beans and are used flexibly.

These beans are always returned.

"Title"
Provides access to the title of certain objects such as Navigation, Content, Theme, Profile, and others.

"Metadata" on page 2766
Provides access to the metadata of certain objects such as Navigation, Content, Theme, Profile, and others.

"Description" on page 2767
Provides access to the description of certain objects such as Navigation, Content, Theme, Profile, and others.

CF05 "UrlGeneration" on page 2768
Creates a portal URL you can control with attributes.

Title:

Provides access to the title of certain objects such as Navigation, Content, Theme, Profile, and others.

If no attribute is used, then the attribute `value` is used by default.

Attributes:

direction
Provides the direction that the title is read.

Example:
`${wp.title.direction}`

Parameters: none

Returns: String, either `ltr` or `rtl`; it is never null.

get(locale)
Gets the title in the specified locale.

Example:
`${wp.title['de']}`

Parameters:

`locale`
The locale code for the language you want.

Returns: String containing the title in the specified locale.

locale Provides the current locale.

Example:
`${wp.title.locale}`

Parameters: none

Returns: Locale object that represents the current locale.

value Provides the title in the current locale.

Example:
`${wp.title.value}`

Parameters: none

Returns: String containing the title in the current locale.

xmlLocale
Provides the current locale as defined by IETF BCP 47.

Example:
`${wp.title.xmlLocale}`

Parameters: none

Returns: String containing the current local in XML format. For example, `en-us`.

Metadata:

Provides access to the metadata of certain objects such as Navigation, Content, Theme, Profile, and others.

Attributes:

get(key)
Gets the metadata for the metadata *key*

Example:
`${wp.themeList.current.metadata['com.ibm.portal.Hidden']}`

Returns: String value of the metadata; it can be null if not found.

Description:

Provides access to the description of certain objects such as Navigation, Content, Theme, Profile, and others.

If no attribute is used, then the attribute `value` is used by default.

Attributes:

direction
Provides the direction that the description is read.

Example:
`${wp.description.direction}`

Parameters: none

Returns: String, either `ltr` or `rtl`; it is never null.

get(locale)
Gets the description in a specified locale.

Example:
`${wp.description['de']}`

Parameters:

`locale`
The locale code for the language you want.

Returns: String containing the description in the specified locale.

locale Provides the current locale.

Example:
`${wp.description.locale}`

Parameters: none

Returns: Locale object that represents the current locale.

value Provides the description in the current locale.

Example:
`${wp.description.value}`

Parameters: none

Returns: String containing the description in the current locale.

xmlLocale
Provides the current locale as defined by IETF BCP 47.

Example:
`${wp.description.xmlLocale}`

Parameters: none

Returns: String containing the current local in XML format. For example, `en-us`.

UrlGeneration:

Creates a portal URL you can control with attributes.

Attributes:

keepNavigationalState
The navigational state is included into the URL.

Example:

```
${wp.selectionModel.selected.urlGeneration.keepNavigationalState}
```

Parameters: none

Returns: UrlGeneration, which is the same object, so multiple method calls can be concatenated.

noNavigationalState
The current navigational state that includes all portlet modes, states, and render parameters is not included in the URL, and the portal is reset to its default state.

Example:

```
${wp.selectionModel.selected.urlGeneration.noNavigationalState}
```

Parameters: none

Returns: UrlGeneration, which is the same object, so multiple method calls can be concatenated.

setParam(name, value)
Use this attribute to add parameters to the URL. Parameters added to the URLGeneration object occur as unscoped query parameters. Parameter handling depends on the target of the URL. If the URL points to a page, the parameters are visible to all IBM portlets on that page. Parameters are not visible to standard portlets if the URL does not point specifically to that portlet.

Example:

```
${wp.selectionModel.selected.urlGeneration.setParam('a','b')}
```

Parameters:

`name`
String; the name of the parameter.

`value`
String; the value of the parameter.

Returns: UrlGeneration, which is the same object, so multiple method calls can be concatenated.

setLocale(locale)
Use this attribute to change the active language in the navigational state in which the URL is generated.

Example:

```
${wp.selectionModel.selected.urlGeneration.setLocale('de')}
```

Parameters:

`locale`
String; the locale to be set.

Returns: UrlGeneration, which is the same object, so multiple method calls can be concatenated.

clearThemeTemplate
Removes the theme template from the navigational state.

Example:

```
${wp.selectionModel.selected.urlGeneration.clearThemeTemplate}
```

Parameters: none

Returns: UrlGeneration, which is the same object, so multiple method calls can be concatenated.

setThemeTemplate(template)
This specifies the theme template that is taken for rendering the requested page. Can be referenced as either a JSP or a class and is used as theme for this URL.

Example:

```
${wp.selectionModel.selected.urlGeneration.setThemeTemplate('Plain')}
```

Parameters:

template
String; the template name.

Returns: UrlGeneration, which is the same object, so multiple method calls can be concatenated.

secure This attribute creates a secure URL (HTTPS).

Example:

```
${wp.selectionModel.selected.urlGeneration.secure}
```

Parameters: none

Returns: UrlGeneration, which is the same object, so multiple method calls can be concatenated.

unsecure
This attribute creates an unsecure URL (HTTP).

Example:

```
${wp.selectionModel.selected.urlGeneration.unsecure}
```

Parameters: none

Returns: UrlGeneration, which is the same object, so multiple method calls can be concatenated.

forceProtected
This attribute creates a URL pointing to the protected (logged in) page of the portal.

Example:

```
${wp.selectionModel.selected.urlGeneration.forceProtected}
```

Parameters: none

Returns: UrlGeneration, which is the same object, so multiple method calls can be concatenated.

forcePublic
This attribute creates a URL pointing to the public (anonymous) page of the portal.

Example:

```
${wp.selectionModel.selected.urlGeneration.forcePublic}
```

Parameters: none

Returns: UrlGeneration, which is the same object, so multiple method calls can be concatenated.

forceAbsolute
If you use this attribute, absolute URLs are enforced; in this case other settings that affect the generation of URLs might be overridden.

Example:

```
${wp.selectionModel.selected.urlGeneration.forceAbsolute}
```

Parameters: none

Returns: UrlGeneration, which is the same object, so multiple method calls can be concatenated.

allowRelativeURL
This attribute indicates that a relative URL is generated. If not called for this `Urlgeneration`, then default is used, which is set by the property com.ibm.portal.state.accessors.url.URLContext.enableRelative in the StateManagerService.

Example:

```
${wp.selectionModel.selected.urlGeneration.allowRelativeURL}
```

Parameters: none

Returns: UrlGeneration, which is the same object, so multiple method calls can be concatenated.

normalize
This attribute indicates that the URL to be generated should be normalized. If more parameters are set, the normalization is run first and afterward the other state modifications are accomplished. It normalizes the URL with the same XSL file used to normalize URLs for search engines. The normalized representation of the URL can also be used to bookmark a page.

Example:

```
${wp.selectionModel.selected.urlGeneration.normalize}
```

Parameters: none

Returns: UrlGeneration, which is the same object, so multiple method calls can be concatenated.

logout Creates a URL that logs out the current user.

Example:

```
${wp.selectionModel.selected.urlGeneration.logout}
```

Parameters: none

Returns: UrlGeneration, which is the same object, so multiple method calls can be concatenated.

login Creates a URL that logs in a user to portal. This attribute is typically used for the login panel. The **`user ID`** and **`password`** parameters must be carried with a login URL.

Example:

```
${wp.selectionModel.selected.urlGeneration.login}
```

Parameters: none

Returns: UrlGeneration, which is the same object, so multiple method calls can be concatenated.

setPageEditMode(flag)
You can set the page edit mode as used by the toolbar.

Example:

```
${wp.selectionModel.selected.urlGeneration.setPageEditMode('true')}
```

Parameters:

flag
String that indicates a Boolean value as a string. The value `true` enables the page edit mode. The value `false` disables it.

Returns: UrlGeneration, which is the same object, so multiple method calls can be concatenated.

url Generates the url with all its attributes and returns it as a string. This is the default method that is called when no further attributes are specified.

Example:

```
${wp.selectionModel.selected.urlGeneration.url}
${wp.selectionModel.selected.urlGeneration}
```

Parameters: none

Returns: String; the generated URL as a string.

UrlGenerationPage:

Extends UrlGeneration on a page. All the UrlGeneration attributes are available in addition to these attributes.

Attributes:

setLayoutNode(node)
This attribute indicates the ID or unique name of the control that holds the portlet. The value wp.currentSelectedPortlet can be used inside a control when you generate a URL to the portlet within that control.

Example:

```
${wp.navigationModel['ibm.portal.Search Center'].urlGeneration.setLayoutNode('ibm.portal.S
```

Parameters:

node
String or Identifiable; defines the control to be used as a target for the URL.

Returns: UrlGenerationPortlet; a URL generation object for portlet URLs.

setPortlet(portlet)
This attribute indicates the ID or unique name of the control that holds the portlet. The value wp.currentSelectedPortlet can be used inside a control when you generate a URL to the portlet within that control.

Example:

```
${wp.navigationModel['ibm.portal.Search Center'].urlGeneration.setPortlet('ibm.portal.Sear
```

Parameters:

node
String or Identifiable; defines the control to be used as a target for the URL.

Returns: UrlGenerationPortlet; a URL generation object for portlet URLs.

autoSelectPortlet
This attribute automatically selects the first portlet on the page and addresses this as the target for the portlet URL.

Example:

```
${wp.navigationModel['wps.Selfcare'].urlGeneration.autoSelectPortlet.normal.setRenderParam('
```

Parameters: none

Returns: UrlGenerationPortlet; a URL generation object for portlet URLs.

UrlGenerationPortlet:

Extends UrlGeneration in a portlet. All the UrlGeneration and UrlGenerationPage attributes are available in addition to these attributes.

Attributes:

view This attribute sets the portlet mode VIEW.

Example:

```
${wp.navigationModel['wps.Selfcare'].urlGeneration.autoSelectPortlet.view}
```

Parameters: none

Returns: UrlGenerationPortlet, which is the same object, so multiple method calls can be concatenated.

help This attribute sets the portlet mode HELP.

Example:

```
${wp.navigationModel['wps.Selfcare'].urlGeneration.autoSelectPortlet.help}
```

Parameters: none

Returns: UrlGenerationPortlet, which is the same object, so multiple method calls can be concatenated.

edit This attribute sets the portlet mode EDIT.

Example:

```
${wp.navigationModel['wps.Selfcare'].urlGeneration.autoSelectPortlet.edit}
```

Parameters: none

Returns: UrlGenerationPortlet, which is the same object, so multiple method calls can be concatenated.

editDefaults
This attribute sets the portlet mode EDIT_DEFAULTS.

Example:

```
${wp.navigationModel['wps.Selfcare'].urlGeneration.autoSelectPortlet.editDefaults}
```

Parameters: none

Returns: UrlGenerationPortlet, which is the same object, so multiple method calls can be concatenated.

configure
This attribute sets the portlet mode CONFIGURE.

Example:

```
${wp.navigationModel['wps.Selfcare'].urlGeneration.autoSelectPortlet.configure}
```

Parameters: none

Returns: UrlGenerationPortlet, which is the same object, so multiple method calls can be concatenated.

render This attribute generates URLs to a standard portlet's render processing method. If this attribute is omitted, the render method of the portlet is called.

Example:

```
${wp.navigationModel['wps.Selfcare'].urlGeneration.autoSelectPortlet.render}
```

Parameters: none

Returns: UrlGenerationPortlet, which is the same object, so multiple method calls can be concatenated.

action This attribute generates URLs to a standard portlet's action processing method. If this attribute is omitted, the render method of the portlet is called.

Example:

```
${wp.navigationModel['wps.Selfcare'].urlGeneration.autoSelectPortlet.action}
```

Parameters: none

Returns: UrlGenerationPortlet, which is the same object, so multiple method calls can be concatenated.

resource
This attribute generates URLs to a standard portlet's resource processing method. If this attribute is omitted, the render method of the portlet is called.

Example:

```
${wp.navigationModel['wps.Selfcare'].urlGeneration.autoSelectPortlet.resource}
```

Parameters: none

Returns: UrlGenerationPortlet, which is the same object, so multiple method calls can be concatenated.

maximized
This attribute sets the window state MAXIMIZED.

Example:

```
${wp.navigationModel['wps.Selfcare'].urlGeneration.autoSelectPortlet.maximized}
```

Parameters: none

Returns: UrlGenerationPortlet, which is the same object, so multiple method calls can be concatenated.

minimized
This attribute sets the window state MINIMIZED.

Example:

```
${wp.navigationModel['wps.Selfcare'].urlGeneration.autoSelectPortlet.minimized}
```

Parameters: none

Returns: UrlGenerationPortlet, which is the same object, so multiple method calls can be concatenated.

normal

This attribute sets the window state NORMAL.

Example:

```
${wp.navigationModel['wps.Selfcare'].urlGeneration.autoSelectPortlet.normal}
```

Parameters: none

Returns: UrlGenerationPortlet, which is the same object, so multiple method calls can be concatenated.

solo This attribute sets the window state SOLO.

Note: The portlet must be able to exist in solo state that uses the `createReturnURI()` method. If a portlet without this method is placed in solo state, then users must log out or close their browser windows to return to the portal.

Example:

```
${wp.navigationModel['wps.Selfcare'].urlGeneration.autoSelectPortlet.solo}
```

Parameters: none

Returns: UrlGenerationPortlet, which is the same object, so multiple method calls can be concatenated.

setRenderParam

Use this attribute to add render parameters to the URL. Parameters are not visible to standard portlets if the URL does not point specifically to that portlet.

Example:

```
${wp.navigationModel['wps.Selfcare'].urlGeneration.autoSelectPortlet.setRenderParam('a','b')
```

Parameters:

`name`
String; the name of the parameter.

`value`
String; the value of the parameter.

Returns: UrlGenerationPortlet, which is the same object, so multiple method calls can be concatenated.

setActionParam

Use this attribute to add action parameters to the URL. Parameters are not visible to standard portlets if the URL does not point specifically to that portlet.

Example:

```
${wp.navigationModel['wps.Selfcare'].urlGeneration.autoSelectPortlet.action.setActionParam('
```

Parameters:

`name`
String; the name of the parameter.

`value`
String; the value of the parameter.

Returns: UrlGenerationPortlet, which is the same object, so multiple method calls can be concatenated.

uiNavigationModel:

By default, the uiNavigationModel lists only visible pages as part of its iterator. When the **Show Hidden Pages** option is selected in the Toolbar, it also lists the hidden pages. There is a mobile hidden flag for pages. The model also helps you specify a mobile test device class expression, which is used to evaluate if the system is rendered as part of a mobile request.

Attributes:

children(node)
Returns an iterator of child nodes.

Example:

```
<c:forEach var="node" items="${uiNavigationModel.children[wp.selectionModel.selected]}" >
    <c:out value="${node}" /><br>
</c:forEach>
```

Parameters:

node
Identifiable, or uiNavigationNode object to look up the children; it must not be null.

Returns: An iterator with uiNavigationNode objects; it is never null.

get(id) Get an individual navigation node.

Example:

```
${uiNavigationModel[id]}
```

Parameters:

id String or Identifiable object to look up the navigation object; it must not be null.

Returns: uiNavigationNode; it can be null if not found.

hasChildren
Determines whether the specified uiNavigationNode has associated nodes.

Example:

```
${uiNavigationModel.hasChildren[wp.selectionModel.selected]}
```

Parameters:

node
Identifiable, or uiNavigationNode object to look up the children; it must not be null.

Returns: Boolean; true if the node has children. Otherwise, it is false.

parent Access to the parent of a uiNavigationNode.

Example:

```
${uiNavigationModel[wp.selectionModel.selected].parent}
```

Parameters:

node
Identifiable or uiNavigationNode object to look up the children; it must not be null.

Returns: uiNavigationNode; the parent node for the node if there is a parent. Otherwise, it is null.

path(node)
Provides access to the path information for the node. The path represents the hierarchy from the root to the give node as a list. It is like a breadcrumb.

Example:

```
<c:forEach var="node" items="${uiNavigationModel.path[wp.selectionModel.selected]}">
 &lt;- ${node}
</c:forEach>
```

Parameters:

node
Identifiable or uiNavigationNode object to look up the children; it must not be null.

Returns: a list of uiNavigationNodes representing the path from the root to the node.

root Returns the root node of the uiNavigation model.

Example:

```
${uiNavigationModel.root}
```

Parameters: none

Returns: uiNavigationNode; it is never null.

uiNavigationNode:

Provides access to a navigation node in a navigation model.

To access the uiNavigationModel Bean, you must use the uiNavigationModel tag and define the variable name that you want the bean to be available under. For example:

```
<portal-navigation:uiNavigationModel var="uiNavigationModel"
    mobileDeviceClassTest="smartphone/tablet">
uiNavigationNode_attribute</portal-navigation:uiNavigationModel>
```

Attributes:

contentNode
Returns the content node that is associated with the navigation node.

Example:

```
${uiNavigationModel.selected.contentNode}
```

Parameters: none

Returns: ContentNode. Never null.

description
The description of the navigation node.

Example:

```
${uiNavigationModel.selected.description}
```

Parameters: none

Returns: Description object for the navigation node; it is never null. You can use the value of the title object to retrieve the description in current locale.

isDraft

Returns whether the navigation node is part of a project or not.

Example:

```
${uiNavigationModel[wp.selectionModel.selected].isDraft}
```

Parameters: none

Returns: Boolean; true if the page is a draft page. False if it is not.

isHidden

Returns whether the navigation node is hidden or not.

Example:

```
${uiNavigationModel[wp.selectionModel.selected].isHidden}
```

Parameters: none

Returns: Boolean; true if the page is hidden. False if it is not.

isInSelectionPath

Returns whether the navigation node is part of the selection path. It returns whether a page is in the breadcrumb trail, but not necessarily the currently active page.

Example:

```
${uiNavigationModel[wp.selectionModel.selected].isInSelectionPath}
```

Parameters: none

Returns: Boolean; true if the page is part of the selection path. False if it is not.

isPrivate

Determines if the current node is private.

Example:

```
${uiNavigationModel.selected.isPrivate}
```

Parameters: none

Returns: Boolean, true if the page is private, otherwise false.

isSelected

Returns whether the navigation node is selected or the currently selected page.

Example:

```
${uiNavigationModel[wp.selectionModel.selected].isSelected}
```

Parameters: none

Returns: Boolean; true if the page is selected. False if it is not.

metadata

The metadata map of this navigation node.

Example:

```
${uiNavigationModel.selected.metadata['com.ibm.portal.Hidden']}
```

Parameters: none

Returns: Metadata, never null.

moduleList

Returns the module list for the currently selected page and theme.

Example:

```
<c:forEach var="node"items="${uiNavigationModel.selected.moduleList}">
    ${node.name}/${node.version},
</c:forEach>
```

Parameters: none

Returns: CurrentModuleList, never null.

objectID

Returns the ObjectID associated with this navigation node.

Example:

```
${uiNavigationModel.selected.objectID}
```

Parameters: none

Returns: ObjectID. Never null.

profileRef

Returns the profile reference for the page. If it is empty or null, the default theme profile reference is used.

Example:

```
${wp.selectionModel.selected.profileRef}
```

Parameters: none

Returns: String, the profile reference within the theme. It can be null if you are using a non-modularized theme.

projectID

Returns the project identifier that is associated with this navigation node, or null of no project is associated.

Example:

```
${uiNavigationModel.selected.projectID}
```

Parameters: none

Returns: String representing the associated project. It can be null if no project is associated.

title The title of this navigation node.

Example:

```
${uiNavigationModel.selected.title}
```

Parameters: none

Returns: Title associated with the current object.

themeID

Returns the set theme ID for the page. If it is not set for the page, this function returns the inherited theme or default system theme.

Example:

```
${wp.selectionModel.selected.themeID}
```

Parameters: none

Returns: ObjectID of the referenced theme. Never null.

url Short hand for urlGeneration that returns a string and cannot be manipulated any further.

Example:

```
${wp.selectionModel.selected.url}
```

Parameters: none

Returns: String; the URL pointing to this page.

urlGeneration

Creates a portal URL you can control with attributes. The URL attributes can be set by using further methods on the UrlGeneration object as shown in the examples section.

Example:

```
${wp.selectionModel.selected.urlGeneration}
```

More examples:

```
<c:set var="node" value="${wp.selectionModel.selected}"/>
<a href="${node.url}">Simple URL, no modifications possible</a>
<a href="${node.urlGeneration}">Simple URL</a>
<a href="${node.urlGeneration.keepNavigationalState}">With NavState</a>
<a href="${node.urlGeneration.noNavigationalState}">Without NavState</a>
<a href="${node.urlGeneration.setThemeTemplate('Plain')}">With ThemeTemplate</a>
<a href="${node.urlGeneration.forcePublic}">Public Link</a>
<a href="${node.urlGeneration.secure}">Secure Link</a>
<a href="${node.urlGeneration.setLocale('de')}">In Deutsch</a>
<a href="${node.urlGeneration.setParam('a','b')}">With Params</a>
<a href="${node.urlGeneration.setParam('a','b').setParam('c','d').forcePublic.setLocale('c
Complex URL</a>
<a href="${node.urlGeneration.logout}">Logout</a>
<a href="${node.urlGeneration.login}">Login</a>
<a href="${node.urlGeneration.normalize}">Normalized URL</a>
<a href="${node.urlGeneration.allowRelativeURL}">Relative URL</a>
<a href="${node.urlGeneration.disallowRelativeURL}">Disallow Relative URL</a>
<a href="${node.urlGeneration.forceAbsolute}">Absolute URL</a>
```

Parameters: none

Returns: UrlGenerationPage; the URL object pointing to this page

Drag-and-drop

Drag-and-drop uses HTML5 code to facilitate the layout of content.

Drag sources and drop targets do not need any JavaScript library. Some convenience functions are provided in the i$.dnd API. For example, with drag-and-drop, you can drag an item, such as a portlet or content item from the toolbar on the page. Or you can drag an item from one position on the page to another position.

"Customizing drop targets on the page" on page 2780
In edit mode, portal renders drop targets for all positions where a portlet can be added to the page. In the default theme, these drop targets are hidden unless an item is dragged over the surrounding container.

"Custom drag sources and drop targets" on page 2780
Drag sources and drop targets must be created according to the HTML 5 specification. In HTML 5, the **DataTransfer** object is used to exchange data from source to target.

Customizing drop targets on the page:

In edit mode, portal renders drop targets for all positions where a portlet can be added to the page. In the default theme, these drop targets are hidden unless an item is dragged over the surrounding container.

The markup for drop targets is defined in a file that is named `DropTarget.html`, which lies in the current skin folder in WebDAV. For example, `fs-type1/themes/Portal8.5/skins/Hidden/DropTarget.html`

The markup in `DropTarget.html` is always surrounded by a `<div>` tag, representing the drop target control component. Inside the `<div>` tag is an anchor tag that references the control-component at the position where a dropped item is inserted. For example:

```
<div class="portlet-window component-control portal-drop-target id-ID_of_control_component_at_insert_
<a class="portlet-window-ref" href="#ID_of_control_component_at_insert_position"></a>

        <DropTarget.html>

</div>
```

The styling for the CSS classes can be found in the `default_edit.css` in `fs-type1/themes/Portal8.5/css/default/default_edit.css`.

Portal also injects the following CSS classes during a drag operation:

- When an item is dragged over a column or row container, the container gets the CSS class `ibmDndDropZonesActive`.
- When an item is dragged over a drop target, the drop target gets the CSS class `ibmDndDropZoneOver`.

The following list shows CSS selector examples and the resulting action:

- `div.ibmDndDropZonesActive div.portal-drop-target` selects a drop target inside the currently active container, either a row or a column.
- `div.ibmDndDropZonesActive div.portal-drop-target.ibmDndDropZoneOver` selects a drop target where an item is currently dragged over.

A theme developer can customize the drop zones either by editing the `default_edit.css` or by modifying the `DropTarget.html`, or both.

Firefox currently does not support automatic scrolling during a drag-and-drop operation. There are plug-ins available to add this support to Firefox.

Custom drag sources and drop targets:

Drag sources and drop targets must be created according to the HTML 5 specification. In HTML 5, the **`DataTransfer`** object is used to exchange data from source to target.

The source sets the data with `dataTransfer.setData(format, data)` and the target reads the data with `dataTransfer.getData(format)`. WebSphere Portal Express always uses `format = "Text"`. The data is a stringified JSON object with the following structure:

```
{ "uri" : URI }
```

In the previous example, the *URI* is the URI of the object that is dragged. If more than one object is dragged, the structure looks like this example.

```
{ "uri" : [ URI_1, URI_2, ... ] }
```

Each URI can have a list of parameters. In this case, the JSON object has the following structure.

```
{ "uri" : [ { key_11 : value_11, key_12 : value_12, ... }, URI_1, { key_21 : value_21, key_22 : va
```

In the previous example, **key_1x:value_1x** are the parameters of *URI_1* and **key_2x:value_2x** are the parameters of *URI_2*.

The theme module `i$.dnd` provides two convenience functions that simplify the creation of HTML 5 drag sources and drop targets. To create a drag source, you can use the following code.

```
i$.dnd.addSource(parameter)
```

In the previous example, the parameter object needs to have the following elements:

{DOMNode} parameter.node
The DOM node that acts as the drag source.

{String} parameter.type
The type of data that is transferred, typically `"Text"`.

{Object} parameter.data
The data to transfer when dropped. When you exchang data with a standard WebSphere Portal Express drop target, it must be a stringified JSON object with the structure described before.

{DOMNode} parameter.avatar
The DOM node to use as the DnD avatar. (optional)

{Function} parameter.dragstart
A function to start when **dragstart** is called. (optional) The following parameters are passed to this function:

{Event} e
The event. For example, you can use the event to set `e.dataTransfer.effectAllowed = "copy";`.

{DOMNode} n
The **DomNode** of the source area.

{Function} parameter.dragend
A function to start when **dragend** is called. It has the same parameters as **parameter.dragstart**. (optional)

To create a drop target, you can use the following code.

```
i$.dnd.addTarget(parameter)
```

In the previous example, the parameter object must have the following elements:

{DOMNode} parameter.node
The **DOMnode** acting as the drop target.

{String} parameter.type
The type of data that is transferred, typically `"Text"`.

{Function} parameter.drop
Function to handle the drop event. The following parameters are passed to this function:

{Event} e
The drop event.

{DOMNode} n
The DOM node of the drop target.

{String} type
The type of data that is transferred.

{Object} data
The data that was transferred from the drag source. When receiving data from standard WebSphere Portal Express drag sources, it is a stringified JSON object with the structure described before.

{Function} parameter.dragenter
Function to start when **dragenter** is called. (optional) The following parameters are passed to this function:

{Event} e
The event.

{DOMNode} n
The DOM node of the drop target.

{Function} parameter.dragleave
Function to start when **dragend** is called with the same parameters as **dragenter**. (optional)

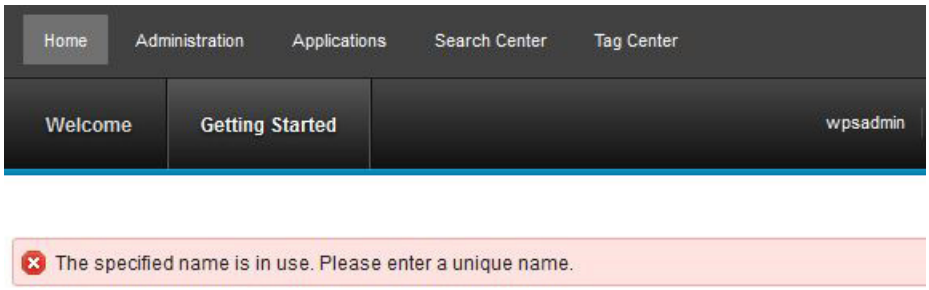
{Function} parameter.dragover
Function to start when **dragover** is called with the same parameters as **dragenter**. (optional)

Displaying messages in the status bar module

Displaying messages in the status bar module.

About this task

All ready-to-use theme profiles include the status_bar module, which provides an area beneath the header for displaying error, warning and information messages. The following image shows an example error message in red.



Procedure

1. Add a line of JavaScript to Use this status bar in your themes, modules, and portlets can use this status bar. The following example shows the StatusMessage JavaScript code:

```
i$.fireEvent("/portal/status",[{message: new com.ibm.widgets.StatusMessage
("error", "The specified name is in use. Please enter a unique name.", ""), uid: 'ibmStatusBox'}]);
```

`i$.fireEvent` is the recommended syntax. However, if you use the Dojo module, you can also use this alternative syntax:

```
dojo.publish("/portal/status",[{message: new com.ibm.widgets.StatusMessage
("error", "The specified name is in use. Please enter a unique name.", ""), uid: 'ibmStatusBox'}]);
```

a. The first argument in the StatusMessage can be either "error", "warning" or "info" to control the icon and color of the message box.
b. The second argument in the StatusMessage is a String representing the message itself to be displayed in the message box.
c. The third argument in the StatusMessage is an optional String representing any further details about the message. The details are not presented to the user by default, rather only if the user clicks the **Show Details** icon to expand the message box.

2. If you have an error that originates on the server, you can set a ibm.portal.operations.error cookie in the response, and the client shows the message when the response completes. Use parsable JSON code for the cookie string value, such as:

```
ibm.portal.operations.error={"errorType":"error","errorMessage":"The specified name is in use. Please enter a unique name.","errorDetails":""}
```

3. Messages are cleared automatically on page reload. If you must clear one or more messages from the client by using JavaScript, then you can use the following code:

   ```
   i$.fireEvent("/portal/status/clear", [{message: mymessage, uid: 'ibmStatusBox'}]);
   ```

   where *mymessage* is the StatusMessage instance you saved when displaying the message. Code like the following example is fine too:

   ```
   i$.fireEvent("/portal/status/clear", [{message: new com.ibm.widgets.StatusMessage ("error", "T
   ```

   The event clears any messages whose StatusMessage type, message, and details strings match, so you do not have to reuse the same actual StatusMessage instance.

   If no StatusMessage is passed, then it clears all messages. You can clear everything with the following code sample:

   ```
   i$.fireEvent("/portal/status/clear", [{uid: 'ibmStatusBox'}]);
   ```

Updating the content menu to open on click instead of on hover

The content menu opens when a user hovers over a portlet containing IBM Web Content Manager items. In combined Cumulative Fix 05, you can set the content menu to open when a user clicks an icon in the portlet skin instead.

About this task

Procedure

1. Add the content menu icon to your custom skin. Update the custom theme CSS sprite to include the new content menu icon image. Replace the master.png static resource with the Portal 8.5 default theme image in WebDAV at `/themes/Portal8.5/css/images/master.png`.
2. Add the HTML for the icon to the custom skins. Add the following code to the custom skin templates after the `<h2>` node.

   ```
   <a aria-haspopup="true" aria-label="Display component action menu" role="button" href="javascr
                   <span title="Display component action menu"><img aria-label="Display component
                   <span class="wpthemeAltText">Component Action Menu</span>
                   <!-- start CAM template -->
                   <span class="wpthemeMenu" data-positioning-handler="horizontallyCenteredBelow"
                       <div class="wpthemeMenuBorder">
                           <!-- define the menu item template inside the "ul" element.  only "css
                           <ul class="wpthemeMenuDropDown wpthemeTemplateMenu" role="menu">
   ```

```
                        <li class="${css-class}" role="menuitem" tabindex="-1"><span
                            class="wpthemeMenuText">${title}</span></li>
                    </ul>
                    <div class="verticalMenuPointer pointer"></div>
                </div> <!-- Template for loading -->
                <div class="wpthemeMenuLoading wpthemeTemplateLoading">${loading}</div>
                <!-- Template for submenu -->
                <div class="wpthemeAnchorSubmenu wpthemeTemplateSubmenu">
                    <div class="wpthemeMenuBorder wpthemeMenuSubmenu">
                        <ul id="${submenu-id}" class="wpthemeMenuDropDown"
                            role="menu">
                            <li role="menuitem" tabindex="-1"></li>
                        </ul>
                    </div>
                </div>
            </span>
            <!-- end CAM template -->
        </a>
```

3. Translate the values of the `aria-label` and `wpthemeAltText` to support any languages you support.
4. Add the CSS for the icon. Add the following code anywhere in the custom theme.

```
.wpthemeControlHeader a.wpthemeIcon.contextMenuInSkinIcon img {
        background-position:0 -1496px;
    }
```

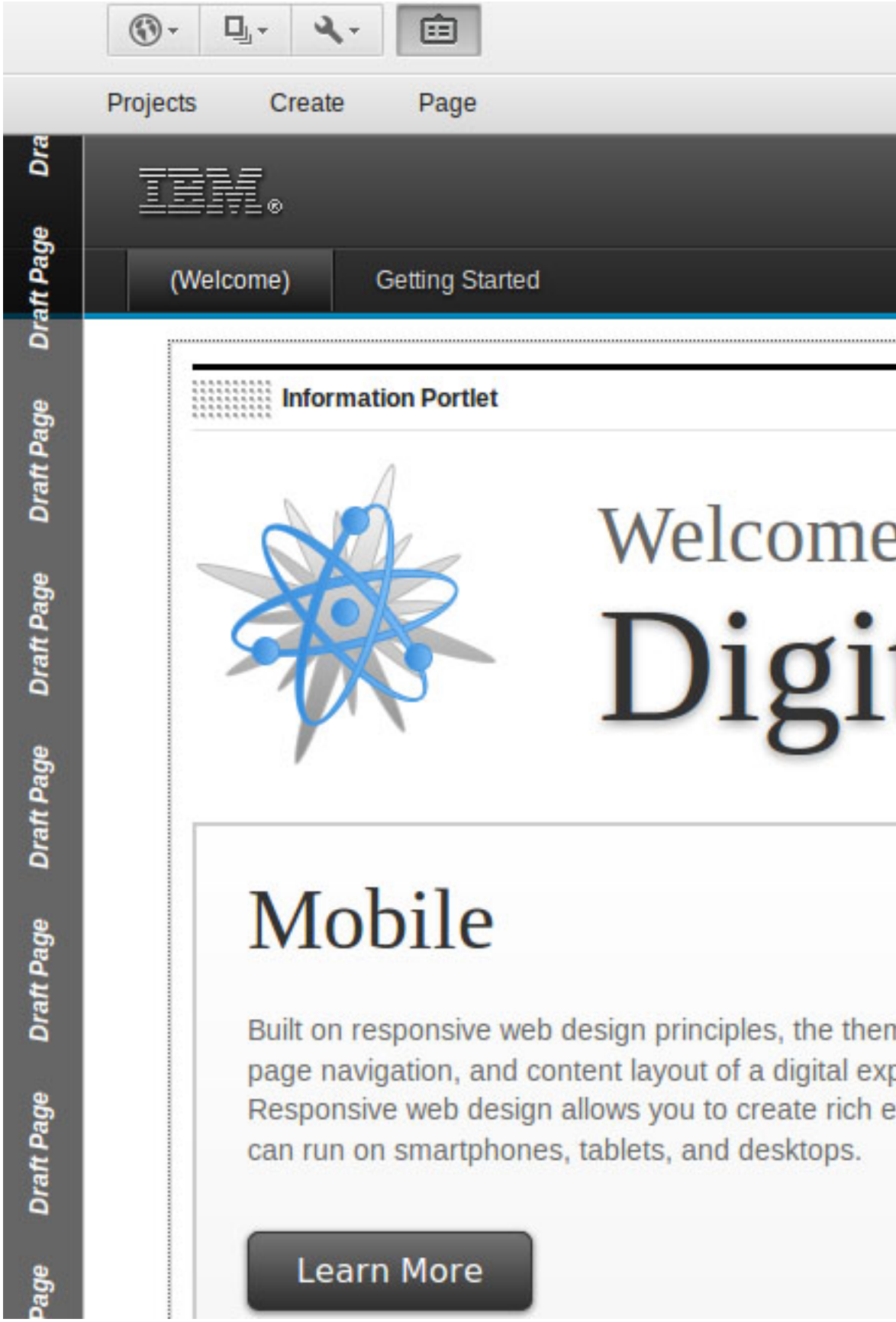
5. Add the new content menu to the custom theme profile. Open the theme profile and add `wp_skin_cam` to the `deferredModuleIDs` section.

Displaying the draft page ribbon

The draft page ribbon is a strip of text that appears along both sides of a page that has a draft in the current project.

About this task

The draft page ribbon makes it easier for users to see if a page has pending changes waiting to be published.


Projects
Create
Page
Draft Page
IBM
(Welcome)
Getting Started
Information Portlet
Welcome
Digi
Mobile
Built on responsive web design principles, the them
page navigation, and content layout of a digital exp
Responsive web design allows you to create rich e
can run on smartphones, tablets, and desktops.
Learn More

Procedure

1. To get the draft page ribbon feature, add the `wp_draft_page_ribbon` module to your theme profile.
2. To add and remove ready-to-use modules to your theme, see Adding or removing a module from a profile.

Related concepts:

"Modules that add features to a theme" on page 2652
Theme modules contribute resources, such as JavaScript, CSS and HTML, to a theme or page. The following modules come with the modularized theme and specifically contribute a feature rather than working in the background. The modules can be added to a theme by listing them in the theme's default profile, or to a single page or set of pages.

Adding jQuery to a theme

The jQuery library is a JavaScript library. IBM WebSphere Portal Express includes a jQuery module for the jQuery core so you can use jQuery in a theme.

About this task

This jQuery module is not turned on by default. jQuery is an open framework and is not supported by IBM. The module makes it quick and easy to use jQuery and any of its plug-ins in a theme.

Procedure

1. To turn on jQuery, copy the `jquery1.10.2.json` module definition file into your theme's contributions folder in WebDAV `fs-type1`. This `jquery1.10.2.json` file can be found at *`PortalServer`*`\theme\wp.theme.jquery\installedApps\wp.theme.jquery.ear\wp.theme.jquery.war\v1.10.2`.
2. Optional: To enable plug-ins with the jQuery core, download the jQuery plug-in from the jQuery website For more information, see *jQuery plug-in download* in the related links. In this example, use the PowerTip plug-in. The `jquery.powertip-1.2.0.zip` file. To download the plug-in, see *jQuery PowerTip plug-in download* in the related links.
   a. Extract the contents of your plug-in file to any location.
   b. Create a subfolder in your theme's modules folder for the plug-in, such as `jquery_powertip` in this example.
   c. The PowerTip plug-in has two files that must be head contributions of your module, so create a head subfolder in your `jquery_powertip` folder.
   d. Copy the needed resource files from your extracted plug-in location into your head folder. The needed resource files in this case are:

      ```
      jquery.powertip.min.js
      jquery.powertip.js
      jquery.powertip.min.css
      jquery.powertip.css
      ```

   e. Copy both the min and non-min versions of the files. The min version is used automatically by default for performance and the non-min version is used automatically when debug mode is on.
   f. This example uses the default color scheme for the tips. But, PowerTip comes with multiple color schemes as defined in separate css files in its CSS folder. You can change to one of the other color schemes or create one of your own by placing the different css files into your head folder. For example, to get the light color scheme, you can instead copy the css files into your head folder.

```
jquery.powertip-light.min.css
jquery.powertip-light.css
```

g. Your `jquery_powertip` module `prereqs` the jQuery module, so create a file that is named `prereqs.properties` in your `modules\jquery_powertip` folder with the following content:

```
jquery
```

h. If you have more plug-ins, similar module definition folders can be defined for them in your theme's modules folder. For each module definition, give the module a folder name that starts with `jquery_`, such as `jquery_powertip` in this example. Prereq the `jquery` module.

3. Add the module `jquery` to the module listings in one of your theme's profiles. Rather than modify one of the existing profile files, it is best to copy one of them and modify the copy. Copy the base profile file that you want to be similar to, such as `profile_deferred.json`, rename it to something such as `profile_jquery_deferred.json` and add the module `jquery`. When you create a profile file, adjust the titles and descriptions so that your profile has a unique and recognizable title and description. For example, change the titles and descriptions to

```
"titles": [
      {
         "value": "jQuery Deferred",
         "lang": "en"
      }
   ],
   "descriptions": [
      {
         "value": "This profile has jQuery plus the full set of modules for the theme, but def
         "lang": "en"
      }
   ]
```

4. Optional: Add the plug-in modules to the module listings in your profile, such as add the module `jquery_powertip` in this example.
5. Invalidate the cache so that your profile and module changes are picked up by the Portal server. Click the **Administration menu** icon. Then, click **Portal Analysis** > **Theme Analyzer**. Then, click **Utilities** > **Control Center** > **Invalidate Cache** > **Click to invalidate**. CF06 Auto invalidation recognizes your changes automatically for WebDAV based themes. No further action is required. For more information, see Utilities.
6. Apply your profile to a page. In **Edit Page Properties** > **Advanced**, select your Profile: `jQuery Deferred`.
7. To verify that jQuery is loaded on a page and what version it is, you can run `$().jquery` in the console of your browser's developer tools. It echoes the jQuery version number.

Results

Now the jQuery core and your plug-ins are ready and available for use. To learn the basics about jQuery core usage, see *jQuery basics* in the related links. You can learn the usage of your plug-ins at the same site where you downloaded the plug-in. For more information, see *PowerTip download*.

"Defining tooltips with PowerTip" on page 2788
Once your PowerTip module is active, you can add one or more tooltips to your pages through a dynamic content spot, or a config markup type subcontribution in a module.

Related information:

jQuery plug-in download

jQuery PowerTip plug-in download

jQuery basics

PowerTip download

Defining tooltips with PowerTip:

Once your PowerTip module is active, you can add one or more tooltips to your pages through a dynamic content spot, or a config markup type subcontribution in a module.

About this task

The PowerTip offers flexible tip positioning at the compass points. You can position the tip so it is never clipped off the edge of the view port. And, you can let the tip remain visible while the mouse is on it so the user can interact with the contents of the tip. Here is an example of PowerTip usage code for those options:

```
<script type="text/javascript">
 $(function() {
  $('#myelement').data('powertiptarget', 'mytip');
  $('#myelement).powerTip({ placement: 's', mouseOnToPopup: true, smartPlacement: true });
 });
</script>
<div id="mytip" style="display:none">
 <input type="text" />
 <input type="button" value="Search" />
</div>
```

In this example, *myelement* must be replaced with the ID of the element on the page that you want to add the tip to. And *mytip* is the ID of the `div` that is the tip itself, which is defined by the `div` in the example. Notice that the tip `div` is set to `display:none` initially. The PowerTip shows it when the user mouses over the `myelement` element.

You can define multiple tooltips by copying the data and PowerTip lines for each tooltip and copying the `div` block for each tooltip. Modify the IDs for each one.

You can put the example code anywhere on the page. You can add the tooltip to a dynamic content spot to add the code to the page. Or, you can add the code to a config markup type subcontribution of a module.

If your markup is visible and needs to be injected at a specific point on a page, use a dynamic content spot. If your markup is not visible initially and can be injected anywhere on a page, use the subcontribution method. The tooltip code can be anywhere on the page, so use the subcontribution method and follow these steps

Procedure

1. Create a `theme_powertips` subfolder in your theme's modules folder.
2. Your `theme_powertips` module will prereq the `jquery_powertip` module, so create a file that is named `prereqs.properties` in your `theme_powertips` folder with the following content:

   ```
   jquery_powertip
   ```
3. Create a config subfolder in your `theme_powertips` folder.

4. Create a `theme_powertips.html` markup file in your `config` folder. You can copy and paste the following code as an example:

```
<script type="text/javascript">
 $(function() {
  $('#wpthemeHelp img').data('powertiptarget', 'wpthemeHelpTip');
  $('#wpthemeHelp img').powerTip({ placement: 's', mouseOnToPopup: true, smartPlacement: true
 });
</script>
<div id="wpthemeHelpTip" style="display:none">
 <form method="get" action="http://www-10.lotus.com/ldd/portalwiki.nsf/xpSearch.xsp" target="w
  <input type="text" name="searchValue" />
  <input type="submit" value="Search" />
 </form>
</div>
```

   This example assumes that your theme has a help element with ID `wpthemeHelp` similar to the default theme. If your theme does not have an element with ID `wpthemeHelp`, then change the two instances of `'#wpthemeHelp img'` to a selector that identifies one of the elements in your theme, and rename the two instances of `'wpthemeHelpTip'`.

   This example uses a simple form on the tooltip that searches the IBM WebSphere Portal Express wiki documentation. You can change the markup in your tooltip `div` entirely to whatever you need to appear on your tooltip.
5. Add the module `theme_powertips` to the module listings in your jQuery profile file.
6. Invalidate the cache so the Portal server includes your profile and module changes. Click the **Administration menu** icon. Then, click **Portal Analysis** > **Theme Analyzer**. Then, click **Utilities** > **Control Center** > **Invalidate Cache**. CF06 Auto invalidation recognizes your changes automatically for WebDAV based themes. No further action is required. For more information, see Utilities.

Results

Go to your page that has your jQuery profile applied, move the mouse over the help icon, or whichever theme element you specified in step 4, and you see your tip appear.

If things do not work correctly, you can troubleshoot your modules. Click the **Administration menu** icon. Then, click **Portal Analysis** > **Theme Analyzer**. Then, click **Examine Modules by Profiles**. Expand your theme in the tree and select your jQuery profile, such as **jQuery Deferred**, in the tree and examine the profile. Double-click your jQuery profile in the tree. Expand the Modules folder in the tree and select one-by-one your various jQuery modules, **jquery**, **jquery_powertip**, and **theme_powertips**. For each, examine the details to see if everything looks correct (in particular the prereqs and the contributions). If you spot and fix a problem, be sure to invalidate the cache after so the Portal server picks up the change.

Using Sametime with the WebSphere Portal Version 8.5 theme

You can add modules to your profile to use IBM Sametime with the IBM WebSphere Portal Express Version 8.5 theme.

About this task

To gain Sametime awareness with the WebSphere Portal Express Version 8.5 theme, you must add one of the following modules to your profile:

- **`wp_sametime_links`**: Provides existing STlinks support.
- **`wp_sametime_proxy`**: New Sametime proxy support.

When using the **wp_sametime_proxy** module you must update the following attribute in the **WP CommonComponentConfigService** resource environment provider on the WebSphere Portal server to activate the module:

Procedure

1. Set the **cc.sametime.proxy.enabled** attribute to `true`.
2. Set the **cc.sametime.proxy.scheme** attribute to `http` or `https`, depending on the way your Sametime Proxy Server is accessed.
3. Set the **cc.sametime.proxy.host** attribute to `hostname`. Include the domain name in the attribute, for example `hostname.domainname.com`.
4. Set the **cc.sametime.proxy.port**.
5. Set the **cc.sametime.connect.client** attribute to `true` if you want the Sametime Proxy to use the Sametime connect client, which is installed on the system with Sametime Proxy Server.
6. Set the **cc.sametime.proxy.version** attribute to 8.5.2 if you want the integrate with Sametime Proxy 8.5.2. Otherwise, set the attribute to 8.5.1 if you want to integrate with Sametime Proxy 8.5.1.

What to do next

For more information about using Sametime with WebSphere Portal Express, see Integrating with IBM Sametime.

Tags used by the portal JSPs

Learn about the most commonly used tags in the portal JSPs. Use these tags to modify the appearance and layout of the portal page.

The following links provide topics with summary descriptions of each tag - grouped by tag type. Each separate topic also provides tag descriptions and code examples:

Note: Do not use portal tags in portlet JSPs. The tags that are mentioned are only for use in theme and skin JSPs.

- "<portal-core/> tags" on page 2791 - Used to provide portal core functions such as entering the main render flow and URL-related aspects of the page.
- "<portal-dynamicui/> tags" on page 2794 - Used to enable dynamic user interface features such as closing dynamic portlets and pages.
- "<portal-fmt/> tags" on page 2794 - Used to provide enhanced portal formatting capabilities.
- "<portal-logic/> tags" on page 2799 - Used to provide conditional logic.
- "<portal-navigation/> tags" on page 2805 - Used to implement navigation tasks such as generating URLs and traversing the portal navigation model.

To use these tags, the following `taglib` declarations must be provided in the parent JSP of the theme:

```
<%@ taglib uri="http://www.ibm.com/xmlns/prod/websphere/portal/v8.5/portal-navigation"
  prefix="portal-navigation" %>
<%@ taglib uri="http://www.ibm.com/xmlns/prod/websphere/portal/v8.5/portal-dynamicui"
  prefix="portal-dynamicui" %>
<%@ taglib uri="http://www.ibm.com/xmlns/prod/websphere/portal/v8.5/portal-logic"
  prefix="portal-logic" %>
<%@ taglib uri="http://www.ibm.com/xmlns/prod/websphere/portal/v8.5/portal-core"
  prefix="portal-core" %>
<%@ taglib uri="http://www.ibm.com/xmlns/prod/websphere/portal/v8.5/portal-fmt"
  prefix="portal-fmt" %>
```

The following `taglib` declarations must be provided to the parent JSP of the skin - in addition to the ones in the theme:

```
<%@ taglib uri="http://www.ibm.com/xmlns/prod/websphere/portal/v8.5/portal-skin"
  prefix="portal-skin" %>
<%@ taglib uri="http://www.ibm.com/xmlns/prod/websphere/portal/v8.5/portal-showtools"
  prefix="portal-showtools" %>
```

Notes:

- The tags in the `portal-internal.tld` tag library are not intended for customization, but for use only by internal portal code.
- The tags in the `engine.tld` and `portal-internal.tld` tag libraries are not intended for customization, but only to support compatibility with an earlier version and migration. The `engine.tld` tag library are not available in subsequent releases of WebSphere Portal Express.

  "<portal-core/> tags"
  The <portal-core/> tags are used to provide portal core functionality such as entering the main render flow as well as URL-related aspects of the page.

  "<portal-dynamicui/> tags" on page 2794
  The <portal-dynamicui/> tags are used to enable dynamic user interface features such as closing dynamic portlets and pages.

  "<portal-fmt/> tags" on page 2794
  The <portal-fmt/> tags are used to provide enhanced portal formatting capabilities.

  "<portal-logic/> tags" on page 2799
  The <portal-logic/> tags are used to provide tags for conditional logic.

  "<portal-navigation/> tags" on page 2805
  The `<portal-navigation/>` tags are used to implement navigation tasks such as generating URLs and traversing the portal navigation model.

<portal-core/> tags:

The <portal-core/> tags are used to provide portal core functionality such as entering the main render flow as well as URL-related aspects of the page.

The following table provides a brief description of each tag.

Note: Do not use portal tags in portlet JSPs. The following tags are only for use in theme and skin JSPs.

Table 447. Descriptions of <portal-core/> tags

Tag	Description
<portal-core:cacheProxyUrl/>	Creates a URL to the caching proxy servlet. **Note:** This tag is deprecated for IBM WebSphere Portal Express 8.0.
<portal-core:constants/>	Makes the <%= wpsBaseURL %> and <%= wpsDocURL %> constants available to the page.
<portal-core:defineObjects/>	Defines a set of objects which can be used later on (for example, `escapeXmlWriter`).
<portal-core:init-lazy-set/>	This tag initializes the lazy set functionality and is required to be called in the outer most JSP. For example in your theme it would be the `default.jsp`. It must not be placed in a dynamic content spot JSP.

Table 447. Descriptions of <portal-core/> tags (continued)

Tag	Description
<portal-core:lazy-set/>	This tag evaluate the given EL expression unless it has been cached already with the current request and stores a variable in the request scope with the given name (var). Attributes: **var** - variable name to be set into the request. **elExpression** EL expression to be evaluated. It must not contain the ${} because the JSP engine evaluates everything that is surrounded by ${} as an EL Bean. This invalidates all performance improvements. For example: `<portal-core:lazy-set var="deviceClass" elExpression="wp.c`
<portal-core:pageRender/>	Used in the Home screen to render the content of the selected page. Do not confuse this with the <portal-core:pageRender/> tag deprecated in V4.2.
<portal-core:screenRender/>	Starts the rendering of the screen. This tag should be used only in theme JSPs.
<portal-core:stateBase/>	Stores a base URL which can be used instead of full, newly-coded URLs on each occurrence of a URL in the markup. This tag should occur only in the header section of the markup, which is provided by theme JSPs.

Detailed descriptions of the <portal-core/> JSP tags

The following section provides detailed descriptions of the <portal-core/> JSP tags:

<portal-core:cacheProxyUrl forwardurl="*url_string*"/>

Creates a URL to the caching proxy servlet. The URL created is fully cacheable and includes information about the requesting client. The CC/PP client profile is used for gathering information about the client for the URL. The purpose of this tag is to link .CSS files into the JSP. The tag has one attribute.

Note: For security reasons, the cache proxy servlet will only serve URLs pointing to resources located in the themes, skins, and screens directories. This makes all resources underneath these directories public. Also, any URLs containing the ".." characters will not be served.

Attributes

forwardurl

Specifies the address to the file you want to be sent to the caching proxy servlet. This attribute can be set by passing in a String or a URL object.

The following code example uses this tag to link a CSS file in a JSP:

```
<link href="<portal-core:cacheProxyUrl
      forwardurl="<portal-logic:urlFindInTheme type="css"
               file="styles.jsp"/>"/>">
```

<portal-core:constants/>

Makes the following constants available to the page:

- <%= wpsBaseURL %>

  Provides the context path of the portal site as specified during installation, for example, `/wps`

- <%= wpsDocURL %>

  Provides the URL to the product documentation located in the WebSphere Portal Express Web application directory, including the Javadoc information, and portal help. The URL returned includes the locale of the client, for example, `/wps/doc/locale/`. For example, the following code would generate a URL to the portal help:

  ```
  <%= wpsDocURL %>/help/index.html
  ```

<portal-core:defineObjects/>

Defines a set of objects which can be used later (for example, `escapeXmlWriter`). Makes the following constants available to the page:

- <%= escapeXmlWriter %>

  This writer can be used to stream out URLs. The writer makes sure that the URL is written out in XML compliant representation:

  ```
  <portal-navigation:urlGeneration contentNode="wps.Links" portletMode="view">
    <c:set var="title"><portal-fmt:text bundle="nls.bundle" key="link" /></c:set>
    <a href="<% wpsURL.write(escapeXmlWriter); %>"><c-rt:out value='${title}' escapeXml='true' /></a>
   </portal-navigation:urlGeneration>
  ```

<portal-core:pageRender/>

Used in the Home screen to render the content of the selected page. Do not confuse this with the <portal-core:pageRender/> tag deprecated in V4.2. This tag renders the portal page without the navigation. When this tag is called, all pages that are available for the user are determined.

<portal-core:screenRender/>

Renders the current screen. Types of screens include Home (default), Login, and Error. This tag can only be used once in the portal. This tag should be used only in theme JSPs.

Example

This part of the `Default.jsp` portlets.

```
<div id="FLYParent">
 <%@ include file="./banner.jspf" %>
 <portal-logic:if portletSolo="no">
   <%@ include file="./topNav.jspf" %>
   <%@ include file="./sideNav.jspf" %>
 <portal-logic:if/>

 <a name="wpsMainContent">
 <%-- Call the portal engine command to render the portlets for this page --%>
 <div id="mainContent"><portal-core:screenRender/></div>

 <portal-logic:if portletSolo="no">
   <%@ include file="./footer.jspf" %>
   <%@ include file="./flyout.jspf" %>
 </portal-logic:if>
 <portal-logic:if loggedIn="yes">
</div>
```

<portal-core:stateBase/>

Stores a base URL which can be used instead of full, newly-coded URLs on each occurrence of a URL in the markup. This enables shorter URLs and can improve the page serving performance. This tag should occur only in the header section of the markup - it is not allowed to occur outside of the header section. Theme JSPs are responsible for the header section, using `Head.jsp` by default.

<portal-dynamicui/> tags:

The <portal-dynamicui/> tags are used to enable dynamic user interface features such as closing dynamic portlets and pages.

The following table provides a brief description of each tag.

Note: Do not use portal tags in portlet JSPs. The following tags are only for use in theme and skin JSPS.

Table 448. Tags for themes and skins JSP files

Tag	Description
<portal-dynamicui:closepage/>	Provides a URL that closes a dynamic page. This tag should be used only within a <portal-navigation:navigationLoop/>
<portal-dynamicui:closeportlet/>	Provides a URL that closes a dynamic portlet. This tag should be used only within a skin JSP.
<portal-dynamicui:pendingTasks/>	Through the use of scripting variables, notifies the user whenever a new, unclaimed task has been assigned. This tag should be used only within theme JSPs.

Detailed descriptions of the <portal-dynamicui/> tags

The following section provides detailed descriptions of the <portal-dynamicui/> JSP tags:

<portal-dynamicui:closePage/>
Provides a URL that closes a dynamically created task page. This tag should be used only within a <portal-navigation:navigationLoop/>.

<portal-dynamicui:closePortlet/>
Provides a URL that closes a dynamic portlet. This tag should be used only within a skin JSP. The content of this tag is rendered only if the portlet is dynamic.

<portal-dynamicui:pendingTasks/>
Through the use of scripting variables, notifies the user whenever a new, unclaimed task has been assigned. This tag should be used only within theme JSPs.

<portal-fmt/> tags:

The <portal-fmt/> tags are used to provide enhanced portal formatting capabilities.

The following table provides a brief description of each tag.

Note: Do not use portal tags in portlet JSPs. The following tags are only for use in theme and skin JSPS.

Table 449. Descriptions of <portal-fmt/> tags

Tag	Description
<portal-fmt:answer/>	Returns the answer text for a key in the specified language. This tag can be used only within a <portal-skin:portletRender/> or <portal-core:pageRender/> tag.
<portal-fmt:bidi/>	This tag is used to support the display of bidirectional languages.
<portal-fmt:description/>	Provides the description of an object that implements the `Localized` interface. This tag can be used in both theme and skin JSPs.
<portal-fmt:identification/>	Transforms a String representation of the ObjectID into an ObjectID. Also transforms an ObjectID into a String representation of the ObjectID.
<portal-fmt:problem/>	Returns the problem text for a key in the specified language. This tag can be used only within a `<portal-skin:portletRender/>` or `<portal-core:pageRender/>` tag.
<portal-fmt:text/>	Returns the text for a key in the specified language. For information about how to set the bundle refer to the `portletExt:setBundle` tag under "JSP tags for standard portlets" on page 3133.
<portal-fmt:textParam/>	If the text that is retrieved contains placeholders in the form of "{0}", "{1}", "{2}", it can be set with this tag. This tag can be used only in the body of the `<portal-fmt:text/>` tag.
<portal-fmt:title/>	Provides the title of an object that implements the interface. This tag can be used in both theme and skin JSPs.
<portal-fmt:user/>	If the user is logged in, this tag returns one of the specified values of **`attribute`**.

Detailed descriptions of the <portal-fmt/> tags

The following section provides detailed descriptions of the <portal-fmt/> JSP tags:

<portal-fmt:answer bundle="*bundle*">

Returns the answer text for a key in the specified language. The following keys can be looked up in the resource bundle.

- content.not.available.answer
- login.invalid.answer
- password.invalid.answer
- userid.invalid.answer
- portlet.not.active.answer
- portlet.not.authorized.answer
- portlet.not.available.answer
- portlet.title.not.available.answer

Each key corresponds to a problem key (without the `.answer` suffix). See the <portal-fmt:problem> tag for a complete description of each problem and an example.

<portal-fmt:bidi dir="rtl|ltr" attribute="portlet" locale="*locale*">

This tag is used to support bidirectional languages. Bidirectional languages contain text that reads in both directions. For example, URLs, code samples, or directory and file names can be read in the opposite direction of the rest of the text.

Attributes:

- `dir`

  Indicates the normal direction of text in the language. This attribute is required.

  - For `dir="rtl"`, the tag content is written only if the client's locale belongs to a bidirectional language. This is the default setting if `dir` is not specified.
  - For `dir="ltr"`, the tag content is written only if the client's locale does not belong to a bidirectional language.

- `attribute="portlet"`

  Indicates that the text is for the title of a portlet. This attribute is optional.

- locale

  The tag content is written only if the language belongs to the bidirectional languages that are defined by the portal. If attribute is specified, locale is ignored.

WebSphere Portal Express JSPs use this tag in a `BidiInclude.jsp` that creates the following scripting variables, which, in many cases, are easier to use than the `<portal-fmt:bidi/>` tag.

<%=bidiAlignRight%>

Resolves as the value `left` for bidirectional languages, `right` for all other languages. This is primarily intended for text alignment, such as in a table cell.

<%=bidiAlignLeft%>

Resolves as the value `right` for bidirectional languages, `left` for all other languages. This is primarily intended for text alignment, such as in a table cell.

<%=bidiImageRight%>

Resolves as the value `Left` for bidirectional languages, `Right` for all other languages. This is primarily intended to append the value to the file name for an image. For example, the theme directory provides two image files that can be used for the portlet window, `Album_Border_TopLeft.gif` and `Album_Border_TopRight.gif`.

<%=bidiImageLeft%>

Resolves as the value `Right` for bidirectional languages, `Left` for all other languages. This is primarily intended to append the value to the filename for an image. For example. The theme directory provide two image files that can be used for the portlet window, `Album_Border_TopLeft.gif` and `Album_Border_TopRight.gif`.

<%=bidiImageRTL%>

Resolves to the value _rtl for bidirectional languages, null for all other languages. This is primarily intended for graphic images. For

example, the skin previews provide two image files to preview the skin, `preview.gif` and `preview_rtl.gif`.

The following example shows how a directional image is invoked. The <%=bidiImageRTL%> scripting variable is used to invoke the appropriate icon, for example, `tab_next_rtl.gif`, when the locale belongs to a bidirectional language.

```
<img alt="<portal-fmt:text key="link.next" bundle="nls.engine"/>"
     src="<portal-logic:urlFindInTheme file='<%="tab_next"+bidiImageRTL+".gif"%>'/>
     border="0" align="absmiddle">
```

<portal-fmt:description locale="*locale*" varname="*scripting_variable*" />

Provides the description of the object that is specified by `varname` or of the navigation node that is set in the `<portal-navigation:navigationLoop/>` tag. This tag can be used in both theme and skin JSPs.

Attributes

locale Optional. Overrides the current language setting.

varname
Optional. Specifies the name of the scripting variable holding the object. The <portal:navigation/> tag sets this object to <%=wpsNavNode%>.

```
<portal-navigation:navigationLoop>
  <!-- write the description of the node in the current locale -->
    <portal-fmt:description varname="<%=wpsNavNode%>"/>
  <!-- write the description of the node in chinese -->
    <portal-fmt:description varname="<%=wpsNavNode%>" locale="zh"/>
</portal-navigation:navigationLoop>
```

<portal-fmt:identification action="*setting*" object="*object_name*" var="*variable_name*"/>

Transforms a String representation of the ObjectID into an ObjectID. Also transforms an ObjectID into a String representation of the ObjectID.

Attributes:

action Specifies the operation to be done on the object. Allowed values are `"serialize"` and `"deserialize"`

object Specifies the object that must be handled. Allowed values are an object of the type `com.ibm.portal.Identifiable` or `java.lang.String`.

var Specifies the name of the scripting variable where the generated value is stored.

The following example shows how this tag is used to set an identifiable object and translate it to a String representation:

```
<portal-fmt:identification object="<%=contentNode%>" action="serialize" var="oid_string">
 <p><portal-fmt:title varname="<%=contentNode%>"/> has the ObjectID <%=oid_string%></p>
</portal-fmt:identification>
```

The following example shows how this tag is used to set a unique name, which can also be an object ID string representation, and translate it to an object ID:

```
<portal-fmt:identification object="ibm.portal.Home" action="deserialize" var="oid">
 <% if (oid instanceof com.ibm.portal.ObjectID) { %>
  <p>This is how it works</p>
 <% } %>
</portal-fmt:identification>
```

<portal-fmt:problem bundle="*bundle*">

Returns the problem text for a given key in the specified language. The following keys can be looked up in the resource bundle.

- content.not.available - Occurs if there is no page content that can be displayed. Used in the <portal-core:pageRender/> tag.
- login.invalid - Occurs when the user ID, password, or both are not valid. This is used in the Login screen.
- password.invalid - Occurs when the password field is empty during login. This is used in the Login screen.
- portlet.not.active - Occurs when a portlet is not active. Used in the <portal-skin:portletRender/> tag.
- portlet.not.authorized - Occurs when a user does not have appropriate permissions on a portlet. Used in the <portal-skin:portletRender/> tag.
- portlet.not.available - Occurs when an error occurs that prevents a portlet from rendering. Used in the <portal-skin:portletRender/> tag.
- portlet.title.not.available - Occurs when a portlet title is not available. Used in the <portal-skin:portletTitle/> tag.
- userid.invalid - Occurs when the user ID field is empty during login. This is used in the Login screen.

There is a corresponding answer key for each problem (see the <portal-fmt:answer/> tag).

The resource bundle must be in a directory that is on the class path of the WebSphere Portal Express enterprise application. It is recommended that you create a new directory to separate your custom code from the base code. Add the new directory to the Portal application class path. For details, refer to the WebSphere Application Server documentation. The following sample retrieves the problem text from `/nls/problem_`*`locale`*`.properties`. The text is displayed only when an error is encountered rendering the page.

For example:

```
<portal-core:pageRender>
   <p align="center">
     <strong><portal-fmt:problem bundle="nls.problem"/></strong>
     <br>
     <portal-fmt:answer bundle="nls.problem"/>
   </p>
</portal-core:pageRender>
```

<portal-fmt:text key="*key*" bundle="*bundle*">

Returns the text for a key in the specified language. The *key* indicates a parameter in a resource bundle or properties file, indicated by *bundle*. Both attributes are required. See the description of <portal-fmt:textParam> for an example and further description of the attributes of <portal-fmt:text/>.

Where possible, the `<i18n:bundle/>` and `<i18n:message/>` tags of the I18N tag library can be used instead of <portal-fmt:text/>..

<portal-fmt:textParam>

If the text that is retrieved contains placeholders in the form of "{0}", "{1}", "{2}", these can be set with this tag. This tag can be used only in the body of the <portal-fmt:text/> tag. The text to be substituted for the placeholders must be entered as content for this tag. It can even be an expression.

Examples

The **welcome** parameter in a resource bundle is set as follows:

```
welcome = Welcome {0}!
```

In the following example, the value of the <portal-fmt:textParam> tag is written in place of the {0}.

```
<portal-fmt:text key="welcome" bundle="nls.engine">
    <portal-fmt:textParam>World</portal-fmt:textParam>
</portal-fmt:text>
```

<portal-fmt:title locale="*locale*" varname="*localized_object*"/>

Provides the title of the object that is specified in `varname`. The tag also implies the object if it is called inside of the <portal-navigation:navigationLoop/> tag.

Attributes

locale Optional. Overrides the current language setting.

varname

Optional. Specifies the name of the scripting variable holding the object. The <portal-navigation:navigation/> tag sets this object to <%=wpsNavNode%>.

<portal-fmt:user attribute="*value*"/>

If the user is logged in, this tag returns one of the specified user attributes. *value* can be any user attribute that is defined to Member Manager.

<portal-logic/> tags:

The <portal-logic/> tags are used to provide tags for conditional logic.

<portal-logic:find>

Used to access the portal-wide find URL that is specified in **Portal Settings**. The tag is conditional. If no URL is set, the body of the tag is not evaluated. A scripting variable that is called <% wpsPortalFindURL %> is available inside the body of the tag for accessing the URL. This tag can be used in both theme and skin JSPs.

<portal-logic:if *attribute*="*value*">

Through the attributes of this tag, several conditions can be checked. If the condition is true, the content of the tag's body is written to the page. Otherwise, the content is skipped. More than one condition can be evaluated. For example, the user must be logged in and the Home screen must be active for the content of the following tag to be rendered:

Note: All earlier attributes that started with "not" are deprecated. Use the <portal-logic:unless/> tag instead of these attributes.

```
<portal-logic:if loggedIn="yes" screen="Home">

   <!--  content area -->

</portal-logic:if>
```

Attributes of the <portal-logic:if/> tag:

capableOf="*capability*"

Indicates whether the client supports the specified capability. *capability* can be one of the following values:

- HTML_2_0

- HTML_3_0
- HTML_3_2
- HTML_4_0
- HTML_CSS
- HTML_FRAME
- HTML_JAVA
- HTML_JAVASCRIPT
- HTML_NESTED_TABLE
- HTML_TABLE
- WML_1_0
- WML_1_2
- WML_TABLE

deviceClass="*deviceClass*"
Compares whether the indicated device classes match the current device class. The value is a string that represents the expected device classes, and can be a single device class name or an equation of multiple device class names that use + for AND, / for OR, ! for NOT, and parentheses.

Example values:

- `smartphone`
- `smartphone/tablet`
- `smartphone+!android`
- `(smartphone+ios)/(tablet+ios)`

If you have a syntax error in your equation, such as mismatched parentheses, an exception with details of the syntax error are written to the `SystemOut.log` file, and the attribute condition is shown as false. So, if the content of the tag's body is being skipped unexpectedly, check the log.

locale="*locale*"
Indicates whether the locale of the client is that of the specified *locale* (or subtype of the specified locale). You can specify a comma-separated list, such as `en, en_US`.

loggedIn="yes|no"
Indicates whether the user is logged in. For example, the following code displays a login link if the user is not logged in:

```
<%-- login button --%>
<portal-logic:if loggedIn="no" notScreen="Login">
   <td class="wpsToolBar" valign="middle" nowrap>
      <a class="wpsToolBarLink"
         href='<portal-navigation:url home="public" screen="Login" ssl="false"/>'>
            <portal-fmt:text key="link.login" bundle="nls.engine"/>
      </a>
   </td>
</portal-logic:if>
```

navigationAvailable="yes|no"
Indicates whether a navigation is available.

newWindow="yes|no"
Indicates whether the portlet is rendered in a separate browser window or iFrame from the portal (HTML only).

nodeInSelectionPath="yes|no"
Checks whether a page is selected. This means that the page is in the selected path by the user of which the content is shown.

pageAvailableNext="yes|no"
Checks if a subsequent set of pages is available to be accessed from the navigation.

pageAvailablePrevious="yes|no"
Checks if a previous set of pages is available to be access from the navigation. For example, use this tag in combination with <portal-navigation:navigationShift> to render a scroll icon. The scroll icon displays when the number of user-defined pages exceeds the number of displayed page tabs. This condition behaves contrary to the **pageAvailableNext** attribute.

pageCompletelyActive="yes|no"
Indicates whether the page and its parents are active. The following example displays a message if the page is not active.

```
<portal-logic:if pageCompletelyActive="no">
   <p align="center" class="wpsFieldErrorText"><b><br>
      &gt;&gt;&gt;
            <portal-fmt:text key="info.pagenotcompletelyactive" bundle="nls.er
      &lt;&lt;&lt;
   <br></b></p>
</portal-logic:if>
```

pageBookmarkable="true|false"
Renders its contents if the page can be bookmarked. A page can be set to bookmark with **Manage pages**. The following example provides an "Add to favorites" option in a select list. Users can bookmark the current page, if it can be bookmarked.

```
<portal-logic:if pageBookmarkable="true">
   <option value='<portal:url command="AddBookmark" alias="Favorites"/>' >
   <portal-fmt:text key='link.favorites.add' bundle='nls.engine'/>
</portal-logic:if>
```

portletMaximized="yes|no"
Renders its contents if the portlet is maximized. Use this call inside a JSP skin.

portletMode="edit|view|configure|help"
Checks if the portlet is in one of the modes. This tag is most useful in a customized skin.

Finding theme resources: See the *Location of theme resources* link in the Related section.

portletState="*portlet_state*"
Checks if the portlet is in the indicated state. Portlet states are `normal`, `maximized`, and `minimized`. For example, in `Control.jsp`, the following code sets the `tableHeight` variable to 100%.

```
<portal-logic:if portletState="Normal,Maximized">
    <% tableHeight = "height=\"100%\""; %>
</portal-logic:if>
```

portletSolo="yes|no"
Checks whether the portal is displaying a portlet in solo state. In the following example from `Default.jsp`, the navigation are hidden when the current portlet is displayed in solo state.

```
<portal-logic:if portletSolo="no">
   <%@ include file="./topNav.jspf" %>
   <%@ include file="./sideNav.jspf" %>
</portal-logic:if>
```

problem="*problem*"
Renders its contents if one of the following problems occurred.

- content.not.available
- login.invalid
- password.invalid
- portlet.not.active
- portlet.not.authorized
- portlet.not.available
- portlet.title.not.available
- userID.invalid

resumeLevel="0|1|2"
Used in the Login.jsp screen to write the content of the tag if the value of this attribute is equal to the setting of the **persistent.session.level** key in the ConfigService.

resumeOption="0|1"
Used in the Login.jsp screen to write the content of the tag if the value of this attribute is equal to the setting of the persistent.session.option key in the ConfigService.

screen="*screen_name*"
Checks the value of the current screen name. Use a comma to separate multiple screen names. In the following example, the content of the tag is displayed only when the selected screen is Home, LoggedIn, or LoggedOut.

```
<portal-logic:if navigationAvailable="yes" screen="Home,LoggedIn,LoggedOut">

....

</portal-logic:if>
```

selection
Specifies the unique name or object ID of the currently selected page. For example:

```
<portal-logic:if selection="ibm.portal.Home">
  You are on Home
</portal-logic:if>
```

showTools="yes|no"
Indicates whether to display more controls for the portlet title bar and page tabs. In the following example, the show_tools_off.gif icon is displayed for the condition when showTools="no".

```
<portal-logic:if showTools="no">
 <% if (firstButton) { firstButton = false; } else { %> | <% } %>
   <a href='<portal-navigation:url command="ShowTools"/>'>
      <img border="0" align="absmiddle" width="16" height="19"
          src='<portal-logic:urlFindInTheme file="show_tools_off.gif"/>'
          alt='<portal-fmt:text key="link.show.tools" bundle="nls.engine"/>'
          title='<portal-fmt:text key="link.show.tools" bundle="nls.engine"/>'
      </a>
</portal-logic:if>
<portal-logic:if showTools="yes">
 <% if (firstButton) { firstButton = false; } else { %> | <% } %>
```

```
        <a href='<portal-navigation:url command="ShowTools"/>'>
           <img border="0" align="absmiddle" width="16" height="19"
                src='<portal-logic:urlFindInTheme file="show_tools_on.gif"/>'
                alt='<portal-fmt:text key="link.hide.tools" bundle="nls.engine"/>
                title='<portal-fmt:text key="link.hide.tools" bundle="nls.engine"
    </portal-logic:if>
```

For either condition, the <portal-navigation:url command="ShowTools"/> command is used to change the value of **showTools**. It allows the icon on the portal page to be used as a toggle. This condition is also checked in ShowTools.jsp to determine whether to render the move and delete portlet icons in the portlet title bar.

userImpersonated="true|false"

Used to access another user's system as though you are that user. Users such as support specialists can use the impersonation feature to find issues and errors. For example, if a portal administrator encounters a problem that they cannot resolve, a support specialist can use the impersonation feature and access that portal administrator's system to determine a solution to the problem.

For more information about the user impersonation feature, see Administering user impersonation.

<portal-logic:pageMetaData/>

Used to access metadata of the currently rendered page. The tag has two attributes:

varname

This attribute is mandatory and specifies the name of the variable that displays the metadata inside the body of the tag. The variable provides an object that implements com.ibm.portal.MetaData.

type This attribute is optional. Allowed values are direct and aggregated. The value aggregated is the default. If the type attribute is set to direct, only the metadata that is set specifically for the page are shown, otherwise the metadata of the page as provided by the content metadata model are used.

The Content metadata model is described in more detail in the Model SPI overview topic. The following example outputs a table with all names and values of the aggregated metadata of the current page.

```
<portal-logic:pageMetaData varname="pageMetaData">
  <table>
    <tr><th>Name</th><th>Value</th></tr>
    <c:forEach var="metaItem" items="${pageMetaData}">
      <tr><td>${metaItem.key}</td><td>${metaItem.value}</td></tr>
    </c:forEach>
  </table>
</portal-logic:pageMetaData>
```

<portal-logic:unless>

This tag operates in contrast to the <portal-logic:if> tag. Through the attributes of this tag, several conditions can be checked. If the condition is true, the content of the tag is not written to the page. Otherwise, the content is written. More than one condition can be evaluated. The following list of attributes can be evaluated. For more information, see the corresponding description for each attribute under the <portal-logic:if> tag.

- capableOf

- locale
- loggedIn
- newWindow
- navigationAvailable
- pageAvailableNext
- pageAvailablePrevious
- nodeInSelectionPath
- portletMaximized
- portletMode
- portletSolo
- portletState
- problem
- screen

<portal-logic:urlFind>

Generates a URL pointing to a file. The resource is searched under different paths that are based on the markup and locale that is supported by the client and the specified attributes of the tag.

Attributes

- `root="root"`
- `path="path"`
- `file="file"`

  This attribute is required
- `allowRelativeURL="true|false"`

The first place where the resource is found is used to construct the URL. The following search order that is used:

1. Root path
2. Markup name
3. Path
4. Markup version
5. Locale in diminishing sequence
6. File name

Read *Aggregation* for more information about how the portal server locates resources. The **allowRelativeURL** attribute indicates whether a fully qualified or relative URL is generated.

<portal-logic:urlFindInSkin file="*file_name*" id="*identifier*">

Similar to <portal-logic:urlFind>, this tag generates a URL that points to a file contained in the theme WAR file.

Finding theme resources: See the *Location of theme resources* link in the Related section.
The skin is taken from the user's or system's settings. The file attribute is required.

When id is specified, the tag initializes a scripting variable with the value normally written out and nothing is written to the output stream. The value of the id attribute is the name of the scripting variable.

Attribute:

forceAbsolute = "true|false"

This attribute is optional. It specifies whether the URL that is generated by this tag is to be absolute or not. If you set this attribute to `true`, absolute URLs are enforced; in these case other settings that affect the generation of URLs might be overridden.

<portal-logic:urlFindInTheme file="*file_name*" id="*identifier*">

Similar to <portal-logic:urlFind>, this tag generates a URL that points to a file contained in the theme WAR file.

Finding theme resources: See the *Location of theme resources* link in the Related section.
The theme is taken from the user or system settings. The file attribute is required.

When id is specified, the tag initializes a scripting variable with the value normally written out and nothing is written to the output stream. The value of the id attribute is the name of the scripting variable.

Attribute:

forceAbsolute = "true|false"

This attribute is optional. It specifies whether the URL that is generated by this tag is to be absolute or not. If you set this attribute to `true`, absolute URLs are enforced; in these case other settings that affect the generation of URLs might be overridden.

Related concepts:

"Administering user impersonation" on page 1230
Use the impersonation feature to access another user's system as though you are that user. Use caution with this configuration. The user who is doing the impersonation might get more permissions than initially granted to them.

"Understanding the Portal Version 8.5 modularized theme" on page 2521
Modern websites and browsers enable incredible new capabilities that can greatly enhance your user's web experiences. However, these capabilities are not without cost in terms of large page sizes and more processing in the browser when each page is rendered. These capabilities are worth it when you need them, but removing them for an entire site or including them only on pages that take advantage of these capabilities provides for more flexibility.

<portal-navigation/> tags:

The `<portal-navigation/>` tags are used to implement navigation tasks such as generating URLs and traversing the portal navigation model.

The following section provides detailed descriptions of the `<portal-navigation/>` JSP tags:

CF05 **<portal-navigation:uiNavigationModel var="*uiNavigationModel*" mobileDeviceClassTest="*device class equation*" showHidden="*true*|*false*" selectedNodeID="*page ID*" selectionPath="*page IDs*" selectionPathSeparator="*selectionPath delimiter*">**

The uiNavigationModel tag helps the JSP developer concentrate on building the navigation rather than worrying about the portal-specific implementation details.

By default, the uiNavigationModel lists the visible pages as part of its iterator. When the **Show hidden pages** option is selected in the toolbar, it lists the hidden pages also. There is a special mobile hidden flag for pages.

The model also allows you to specify a mobile test device class expression, which is used to evaluate if the system is rendered as part of a mobile request.

The uiNavigationModel tag makes the uiNavigationModel EL Bean available as a variable that can be specified as part of the var attribute.

Example:

```
<portal-navigation:uiNavigationModel var="uiNavigationModel" mobileDeviceClassTest="smartphon
    <%-- loop through all children of the page at the given curLevel --%>
    <c:forEach var="node" items="${uiNavigationModel.children[selectionPath[curLevel]]}">

        <%-- print out the page and highlight it if it is in the selection path
             (the current page or an ancestor of the current page) --%>
        <li class="wpthemeNavListItem wpthemeLeft<c:if test="${node.isInSelectionPath}"> wpth

            <%-- output a link to the page --%>
            <a href="${fn:escapeXml(node.urlGeneration)}" class="wpthemeLeft ${node.isHidden
                 'wpthemeHiddenPageText' : ''} ${node.isDraft? 'wpthemeDraftPageText' : ''}">

                <%-- start page title markup --%>
                <span lang="${node.title.xmlLocale}" dir="${node.title.direction}"><%--

                    <!-- print out the page title -->
                    --%><c:out value="${node.title}"/><%--

                    <!-- mark the page if it is currently selected for accessibility -->
                    --%><c:if test="${node.isSelected}"><span class="wpthemeAccess">
                    <portal-fmt:text key="currently_selected" bundle="nls.commonTheme"/></spa

                --%></span><%-- end page title markup --%>
            </a>
        </li>
    </c:forEach>
</portal-navigation:uiNavigationModel>
```

Required parameters:

var This mandatory attribute specifies the name of a scripting variable that is exposed in the body of the tag. This variable is a uiNavigationModel EL Bean.

mobileDeviceClassTest
This attribute is a device class equation, that describes whether the request is part of a mobile device or not.

The following parameters are all optional. If they are not provided, the information is automatically fetched from the request. In some cases, the selection information is not available and must be passed in explicitly. Use the following attributes in that case.

showHidden
Defines whether to show the hidden pages or not.

selectedNodeID
String; specifies the selected page ID.

selectionPath
String; list of page IDs defining the selection path, or breadcrumb, in one string. The delimiter can be configured through selectionPathSeparator.

selectionPathSeparator
String; is the delimiter that is used to parse the selectionPath.

Returns: Each individual navigation node that is returned by the uiNavigationModel is also a wrapper around the NavigaitonNode EL Bean and provides additional information, such as isHidden, isDraft, isInSelectionPath, and isSelected. For more information, see uiNavigationModel.

<portal-navigation:url command="ChangeLanguage">
This tag is used to change the active language in the navigational state in which the URL is generated.

The following code example uses this tag to change the language to German.

```
<a href='<portal-navigation:url
        command="ChangeLanguage"><portal-navigation:urlParam name="locale"
        value="de"/></portal-navigation:url>'>Diese Seite in deutsch
</a>
```

<portal-navigation:urlParam name="*parameter_name*" value="*parameter_value*">
This tag is used to add parameters to the parent URL. Parent tags include `<portal-navigation:url/>` and `<portal-navigation:urlGeneration/>`. Parameters added to the `<urlGeneration/>` tag occur as unscoped query parameters unless the attributes specified on `<urlGeneration/>` specify otherwise. Parameter handling depends on the target of the URL. If the URL points to a page, the parameters are visible to all IBM portlets on that page. Parameters are not visible to standard portlets if the URL does not point specifically to that portlet.

Attributes are as follows:

name This attribute is required. It indicates the name of the parameter.

value This attribute is required. It indicates the value of the parameter.

type This attribute is optional. It indicates one of the following types:

query The name-value pair is added to the URL as a query parameter. It is the default value if type is not specified.

render The parameter is available as a render parameter for the portlet.

action
The parameter is available as an action parameter for the portlet.

<portal-navigation:urlGeneration *attribute*="*value*">
This tag creates a URL to pages or portlets. The tag is conditional. If the URL cannot be found, the body of the tag is not evaluated. Inside the body of the tag, the <% wpsURL %> scripting variable can be used to write the URL directly to the output stream. For example:

```
<a class="wpsToolBarLink" href='<% wpsURL.write(out); %>'>My page</a>
```

Attributes are as follows:

accessControlCheck="permission_constant"
This attribute indicates the permissions to be checked. The following constants can be used:

- CreatePage
- EditLayout
- DeletePage
- AssignRoles
- NoCheck

- EditApplicationProperties
- EditApplicationLayout
- AssignApplicationMember
- ManageApplicationRoles
- SaveAsTemplate

If the user does not have the required permission, the body of the tag is not executed. If the user has the necessary permissions, the current page ID is appended to the URL. If the parameter is set to `'NoCheck'`, the current page ID is appended without checking the access control permissions. This is necessary for the target page or portlet to create a back button.

actionName="name"

This attribute indicates the name of an action that would be called by the URL to an IBM portlet. The following example generates a URL that calls the `myAction` action of the portlet, which is contained by `my.LayoutNode` on `my.ContentNode`.

```
<portal-navigation:urlGeneration actionName="myAction" contentNode="my.ContentNode"
                                 layoutNode="my.LayoutNode">
       <a href="<%wpsURL.write(out);%>">Link to portlet with myAction</a>
    </portal-navigation:urlGeneration>
```

The following code is from the `actionPerformed()` aspect of the portlet that handles the `myAction` action.

```
PortletURI actionURL = portletResponse.createURI();
actionURL.addAction("myAction");
```

allowRelativeURL="true | false"

This attribute indicates whether a fully qualified or relative URL is generated. The default is set by the property `com.ibm.portal.state.accessors.url.URLContext.enableRelative` in the `StateManagerService`.

contentNode="*id* | *name*"

This attribute indicates the ID or unique name of the page. The name or ID of the content node is also used to specify the page where the portlet can be found.

keepNavigationalState="true | false"

If you set this attribute to `false`, the current navigational state that includes all portlet modes, states, and render parameters is not included in the URL, and the portal is reset to its default state. If this value is set to `true`, the navigational state is included. True is the default setting.

layoutNode="*id* | *name* | *wp.currentSelectedPortlet*"

This attribute indicates the ID or unique name of the control that holds the portlet. It must be used in combination with `contentNode` to identify the page where the portlet is located. The value `wp.currentSelectedPortlet` can be used inside a control when you generate a URL to the portlet within that control.

newWindow="true | false"

This attribute creates a session partition. For portlet URLs, you can use it if you want to display the portlet either in a new window or in an iFrame. The default value is false. The portlet window state for the addressed portlet in the new window is set to maximized. The portlet mode is set to the value of the current parent window. In the `control.jsp` of skins that use iFrames, the `<portal:if/>` tag

can be used to distinguish between rendering in the main window or in an iFrame or detached window.

portletMode="view | help | edit | edit_defaults | configure"
For URLs to a portlet as specified by the parameter `layoutNode`, this attribute sets the portlet mode. This parameter is ignored if the parameter `layoutNode` is not set. Specifying `portletMode="edit_defaults"` opens the portlet in the Edit defaults mode directly.

portletParameterType="render | action"
This attribute generates URLs to a standard portlet's render or action processing methods. If this attribute is omitted, the render method of the portlet is called. Any parameters added to the body of the tag that uses `<portal-navigation:urlParam/>` are passed to the corresponding method.

portletWindowState="maximized | minimized | solo | normal"
For a portlet, this attribute indicates the state of the portlet window when it is displayed. If portlet state is not specified, the page is shown with the previous state of the portlet. This parameter is ignored if `layoutNode` is not set.

Note: Use caution when you use this tag to address portlets in solo state. The portlet must be able to exist in solo state that uses the `createReturnURI()` method. If a portlet without this method is placed in solo state, then users are forced to log out or close their browser windows to return to the portal.

themeTemplate="template_name"
This specifies the theme template that is taken for rendering the requested page. Can be referenced as either a JSP or a class and is used as theme for this URL.

normalize="true | false"
This attribute indicates that the URL to be generated must be normalized. If more parameters are set, the normalization is executed first and afterward the other state modifications are accomplished. Setting the "normalize" parameter to true normalizes the URL with the same XSL file used to normalize URLs for search engines. The normalized representation of the URL can also be used to bookmark a page. The following example shows how to use the tag with this attribute:

```
<portal-navigation:urlGeneration normalize="true" >
      <a href="<% wpsURL.write(out); %>">
        This is the normalized URL of the current
        selected page.
      </a>
</portal-navigation:urlGeneration>
```

If more parameters are set for the `<portal-navigation:urlGeneration>` tag the XSL transformation is executed first and all other state modifications is accomplished afterward.

forceAbsolute = "true | false"
This attribute is optional. It specifies whether the URL that is generated by this tag is to be absolute or not. If you set this attribute to `true`, absolute URLs are enforced; in this case other settings that affect the generation of URLs might be overridden.

<portal-navigation:url>

Creates a portal URL depending on the specified attribute. Attributes are as follows:

home="public | protected"

This attribute creates a URL pointing to the public or protected (logged in) page of the portal.

screen="screen_name"

This attribute creates a URL pointing to the screen name to be displayed.

command="LoginUser | LogoutUser | ShowTools"

This attribute creates a URL that issues the command to the portal. `command="LoginUser"` is used for the login panel, and `command="LogoutUser"` is used for the logout button. The `"userid"` and `"password"` parameters must be carried with a login URL. `command="ShowTools"` toggles the value of the showTools indicator. See the `<portal-logic:if/>` tag `"showtools"` attribute in for an example.

commandParam="parameter_name"

This attribute directs the portal engine to obtain the actual command to execute from an HTTP request parameter instead of on the URL directly. The name of the parameter is the value of the commandParam attribute. This is useful in situations where different commands need to be conditionally executed yet only one URL can be specified. Such is the case when using a `<form>` tag with a `<select>`. This enables use of the HTML form without requiring JavaScript. For example:

```
<form name="someFormName" method="GET" style="margin-bottom: 0"
            action='<portal-navigation:url commandParam="requestParamName"/>'>
        <select name="requestParamName" onchange="javascript: this.form.submit(); ">
                <portal-navigation:someLoop>
                        <option value="<%= theUrl %>" >Some title goes here
                </portal-navigation:someLoop>

        </select>
        <noscript>
                <input type="image" border=0 align=absmiddle src='go.gif'/>Go
        </noscript>
</form>
```

The previous code example works both with and without JavaScript enabled.

ssl="yes | no | true | false"

This attribute creates a secure URL (HTTPS).

forceAbsolute = "true | false"

This attribute is optional. It specifies whether the URL that is generated by this tag is to be absolute or not. If you set this attribute to true, absolute URLs are enforced; in this case other settings that affect the generation of URLs might be overridden.

Example: The following example shows part of a user login form that uses the `<portal:url>` tag to process input fields, user ID, and password.

```
<FORM method="POST"
     action='<portal-navigation:url command="LoginUser"/>'
     enctype="application/x-www-form-urlencoded"
     name="LoginPage">
```

<portal-navigation:navigationUrl type="link | expand | collapse | launch" varname="*node_name*" var="*variable_name*"/>

Creates URLs for navigation nodes. The tag is used inside the body of the <portal-navigation:navigationLoop> tag and outputs links for the current navigation node according to the *type* attribute.

Attributes are as follows:

type Use one of the following values:

type="link"
This attribute creates a URL to change the selected node.

type="expand"
This attribute creates a URL that expands the node to reveal its child nodes. This is intended for expanding the navigation tree.

type="collapse"
This attribute creates a URL that collapses the node to conceal its child nodes. This is intended for collapsing the navigation tree.

type="launch"
This attribute creates a URL that either launches a page if all conditions for the page launch are fulfilled for the navigation node or if just like in the selection URL in `type="link"`.

Note: The global state of the portal navigation trees is collapsed by default (with some exceptions, such as the Portal Administration navigation). You can configure the default state of the portal navigation trees to expand all nodes by setting the Portal Configuration Service property `navigation.expansion.defaultstate` to `true`.

varname
This attribute specifies an object of type `com.ibm.portal.navigation.NavigationNode` for which the URL is to be generated. The attribute is optional.

var This attribute specifies the name of a scripting variable that is exposed in the body of the tag. The attribute is optional. The variable exposes an object that implements `com.ibm.portal.DisposableURL` that can be used to stream the URL to the output. If the content node referenced by the navigation node is an internal URL, the body is evaluated only if the target of the internal URL is accessible.

forceAbsolute = "true | false"
This attribute is optional. It specifies whether the URL that is generated by this tag is to be absolute or not. If you set this attribute to `true`, absolute URLs are enforced; in this case other settings that affect the generation of URLs might be overridden.

keepNavigationalState="true | false | auto"
If this attribute is set to false, the current navigational state (including all portlet modes, states, and render parameters) is not included in the URL and the portal is reset to its default state. If set to true, which is the default, navigational state is included. If it is set to auto, the configuration determines if the navigational state is included. See the descriptions for `stateless.urls.enabled` and `generate.stateless.urls` in *Configuration Service*.

To generate the title or description of a navigation node, use the `<portal-fmt:title/>` or the `<portal-fmt:description/>` tags. This tag is used only in theme JSPs.

<portal-navigation:navigationShift by="*number*" maxPages="*number*">

This tag is used in the navigation to create a URL that scrolls to the next set of page links when the number of available pages exceeds maxPages. The *by* attribute indicates the number of page links to scroll. If this tag is not used, all page links for the current level are rendered.

This tag is nested in the <portal-navigation:navigation> tag and requires values for the startLevel and stopLevel attributes of the <portal-navigation:navigation> tag for its correct functioning.

<portal-navigation:navigationLoop>

This tag traverses through the navigation model. This tag is nested inside of the <portal-navigation:navigation/> tag when used. This tag indicates the part of the markup that is repeated once for each navigation node. There are no attributes for this tag. The body of this tag is executed for each navigation node. This tag makes several scripting variables available for obtaining information for the navigation. These scripting variables are accessible only within the body of the tag.

<portal-navigation:navigation>

This tag initializes a set of objects and makes them available through scripting variables. These objects are required for the rendering process of the navigation. The scripting variables are accessible only within the body of the tag. The settings of the `startLevel` and `stopLevel` attributes determine whether the content of the navigation tag is evaluated. The navigation tag uses an "in-order" traversal of the navigation tree to select the nodes.

Attributes are as follows:

startLevel

Optional. The level on which this navigation is to begin showing information. If no start level is given, this tag will start at the navigation node after the levels that where shown by other JSPs. Otherwise, the default is level 1.

stopLevel

Optional. The level on which this navigation is to stop showing information. Default is to render all levels.

computeNumLevelsToDisplay

Optional. Compute the number of levels that are shown by this navigation. When this attribute is set to `true`, a scripting variable is made available inside the body of the navigation tag named `wpsNumLevelsToDisplay` of Java type java.lang.Integer.

scopeUniqueName

Specifies where to start the rendering of the portal navigation. The expected value is a unique name or string representation of the navigation node's ObjectID. This also replaces the `<portal:favoritesLoop/>` tag for enabling Organize Favorites functionality. The following code example shows how to use the `<portal-navigation:navigation>` with the `scopeUniqueName` attribute to enable Organize Favorites:

```
<portal-navigation:navigation scopeUniqueName="wps.Favorites">
   <portal-navigation:navigationLoop>
     <a href='<portal-navigation:navigationUrl type="link"/>'
        title='<portal-fmt:description varname="<%=wpsNavNode%>"/>'>
           <portal-fmt:title varname="<%=wpsNavNode%>"/>
     </a>
   </portal-navigation:navigationLoop>
 </portal-navigation:navigation>
```

<portal:favoritesLoop/>

This tag is deprecated and must be replaced by using the `<portal-navigation:navigation/>` tag with its `scopeUniqueName` attribute.

The following code example uses this tag to create the Organize Favorites functionality. To use this code, you must place it within the themes that you want to have this functionality.

```
<%@ taglib uri="/WEB-INF/tld/portal.tld" prefix="portal" %>
<portal-logic:if loggedIn="yes" notScreen="SelfcareUserForm,SelfcareUserConf">
<table border="0" cellspacing="0" cellpadding="0" >
 <tr>
 <td align="<%=bidiAlignRight%>" valign="middle" dir="ltr" nowrap>
 <%
 boolean firstItem = true;
 %>
 <table border="0" cellpadding="0" cellspacing="0" <%= bidiDirAttr %> >
 <tr>
 <td valign="middle">
 <form tabIndex="8" name="wpsFavoritesSelectionForm" method="GET"
  style="margin-bottom: 0">
   <select name="favoritesCommand" onchange="document.location.href=
  this.options[this.selectedIndex].value">
   <option value='#' selected><wps:text key='link.favorites.myfavorites'
   bundle='nls.engine'/>

   <portal-logic:if pageBookmarkable="true">
   <option value='<portal-navigation:url command="AddBookmark"
   alias="wps.My Favorites"/>'><portal-fmt:text key='link.favorites.add'
   bundle='nls.engine'/>
   </portal-logic:if>

   <portal-navigation:urlGeneration contentNode="wps.Organize Favorites"
   portletWindowState="Normal" pacCheck="NoCheck">
   <option value='<% wpsURL.write(escapeXmlWriter); %>' >
   <portal-fmt:text key='link.favorites.orgainize' bundle='nls.engine'/>
   </portal-navigation:urlGeneration>

   <option value='#'>------------

    <portal:favoritesLoop>
    <%
    // wpsFavoritesURL is null for folders in the list of favorites
    if (wpsFavoritesURL != null)
    {
    // Check the favorite type.  If it is an external URL, add a symbol to the URL
    // so the JavaScript on the select can detect when to open a new window
    // wpsFavoritesType may be null.  wpsFavoritesType=1 means external URL
     if ("EXTERNALURL".equals(wpsFavoritesType))
     wpsFavoritesURL = "@" + wpsFavoritesURL;
    //Phone number links are only supported on WML devices... and
     favorites are not markup-specific.
    // wpsFavoritesType may be null.  wpsFavoritesType=2 means Phone Number
     if (!"2".equals(wpsFavoritesType))
     {
     %>
     <option value="<%= wpsFavoritesURL %>">
     <% for (int favSpace=1; favSpace < wpsFavoritesLevel.intValue();
     favSpace++) {%>   <% } %><%= wpsFavoritesTitle %>
     <%
     }
     }
     else
     {
     %>
     <option value="#" ><% for (int favSpace=1;
     favSpace < wpsFavoritesLevel.intValue();
     favSpace++) {%>   <% } %>--<%= wpsFavoritesTitle %>--
     <%
     }
     %>
     </portal:favoritesLoop>
    </select>
    <noscript>
    <input type="image" border="0" align="absmiddle"
    src='<portal-logic:urlFindInTheme file="go.gif"/>'/>
    <span class="wpsPlaceBarLink" > <portal-fmt:text key="go"
    bundle="nls.button"/></span>
    </noscript>
    </form>
    </td>
   </tr>
  </table>
  </td>
 </tr>
</table>
</portal-logic:if>
```

Developing themes for a production portal

Use the theme artifacts to package a theme for staging to production.

"Development and operations overview"
The steps for developing themes for a production portal, includes the development of the theme components, their packaging, and their deployment to systems.

"Packaging themes for deployment" on page 2816
You must repackage the static content as a WAR file, or an EAR file containing the WAR file when it is not possible to exchange the static content with the Operations Team as a separate compressed file.

"Exporting content from the filestore" on page 2820
You must export content from the filestore to create your custom theme. There are different options available to export files from the portal file store.

Development and operations overview

The steps for developing themes for a production portal, includes the development of the theme components, their packaging, and their deployment to systems.

The package is either created as the result of a build process that takes the code from a version control system and creates the correct artifacts or the artifacts are created by developers. Multiple artifacts create a release and the team that operates the integration and production servers receive and deploy them.

The components of a portal theme include:

Static content
Is similar to the content of a static website. The markup is defined by HTML files. Static content also includes the CSS and JavaScript files that are used by the themes.

The difference between a static website and a portal is that three kinds of HTML files exist:

`theme.html`
Defines the markup that is identical for all the pages that this theme is applied to.

`layout.html`
Defines the decoration of the content area that can be different between pages.

`skin.html`
Defines the decoration of the individual portlets on the page.

If the files used by the theme are not part of the theme folder, and therefore not part of the theme structure, then those files are named external. External files are shared by multiple themes.

Dynamic content
Generated based on the data model of WebSphere Portal Express with technologies such as JavaServer Pages or Java code.

Configuration
Scripts are used to register the theme and skins with WebSphere Portal.

Runtime Configuration components are server configurations, similar to Resource Environment Provider settings, that are required for the themes and skins to work correctly.

Developing Theme Components

There are several ways to develop a custom theme.

Developing within an IDE

With an integrated development environment, you can set up a project first, usually using the form that you selected for packaging afterward, and add the artifacts like HTML, CSS, and JavaScript files. Next, you export the project and deploy it to your server, either as a compressed file in the file store or an EAR file. Finally, you register the theme with a custom developed `theme.xml` file.

You might fill the project initially with the content of one of the ready-to-use themes or start from scratch.

Developing on the live server

This method of developing a theme was introduced recently and is driven through the WebDAV entry point to the file store. You can copy a theme to start with the existing content from one of the ready-to-use WebSphere Portal Express themes or create a folder to start. Add your custom HTML, CSS, and JavaScript files and export these files to create your theme component package. For more information about exporting files, see Exporting content from the file store.

Packaging theme Components

The packaging of a custom developed theme depends on how it was developed and what the preferred approach for static files is. For example, you can combine all static files for all themes and all external files that are shared between themes into one installable artifact or split the files into multiple artifacts. Because the method that you choose has different implications for the development team and the team operating the portal, include both groups when deciding how to package the components.

All packages must contain an XML Access script to deploy the theme and a list of runtime configurations that need to be applied to the server.

The following description shows the options available when all files are packaged together. If the files are separated, the number of files changes but the options are identical.

- If you only created static resources, but used dynamic spots that are included within WebSphere Portal Express or custom dynamic spots that are generally available, the following options exist:
  - Create a package that contains a compressed file that contains the static resources that can be deployed to the file store.
  - Create a package that contains an EAR file that contains a WAR file with the static resources that can be deployed to the application server.
- If you created custom dynamic spots, like components that are based on JSP technology, that are to be packaged with the theme, the following options exist:
  - Create a package that contains a compressed file containing the static resources that can be deployed to the file store and an EAR file containing a WAR file that contains the dynamic resources.
  - Create a package that contains an EAR file that contains one WAR file with the static resources and a second WAR file that contains the dynamic resources. This file can be deployed to the application server.

You can also package the artifacts that you created in a Portal Application Archive (PAA) file for the Solution Installer. The Solution Installer automatically performs the steps to deploy the artifacts.

Deploying theme components

How the theme components were packaged determines the method that is used to deploy themes. All options require an **XMLAccess** script and runtime configuration options or a PAA file.

Compressed file without EAR file
Use the **webdav-deploy-zip-file** command to deploy the compressed file and use the **XMLAccess** command to run the script. Use the WebSphere Integrated Solutions Console to add the runtime configuration.

Compressed file with EAR file
Use the **webdav-deploy-zip-file** command to deploy the compressed file. Use the WebSphere Integrated Solutions Console or scripts to deploy the EAR file. Use the **XMLAccess** command to run the script.

An EAR file with no compressed file
Use the WebSphere Integrated Solutions Console or scripts to deploy the EAR file, and use the **XMLAccess** command to run the script.

PAA file
The Solution Installer automatically detects the steps to perform deployment.

Packaging themes for deployment

You must repackage the static content as a WAR file, or an EAR file containing the WAR file when it is not possible to exchange the static content with the Operations Team as a separate compressed file.

Create a custom theme before starting this process. To create a theme based on the Version 8.5 theme see Create a copy of the theme.

You must create and collect the required code artifacts, the scripts to register the theme and skins, and the list of required runtime configuration changes. Create an EAR file that contains one WAR file containing the dynamic content of the theme and one WAR file containing the static content.

Theme is created based on the modularized portal theme.
Create a custom version of the dynamic content as a new EAR file and a custom version of the static content by creating a folder in the file store. For more information about copying a theme for customization, see Create a copy of the theme. No further customization was performed so far. Also an XML Access script is used to register the themes and skins that are part of the EAR file.

If you performed additional changes, like deleting files or changing file names, then you need to map the following steps accordingly.

Cache headers are not automatically set for static files served from a war file. External means, such as overwriting headers in the HTTP Server, must be used in this case. When you use this approach, complete the Base Portal Tuning - Web Server tuning chapter of the *Performance Tuning Guide* for optimal performance.

"Add static content to your custom theme" on page 2817
Add static content for you custom theme that you created based on one of the ready-to-use themes.

"Adapt the scripts that register the custom theme and skins" on page 2818
You must adapt the scripts that register the custom theme and skins that were moved from the file store.

"Adapting the list of required runtime configuration changes for your theme" on page 2819
You must adapt the list of required runtime configuration changes for your theme.

"Test the custom EAR file" on page 2819
Test the new custom EAR file on your test server to verify that it deploys successfully.

Add static content to your custom theme:

Add static content for you custom theme that you created based on one of the ready-to-use themes.

Procedure

1. Export the content from the file store. For more information, see *Exporting content from the file store*.
2. Export the files that are required by your custom theme to the following folders:
   - `themes`
   - `skins`
   - `layout-templates`
   - `common-resources`
3. Save the files to your disk.
4. Add a web module to your existing custom theme. For this example, an EAR file name `MyEar` was created for the existing custom theme. The EAR file contains a web module named `MyDynamicContent.war` that uses the context path `/MyDynamicContent`. If you created a WAR file instead, you must create an EAR file before you can proceed.
   a. Open your existing project, or import your existing EAR file as a project in the tool used to create the custom theme.
   b. Add a WAR file to the existing EAR file. In this sample, the name is `MyStaticContent.war` and the context root is `/MyStaticContent`.
   c. The `web.xml` file for the new web module can be as simple as the following example:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://java.sun.com/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
         http://java.sun.com/xml/ns/javaee/web-app_2_4.xsd"
         version="2.4">
    <display-name>StaticHTMLContent</display-name>
    </web-app>
```

The structure of the resulting EAR file looks like the following sample:

```
MyEar
    META-INF
        application.xml
    MyStaticContent.war
    MyDynamicContent.war
```

The following code is a sample of the `application.xml` file:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE
    application PUBLIC "-//Sun Microsystems, Inc.//DTD J2EE Application
    1.3//EN" "http://java.sun.com/dtd/application_1_3.dtd">
<application id="MyEar">
    <display-name>MyEar</display-name>
    <module id="MyStaticContent">
        <web>
            <web-uri>MyStaticContent.war</web-uri>
            <context-root>/MyStaticContent</context-root>
        </web>
    </module>
```

```
    <module id="MyDynamicContent">
        <web>
            <web-uri>MyDynamicContent.war</web-uri>
            <context-root>/MyDynamicContent</context-root>
        </web>
    </module>
</application>
```

5. Add the exported static content to the web module.
   a. Copy the static content that you exported in step 1 into the WAR file.
      The structure of the WAR file looks like the following sample:

```
MyStaticContent
  themes
  skins
  layout-templates
  common-resources
  WEB-INF
  web.xml
```

6. Make whitelist and black lists For security reasons, the WAR data source does not serve content until a special context parameter is set. This context parameter defines which files from your web module WebSphere Portal Express is able to serve.

   You must define a whitelist using a regular expression that matches the files that you want to make available. In addition, with a blacklist you remove certain entries from the set of files that are available in the whitelist. A blacklist is helpful if you want to serve a folder but not a certain file within that folder.

   The expressions are case-sensitive, for example `WEB-INF` is different from `Web-Inf`.

   The parameters are set in the `web.xml` file of the web module that contains the static theme content.

   Sample: Serve all files that are not part of the `WEB-INF` folder.

```
<web-app>
  ........
  <context-param>
  <description>A regular expression that defines which of the resources in the war file can
               be served by the portal war datasource.</description>
  <param-name>com.ibm.portal.whitelist</param-name>
  <param-value>.*</param-value>
  </context-param>
  <context-param>
  <description>A regular expression that defines which of the resources in the war file cannot
               be served by the portal war datasource.</description>
  <param-name>com.ibm.portal.blacklist</param-name>
  <param-value>WEB-INF/.*</param-value>
  </context-param>
  ....
</web-app>
```

Adapt the scripts that register the custom theme and skins:

You must adapt the scripts that register the custom theme and skins that were moved from the file store.

Procedure

1. Open the `theme` file and adapt the following properties of the theme by changing the code `dav:fs-type1` to `war`:

```
   <parameter name="com.ibm.portal.theme.template.ref" type="string" update="set">
<![CDATA[war:<context-root-static-war>/<customThemePath>/]]>
   </parameter>

   <parameter name="com.ibm.portal.layout.template.href" type="string" update="set">
<![CDATA[war:<context-root-static-war>/layout-templates/2ColumnEqual/]]>
   </parameter>
```

2. Adapt the following properties for all the skins moved from the file store to an EAR file by changing the code `dav:fs-type1` to `war` for the following parameters:

```
<parameter name="com.ibm.portal.skin.template.ref" type="string" update="set">
   <![CDATA[war:<context-root-static-war>/<customSkinPath>/skins/Hidden/]]>
</parameter>
```

Adapting the list of required runtime configuration changes for your theme:

You must adapt the list of required runtime configuration changes for your theme.

Procedure

1. Modify the dynamic resource references in static resources of your theme static resources. For more information about dynamic resource references, see Copying the theme.
2. Change the common-resources root node.
   a. Open the WebSphere Integrated Solutions Console.
   b. Open**Resources** > **Resource Environment** > **Resource Environment Providers** > **WP GlobalThemeConfig** > **Custom Properties**.
   c. Change **behaviors.layout.defaultLayout** from `dav:fs-type1/layout-templates/2ColumnEqual/` to `<context-root-static-war>/layout-templates/2ColumnEqual/`. For example, if your context root is `/MyStaticContent`, the value would be `war:MyStaticContent/layout-templates/2ColumnEqual/`.
3. Change **resources.commonResourcesRootURI** from `dav:fs-type1/common-resources` to `war:/common-resources.` For example, if your context root is `/MyStaticContent`, the value would be `war:MyStaticContent/common-resources`
   a. Open the WebSphere Integrated Solutions Console.
   b. Open**Resources** > **Resource Environment** > **Resource Environment Providers** > **WP ConfigService** > **Custom Properties**.
   c. Create or adapt a regular expression for the **refreshPageLayout.template.regexp** parameter. The portal default is `dav:fs-type1/layout-templates/.*|dav:fs-type1/themes/.*`. The following example shows a regular expression:

      ```
      dav:fs-type1/layout-templates/.*|dav:fs-type1/themes/.*|war:/themes/.*
      ```

      Note: If you do not perform the last step, users must have the **Markup Editor** role that is assigned to be able to change a layout.

What to do next

You can define the secure locations of layout templates by using this setting. If portal receives a request to create or update a page that contains a layout link, and that link is matching this regular expression, the markup editor role is not enforced on that layout. Include only locations in the regular expression that are under Access Control enforcement. For example, the layout templates and theme folders can be changed only by users that have the Theme Manager role. References to war files are also usable, as the war file deployment is secured by Java Platform, Enterprise Edition rights.

Test the custom EAR file:

Test the new custom EAR file on your test server to verify that it deploys successfully.

If you already deployed the dynamic WAR file, you must uninstall the WAR file before this step because it is redeployed in the ear file. Do not delete the theme in portal when you uninstall the WAR file.

Deploy the EAR file, run the XML Access scripts to register the theme and skins. Also, update the runtime configuration as described before.

Restart your portal server and test the new custom ear file on your test server to verify that it deploys successfully.

Exporting content from the filestore

You must export content from the filestore to create your custom theme. There are different options available to export files from the portal file store.

Exporting content from the filestore is required in different scenarios, however with different files to be exported. See the concrete scenario description for the detailed list of files to export.

You can access the filestore by using the following URL:

- http://<server>:<port>/wps/mycontenthandler/dav/fs-type1/

Use one of the following options to export the files stored there.

Get a compressed file using your browser

You can obtain a compressed file of the content in the filestore using your browser. Enter the following url in your browser:

```
http://<server>:<port>/wps/mycontenthandler/dav/fs-type1/<folder-name>/?mime-type=application/zip
```

Where:

- The <server> value is the host name of the portal.
- The <port> value is the port number for WebSphere Portal Express.
- The <folder-name> value is the folder to be compressed. This value is optional.

Note: A / must follow the folder name.

The URL triggers a download of a compressed file. If you are prompted for a user and password enter the admin user ID and password for WebSphere Portal Express. Store the file on the local file system.

This following url downloads the complete content of the filestore:

```
http://<server>:<port>/wps/mycontenthandler/dav/fs-type1/<folder-name>/?mime-type=application/zip
```

The following url downloads the content of the themes folder:

```
http://<server>:<port>/wps/mycontenthandler/dav/fs-type1/themes/?mime-type=application/zip
```

Automate the export using an Ant task

If you want to automate the export, you can write an Ant task as depicted in the following example:

```
<target name="export-mytheme">
    <get src="http://<server>:<port>/wps/mycontenthandler/dav/fs-type1/<folder-name>/?mime-type=application/zip"
        username="PortalAdminID"
        password="PortalAdminPwd"
        dest="/tmp/mytheme.zip" />
</target>
```

Where:

- The <server> value is the host name of the portal.
- The <port> value is the port number for WebSphere Portal Express.

- The <folder-name> value is the folder to be compressed. This value is optional.

Note: A / must follow the folder name.

Use WebDAV to connect to the filestore

Use a WebDav Client to connect to the filestore using the following url:

- http://<server>:<port>/wps/mycontenthandler/dav/fs-type1/

Browse to the folder you need and copy the files to your local drive.

Device classes

Device classes are used in IBM WebSphere Portal Express as an abstraction for common properties for the device of a client. For instance, tablet computers can be grouped into a device class `tablets`, since they share a form factor and possibly other traits such as touch interface, or additional hardware sensors.

Note: The abstraction provided by device classes does not make the defining properties explicit, but is rather indirectly done by the assignment of device classes to clients.

1. "Device classes overview"
   Device classes can be implemented to organize Clients in to groups. After you defined a device class, you can then assign it to clients. You can assign multiple device classes to a client to help scope your environment.
2. "Additional information about device classes for developers" on page 2822
   The `DeviceClass` profile attribute contains only the highest priority device class on the client. Highest priority is determined as the first device class listed for the client. `DeviceClassList` provides access to all device classes on a client, as a string of comma-separated values.
3. "mvc:URI scheme" on page 2823
   The `mvc:URI` scheme is a special URI format that accesses different resources, depending on the device class. This scheme is used by the Portal 8001 theme in the definition of several dynamic content spots.
4. "Creating and deleting device classes" on page 2824
   You can create and delete device classes using the XML configuration interface.
5. "Assigning device classes" on page 2824
   The process of assigning a device class to one of the supported clients in WebSphere Portal. Usually, when a certain device class needs to be supported, the first step is to create client definitions for each of the devices that belong to this class. Then, you create a device class and assign the device class to the clients. The assignment of device classes on clients is done in the Supported Clients administration section, using a specific capability on the client.
6. "Device class equations" on page 2825
   Device class equations are expressions that involve a mixture of device class operands and boolean logic operators.

Device classes overview

Device classes can be implemented to organize Clients in to groups. After you defined a device class, you can then assign it to clients. You can assign multiple device classes to a client to help scope your environment.

The following device classes are defined.

- `smartphone`

- tablet
- android
- blackberry
- iemobile
- ios
- worklight

Note: You can create extra device classes, or you can remove any of these device classes.

These are the device classes and Portal resources.

Clients
A client can be assigned a device class by assigning a capability. The name of the capability must start with `com.ibm.portal.devicesupport.deviceclass=` and continue with the administrative name of a device class, as provided with their definition.

Pages Device classes can be used to filter pages from the content model. For more information, see Filtering the content model.

Layout templates
A layout template for a static portal page can specify variants for device classes that use a naming convention. The layout template that matches the client's device class can then be used to render the page. For more information about how to define static pages and layout templates, see Creating a static page.

Theme modules
A contribution in a theme module can define subcontributions for a specific device class. For more information, see Defining theme modules.

Additional information about device classes for developers

The `DeviceClass` profile attribute contains only the highest priority device class on the client. Highest priority is determined as the first device class listed for the client. `DeviceClassList` provides access to all device classes on a client, as a string of comma-separated values.

EL Bean
The `clientProfile` bean can be used to obtain the list of device classes that are assigned to the client. For instance:`<c:set var="deviceClasses" scope="request" value="${wp.clientProfile['DeviceClass']}" />`

CC/PP Profile
The device class can also be directly obtained from the client profile information (CC/PP). For more information about the CC/PP API, see Client profile information (CC/PP) in portlets.

JavaScript
An array of device classes is available on the window object named `com_ibm_device_class`. This array object includes the same set of device classes that are defined for DeviceClassList within the CC/PP profile. You can query the array to check whether the environment is available. For example:

```
if(com_ibm_device_class.indexOf("smartphone") > -1){
 // action performed for smartphone devices
}else{
 // action performed for all non-smartphones
}
```

SPI Portal defines several services to obtain the list device classes that are assigned to the client. There is also a service to determine which device class matches best for a client. The package documentation for `com.ibm.portal.devicesupport` helps get you started.

URI schemes
The Portal theme provides and uses special URI formats that work with device classes.

mv: URI scheme
The `multiview` URI scheme provides a way to select the best matching resource for a client from an Atom listing of available resources that are based on naming conventions.

mvc:URI
The `multiview choice` URI scheme provides a way to select the best matching resource for a client from a listing of available resources that is directly contained in the scheme-specific part.

Related concepts:

Client profile information (CC/PP) in portlets
The Portal provides a standard API named "CC/PP" for accessing client profiles; learn how the client profile can be accessed through a request attribute. Learn about the attributes and components that are supported by the default profile implementation in the portal.

Related information:

Package com.ibm.portal.devicesupport

mvc:URI scheme

The `mvc:URI` scheme is a special URI format that accesses different resources, depending on the device class. This scheme is used by the Portal 8001 theme in the definition of several dynamic content spots.

The syntax of its scheme-specific part allows the following options:

- Specify the default URI to be used when no other listed URI matches.
- Map to an empty URI by using the syntax `...,name@,...`
- Create a comma-separated list of entries where:
  - The individual entries of the list are key and value pairs that are separated with '@'.
  - The keys represent one device class name or multiple device class names in equation form, where the equation can use '/' for OR, '+' for AND, '!' for NOT, and parentheses for grouping.
  - The value is a URI that must be properly encoded to not use any of the special characters that are described here. Therefore, certain values such as the comma must be double-encoded.

You can also use URIs with query parameters, for example `mvc:uri1?foo=bar &hugo=123,tablet@uri1_tablet%252ftoken1`. Note the use of double encoding to represent a comma as part of a resource URI `, = pct %2f, % = pct %25`.

The following examples demonstrate some of the possible combinations:

- `mvc:res:/hello.jsp`: Uses a single default URI.
- `mvc:res:/hello.jsp,smartphone@res:/hello_smartphone.jsp`: Uses `res:/hello.jsp` as the default URI and `res:/hello_smartphone.jsp` as the URI for smartphones.

- `mvc:res:/hello.jsp,smartphone/tablet@res:/hello_mobile.jsp`: Uses `res:/hello.jsp` as the default URI and `res:/hello_mobile.jsp` as the URI for smartphones and tablets.
- `mvc:res:/hello.jsp,smartphone@,tablet@res:/hello_tablet.jsp`: Uses `res:/hello.jsp` as the default URI and `res:/hello_tablet.jsp` as the URI for tablets. No URI is assigned for smartphones.
- `mvc:res:/hello.jsp,smartphone+ios@res:/hello_smartphone_ios.jsp,(smartphone/tablet)+android@res:/hello_mobile_android.jsp`: Uses `res:/hello.jsp` as the default URI, `res:/hello_smartphone_ios.jsp` as the URI for iOS smartphones, and `res:/hello_mobile_android.jsp` as the URI for Android smartphones and tablets.

Creating and deleting device classes

You can create and delete device classes using the XML configuration interface.

Create a device class

The following example creates a device class with the name `smartphone` and the unique name `wps.deviceclass.smartphone`.

```
<?xml version="1.0" encoding="UTF-8"?>
<request type="update" create-oids="true"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd">
 <portal action="locate">
  <device-class action="update" name="smartphone" uniquename="wps.deviceclass.smartphone" />
 </portal>
</request>
```

Delete a device class

The following example deletes the device class with the unique name `wps.deviceclass.smartphone`.

```
<?xml version="1.0" encoding="UTF-8"?>
<request type="update" create-oids="true"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd">
    <portal action="locate">
        <device-class action="delete" uniquename="wps.deviceclass.smartphone"/>
    </portal>
</request>
```

Assigning device classes

The process of assigning a device class to one of the supported clients in WebSphere Portal. Usually, when a certain device class needs to be supported, the first step is to create client definitions for each of the devices that belong to this class. Then, you create a device class and assign the device class to the clients. The assignment of device classes on clients is done in the Supported Clients administration section, using a specific capability on the client.

"Assigning a device class manually to a client" on page 2825
You can manually assign a device class to a client.

"XML sample for assigning a device class" on page 2825
This example creates a client that matches devices that send `iPhone` in the user agent. The device class `smartphone` and `ios` are assigned to the client.

Assigning a device class manually to a client:

You can manually assign a device class to a client.

About this task

In the Supported Clients administration section, add a specific capability on the client:

Procedure

To assign a client the device class `smartphone`, add the following capability: `com.ibm.portal.devicesupport.deviceclass=smartphone`.
You can add multiple device classes to a client, ensure that they are separated by a comma: `com.ibm.portal.devicesupport.deviceclass=smartphone,worklight,ios`.

Note: During the matching process between the client user agent and the client, only one client is chosen. If multiple clients match, the highest priority client is used. Only the device classes that are assigned to the highest priority client are available. The priority is set with the ordinal attribute on the client definition. The larger the value of the ordinal, the higher priority it has during the matching process. The lowest priority client has an ordinal set to zero.

XML sample for assigning a device class:

This example creates a client that matches devices that send `iPhone` in the user agent. The device class `smartphone` and `ios` are assigned to the client.

```
<?xml version="1.0" encoding="UTF-8"?>
<request type="update" create-oids="true"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd">
  <portal action="locate">
      <!--  iPhone -->
      <client action="update" manufacturer="Apple" markup="html"
        markup-version="" name="iPhone" ordinal="10000" uniquename="wps.client.iphone"
        version="7.0">
        <useragent-pattern>.*iPhone.*</useragent-pattern>
        <client-capability update="set">HTML_4_0</client-capability>
        <client-capability update="set">com.ibm.portal.devicesupport.deviceclass=smartphone
        </client-capability>
      </client>
  </portal>
</request>
```

Note: The ordinal attribute on the client definition sets the priority of the client during the matching process. Only one client is chosen and the device classes on that client are made available.

Device class equations

Device class equations are expressions that involve a mixture of device class operands and boolean logic operators.

Valid syntax includes the following operators:

- Parentheses
- The `NOT` operator that is represented by "`!`"
- The `AND` operator that is represented by "`+`"
- The `OR` operator that is represented by "`/`"

The order of operations when you parse an equation follows the normal boolean logic order of operations: parentheses, `NOT`, `AND`, `OR`.

For example:

```
android+smartphone
worklight+(ios/android)
(android/ios)+smartphone+!blackberry
```

This dynamic content spot only displays for android smartphones.

```
{
  "modules": [{
    "id" : "topnavoverlay",
    "prereqs": [{
      "id":"wp_dynamicContentSpots_85"
    }],
     "contributions": [{
      "type":"dyn-cs",
      "sub-contributions": [{
        "type":"markup",
        "ref-id":"85theme_topNav",
        "uris": [{
          "value":"res:/your/sample.html"
         "deviceClass": "android+smartphone"
        }]
      }]
    }]
  }]
}
```

Where to use device equations

Clients can have multiple device classes that are assigned in a comma separated list. These device classes on a client are then used in the device equations to determine what resources to provide or logic to run.

Device class equations are currently used with the Resource Aggregator. Subcontributions that use the device class attribute can now use equations to target resources to the device classes of the client that is accessing the resources.

MVC architecture for use in jsp's for loading dynamic spots that are based on client device classes.

Portal-if tag has a device class attribute that allows checking the client for a device equation.

Responsive Web Design

Responsive Web Design provides content parity between mobile devices and desktop channels, which enhances user experience and brand consistency. Seamless changes in screen size, from small to large, are now possible while the order of the content is maintained. Content maintenance is simplified by having one site that is represented by one set of assets.

"Mobile navigation" on page 2827
The IBM WebSphere Portal Express 8.5 ready-to-use theme provides two new responsive page navigation designs for mobile devices. One is aimed at smartphones, while the other is designed for tablets. The user agent for a device is parsed to determine which navigation to render on the portal page.

"Standard portal pages and mobile devices" on page 2831
IBM WebSphere Portal Express has two types of pages, Standard Portal Pages and Static Pages. Static pages use HTML templates to organize layout

containers and controls, and are used as the default page type since version 7.0. Standard Portal pages use table-based layouts.

"Relative width CSS classes for theme layouts" on page 2831
To assist with making the elements of a page responsive to various widths, relative width CSS classes have been added to the theme layouts. These classes are marker classes only. There are no styles that are defined for these classes by default. They are there on the containers, rows and columns, of the layouts so that you can define your own styles that are associated with the classes as needed to make your page elements responsive to various widths.

Mobile navigation

The IBM WebSphere Portal Express 8.5 ready-to-use theme provides two new responsive page navigation designs for mobile devices. One is aimed at smartphones, while the other is designed for tablets. The user agent for a device is parsed to determine which navigation to render on the portal page.

Smartphone navigation design

Figure 32. Welcome page displayed on a smartphone

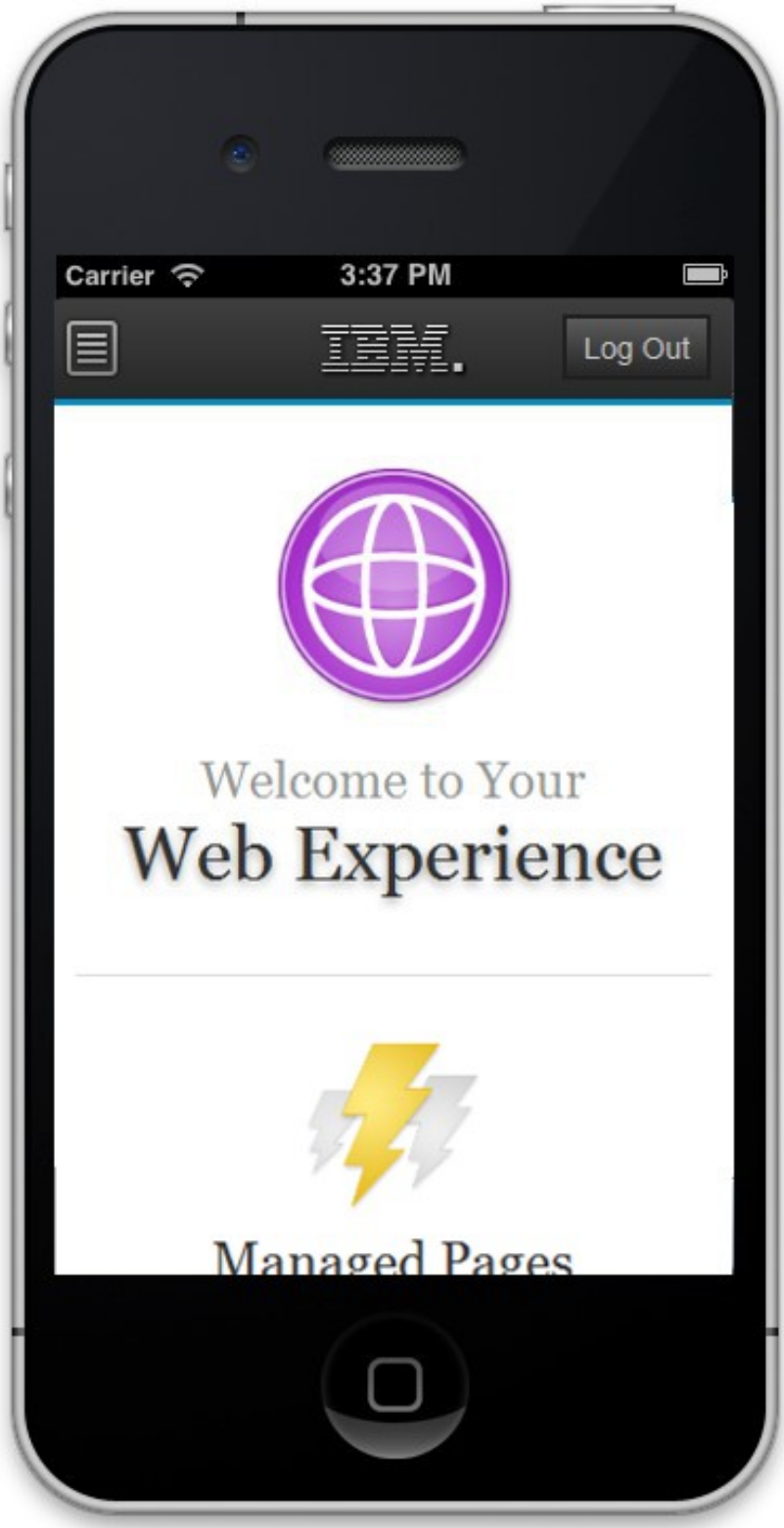
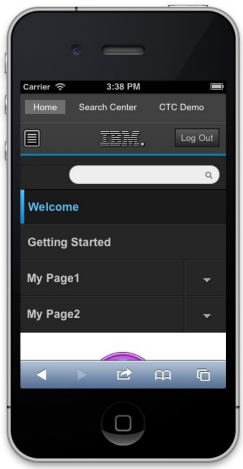


Figure 33. Mobile navigation displayed on a smartphone.



1. Scroll up to display the navigation toggle: When the portal loads, this level of navigation is hidden to maximize the limited real estate on a smartphone.
2. Tap the navigation button to show the navigation for your portal.
3. Tap this link to return to the portal home page.
4. Tap the name of the page to load a specific page.
5. Tap the arrow near the page name to display the child pages for that page.

Tablet navigation design

Figure 34. Welcome page displayed on a tablet.

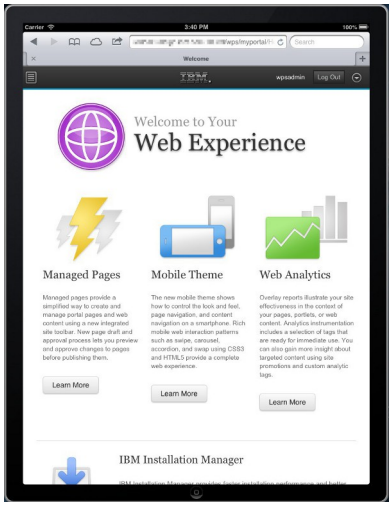
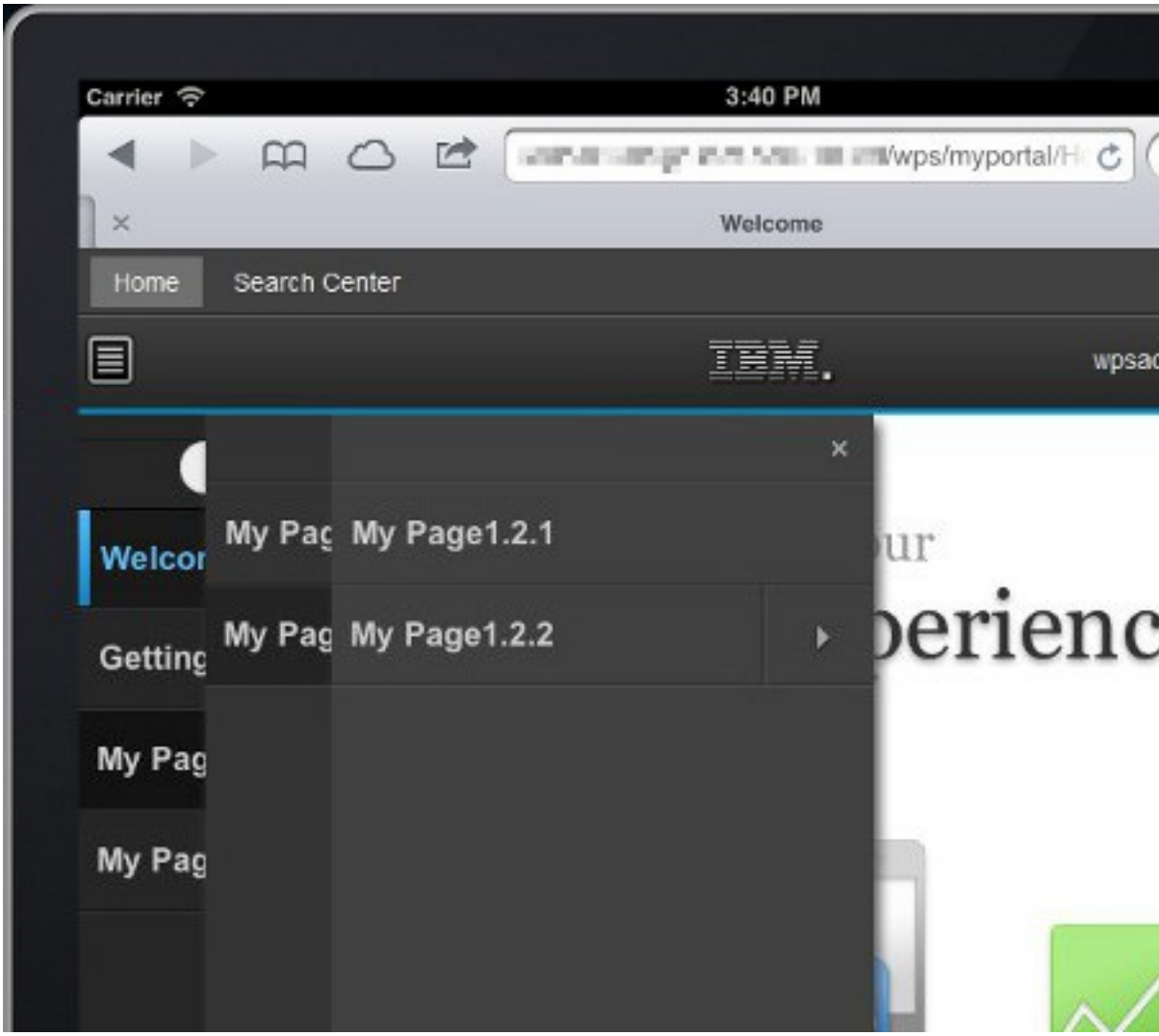


Figure 35. Mobile navigation displayed on a tablet.



1. Tap the navigation button to show the navigation for your portal.
2. Tap this link to return to your portal home page.
3. Tap the arrow button to show the navigation.
4. Tap the name of the page to open a specific page.
5. Tap the arrow near the page name to display the child pages for that page.

You can have the navigation and portal navigation open individually or at the same time without affecting the functionality of the portal.

Mobile navigation markup

The mobile navigation markup for both smartphone and tablet is created by the `mobileNavigation.jsp` file of the theme, found at *PortalServer_root*`/theme/wp.theme.themes/default85/installedApps/DefaultTheme85.ear/DefaultTheme85.war/themes/html/dynamicSpots`. The `mobileNavigation.jsp` file is controlled by the mobile navigation dynamic spot in `theme.html`: `dyn-cs:id:85theme_mobileNavigation`

The navigation on mobile devices is rendered by the `mobileNavigation.jsp` file. Therefore, the primary, secondary, breadcrumb, and side navigation that is used on the desktop do not produce any output for a mobile device. The navigation is rendered for mobile, but is hidden when the page loads. On a tablet, the arrow

button in the banner can be tapped to reveal the navigation. On a smartphone, the user can scroll up to see the first-level navigation pages. Since smartphones have little real estate, site designers might want to hide certain first-level pages, such as Administration, for these devices. Add the `com.ibm.portal.mobile.Hidden` metadata to the page to hide certain first-level pages. By default, the Administration, Application, and Tag Center pages are hidden.

Note: If the expanded navigation for your Portal site ends up being long on a smartphone, it is possible part of the navigation can get cut off. This cut off happens because a maximum height is required to be set on the navigation in order for the CSS3 animations to work correctly. If your navigation does not fit into this maximum height, it can be adjusted by editing the `mobilenav.css` file of your custom theme in WebDAV at `dav:fs-type1/themes/myCustomTheme/css/`. Look for the following style declaration, increase the max-height, and save.

```
.wpthemeMobileNav ul.wpthemeExpandNav {
   /* navigation lists */
   max-height: 100em;
}
```

Standard portal pages and mobile devices

IBM WebSphere Portal Express has two types of pages, Standard Portal Pages and Static Pages. Static pages use HTML templates to organize layout containers and controls, and are used as the default page type since version 7.0. Standard Portal pages use table-based layouts.

Since standard pages are table-based, it is more difficult to implement responsive web design on standard pages. Limited support for responsive web design is implemented for standard pages on smartphone devices. When a standard page is viewed on a smartphone, all of its layout containers and portlets stack vertically. Stacking the containers and portlets ensures that the page contents fit better on the screen. The look of standard pages remains unchanged on desktop and tablet devices, which have more available real estate.

If you want to introduce more responsiveness to the standard page layouts of your portal site, use CSS selectors for pinpointing containers and controls to achieve the preferred designs. To add this support to your theme, add the `wp_legacy_layouts` module to your theme profile. For more information, see *Adding or removing a ready-to-use module to a theme*.

For example, if you want to hide the third column of a three column layout on a tablet sized device, you can target the screen size by using a media query and the third column with an appropriate CSS selector:

```
@media (max-width: 800px) {
    table.layoutRow tbody tr td:nth-child(3) {
        display: none;
    }
}
```

Related information:

CSS Selector Reference

Relative width CSS classes for theme layouts

To assist with making the elements of a page responsive to various widths, relative width CSS classes have been added to the theme layouts. These classes are marker classes only. There are no styles that are defined for these classes by default. They are there on the containers, rows and columns, of the layouts so that you can

define your own styles that are associated with the classes as needed to make your page elements responsive to various widths.

There are two types of relative width classes, and one of each is used on each layout container.

Semantic type, one of:

wpthemeThin

wpthemeNarrow

wpthemeMedium

wpthemeWide

wpthemeFull

These types allow for up to five variations in width and are described by using common width words, such as narrow and wide.

Grid type, one of:

wpthemeCol1of12

wpthemeCol2of12

wpthemeCol1of5

wpthemeCol3of12

wpthemeCol4of12

wpthemeCol2of5

wpthemeCol5of12

wpthemeCol6of12

wpthemeCol7of12

wpthemeCol3of5

wpthemeCol8of12

wpthemeCol9of12

wpthemeCol4of5

wpthemeCol10of12

wpthemeCol11of12

wpthemeCol5of5

wpthemeCol12of12

These types allow for up to 16 variations in width and are described by using a numbering system of nofn width units. At first glance it looks like a simple numbering system where a layout with five columns would use `wpthemeCol1of5`, `wpthemeCol2of5`, `wpthemeCol3of5`, `wpthemeCol4of5`, and `wpthemeCol5of5` to number the columns. However, the best way to look at this system is not as the number of the column, but rather as the number of width units that the column takes up on the page. So, `wpthemeCol2of5` means that the column takes up two of five width units of the page, or `2/5ths` of the width of the page, and `wpthemeCol8of12` means that the column takes up 8 of 12 width units of the page, or `3/4ths` of the width of the page. So, a layout with five columns of equal width would use `wpthemeCol1of5` on all of the columns.

The two class types map to each other in the following way:

Table 450. Relationship of the 2 relative width class types

Width of the page	Semantic Type	Grid Type
1/12th	wpthemeThin	wpthemeCol1of12
2/12ths (1/6th)	wpthemeThin	wpthemeCol2of12
1/5th	wpthemeThin	wpthemeCol1of5
3/12ths (1/4th)	wpthemeNarrow	wpthemeCol3of12
4/12ths (1/3rd)	wpthemeNarrow	wpthemeCol4of12
2/5ths	wpthemeNarrow	wpthemeCol2of5
5/12ths	wpthemeMedium	wpthemeCol5of12
6/12ths (1/2)	wpthemeMedium	wpthemeCol6of12
7/12ths	wpthemeMedium	wpthemeCol7of12
3/5ths	wpthemeMedium	wpthemeCol3of5
8/12ths (2/3rds)	wpthemeMedium	wpthemeCol8of12
9/12ths (3/4ths)	wpthemeWide	wpthemeCol9of12
4/5ths	wpthemeWide	wpthemeCol4of5
10/12ths (5/6ths)	wpthemeWide	wpthemeCol10of12
11/12ths	wpthemeWide	wpthemeCol11of12
5/5ths (1)	wpthemeFull	wpthemeCol5of5
12/12ths (1)	wpthemeFull	wpthemeCol12of12

The Semantic types maps to the following page width amount:

Table 451. Semantic type mapping to page width amounts

Semantic type	Amount of the page width
wpthemeThin	1/12th to 1/5th
wpthemeNarrow	1/4th to 2/5ths
wpthemeMedium	5/12ths to 2/3rds
wpthemeWide	3/4ths to 11/12ths
wpthemeFull	all

"Creating your own layout"
When you create your own `layout.html` files, it is important to apply one semantic type and one grid type relative width class to each container. Such as `wpthemeMedium` and `wpthemeCol6of12`. Applying both ensures that those creating portlets and other page elements do not have to define styles for both types. Along with neither type and both types, they also have the choice to define styles for just one of the types, depending on their needs of granularity.

"Creating portlets and page elements" on page 2834
When you create portlets and the page elements within them, you can now define your own style overrides as needed. Use these relative width CSS classes to make your page elements responsive to various widths.

Creating your own layout:

When you create your own `layout.html` files, it is important to apply one semantic type and one grid type relative width class to each container. Such as

wpthemeMedium and wpthemeCol6of12. Applying both ensures that those creating portlets and other page elements do not have to define styles for both types. Along with neither type and both types, they also have the choice to define styles for just one of the types, depending on their needs of granularity.

When you apply one of each type, it is also important to apply the properly matching ones. For example, it is important to use either wpthemeCol3of12, wpthemeCol4of12 or wpthemeCol2of5 with wpthemeNarrow. Never add one of the other mismatching ones, such as wpthemeCol9of12 with wpthemeNarrow.

Creating portlets and page elements:

When you create portlets and the page elements within them, you can now define your own style overrides as needed. Use these relative width CSS classes to make your page elements responsive to various widths.

Your portlet can be placed in a column of any width and run on a device of any screen width. The primary styles for the elements use elastic percentage widths rather than fixed widths in general to be responsive. For certain width columns, you might need to make additional overrides. For example, if something is too large to display correctly in a narrow column, you can add display:none to hide it entirely. You can also add font-size:1.5em for wide columns or font-size:0.8em for narrow columns. To load a smaller version of an image use background-image:url("myImage_small.jpg") for narrow columns.

If you care only about five levels of variation in width or less, then you need only to override for the semantic types. The following example shows three levels of variation:

```
/* Narrow width columns */
.wpthemeThin .myComponent, .wpthemeNarrow .myComponent {
    ...
}
/* Medium width columns */
.wpthemeMedium .myComponent, {
    ...
}
/* Wide width columns */
.wpthemeWide .myComponent, .wpthemeFull .myComponent {
    ...
}

If you care about more than 5 levels up to 16 levels of variation in width, then you only have to ov

/* 1/12th, 1/6th and 1/5th width columns */
.wpthemeCol1of12 .myComponent, .wpthemeCol2of12 .myComponent, .wpthemeCol1of5 .myComponent {
    ...
}
/* 1/4th and 1/3rd width columns */
.wpthemeCol3of12 .myComponent, .wpthemeCol4of12 .myComponent {
    ...
}
/* 2/5ths and 5/12ths width columns */
.wpthemeCol2of5 .myComponent, .wpthemeCol5of12 .myComponent {
    ...
}
/* 1/2 and 7/12ths width columns */
.wpthemeCol6of12 .myComponent, .wpthemeCol7of12 .myComponent {
    ...
}
/* 3/5ths and 2/3rds width columns */
.wpthemeCol3of5 .myComponent, .wpthemeCol8of12 .myComponent {
```

```
    ...
}
/* 3/4ths and 4/5ths width columns */
.wpthemeCol9of12 .myComponent, .wpthemeCol4of5 .myComponent {
    ...
}
/* 5/6ths and 11/12ths width columns */
.wpthemeCol10of12 .myComponent, .wpthemeCol11of12 .myComponent {
    ...
}
/* 5/5ths and 12/12ths (full) width columns */
.wpthemeCol5of5 .myComponent, .wpthemeCol12of12 .myComponent {
    ...
}
```

You can override for both types if you choose, but it is only necessary if you created a layout that does not use both types on all containers.

URL generation in WebSphere Portal

Generating Portal URLs correctly is one of the most important tasks in programming a WebSphere Portal Express based application. There are several programming tools and techniques available for generating WebSphere Portal Express URLs in custom code. The following section introduces the programming tools available and discusses when it is most appropriate to use each of the tools.

Types of Portal Urls

There are several different types of Portal URLs depending on what task you are trying to accomplish.

Render URLs
: This type of URL is used for retrieving a general view of a Portal page. It specifically does not include any portlet actions or cause any server-side state changes. A Render URL corresponds to an HTTP GET operation and is idempotent, that is, it can be run more than once without any harm. Normal WebSphere Portal Express page navigation is made up of render URLs.

Action URLs
: Action URLs are used for activities within portlets. The URLs correspond to HTTP POST or PUT and are often non-idempotent, meaning they must be run at most once. An Action URL typically targets a specific portlet, and might cause server-side state changes. The portlet action and the portlet at which the action is targeted are carried as parameters within the Navigation State document.

Friendly URLs
: Friendly URLs have human-readable strings in the URL that describe the path to a Portal page. These human-readable strings correspond to the Friendly URL Names that are associated with the pages or labels. In addition, there might also be Friendly Content Path tokens in the URL. The Friendly Content Path tokens are human-readable strings that describe the site area path to Web Content Management library associated with the page.

  Note: A friendly URL might also include an encoded Navigational State document. If it does not, it is a Stateless Friendly URL. There is a programming API specifically for working with Friendly URLs.

Vanity URLs

Vanity URLs are similar to Stateless Friendly URLs, in that they are human-readable and do not have an encoded Navigational State document. However, Vanity URLs are not tied to the Friendly URL Names associated with the Portal pages. Instead, Vanity URLs are intended to be aliases that are simple, easily remembered, and easily entered by hand if necessary. Vanity URLs are similar to Mapped URLs that were introduced in prior releases of WebSphere Portal. They are intended only as an initial entry point, and are not persistent in the browser address bar after interaction with the Portal site begins. There is a programming API specifically for working with Vanity URLs.

Piece-of-Content URLs

Piece-of-content URLs or PoC URLs are late binding mechanism that targets content instead of Portal artifacts such as pages. They use a different URL entry point into WeSphere Portal (typically *mypoc* or *mycontenthandler* instead of *myportal*). A programming API is available for working with Piece-of-Content URLs.

Methods for generating portal URLs

The complexity of a WebSphere Portal Express URL makes them difficult to hand-code, therefore do not try to build Portal URLs by string concatenation. The design intention is that most self-referential URLs in Portal are generated in code at run time to avoid broken links and to avoid manually maintaining links within a Portal-based site.

WebSphere Portal Express makes several options available to a programmer for generating these complex Portal URLs. These different options are designed to address different use-cases, from the simplest to the most complex.

- Portal JSP tags method is used in Theme and Skin JSPs.
- JSR 286 Portlet API and corresponding JSP tags. This method of URL generation addresses almost all URL generation requirements within a standard portlet.
  - Or if necessary when you modify an existing portlet, and upgrading is not a possibility, the older JSR 168 Portlet API is used.
  - The IBM Portlet API is no longer supported. Older portlets that are written to this API must be migrated to the current standard.
- CF05 WebSphere Portal defined public render parameters. This method can support many use cases that previously required the use of the Navigational State API.
- Friendly URL API method is specifically for use cases that involve Friendly URLs, including URLs that must be stateless (have no encoded Navigational State document).
- PoC URL API method is specifically for creating Piece-of-Content URLs.
- Vanity URL API method is specifically for working with Vanity URLs.
- Navigational State API method is the most full-featured and general programming tool for URL generation, but requires the most in-depth understanding and programming skill.

When you create cooperating portlets that require inter-portlet communication, the inter-portlet messaging might be carried in the URLs that are generated. Render parameters as supported by JSR 286 are one way of accomplishing this, but additional programming tools are also provided. JSR 286 techniques for cooperative portlets, and the additional tools are described in more detail in the

Portlet communication section. One example of such a tool is the Cooperative Portlet API for interoperability between JSR 286 and JSR 168 portlets.

Mapping use cases to available programming tools

Creating page navigation links between Portal pages at the theme level. For example, standard tabbed pages navigation.

These URLs are typically simple render URLs. In JSPs, use the *<portal-navigation/>* JSP tags. For more information about programming a Portal theme, see *Developing Themes and Skins* section.

A JSR 286 portlet, self-contained (no inter-portlet communication required), generating action URLs to itself and setting its own render parameters.

- JSR 286 defines a tag library for use in JSPs. This tag library is greatly expanded over the V1.0/JSR168 version. It has its own namespace to avoid collisions with the V1.0 library.

  ```
  <%@ taglib uri="http://java.sun.com/portlet_2_0"
  Prefix="portlet"%>
  ```

  Examples:

  – To generate a normal rendering URL with a render parameter:

  ```
  <a href="<portlet:renderURL>
  <portlet:param name="myRenderParameter"
  value="someValue"/>
  This is the text for the link</a>
  ```

  – To generate a portlet Action URL, which targets the current portlet on the current page:

  ```
  form method="POST"
  action="<portlet:actionURL/>">
  ```

  Note: An action URL must have the Navigational State document encoded on it.

- To do the equivalent in java code instead of using the JSP tags, use the JSR 286 Portlet API, which is in package javax.portlet.* For more information, see the related link *Package javax.portlet.* In particular, see the RenderResponse interface. A RenderResponse is passed to the portlet's render method by the portlet container. RenderPesponse implements MimeResponse, which provides 3 methods for creating self-referential URLs:
  – createRenderURL()
  – createActionURL()
  – createResourceURL()

A JSR 286 portlet that requires inter-portlet communication with another JSR 286 portlet, but no page navigation (Portal view remains on the current page).

It might be sufficient to use the JSR 286 render parameter support. However, other techniques also exist. For more information, see the *Portlet communication* section.

A JSR 286 portlet that needs to interoperate with a JSR 168 portlet.

A JSR168 portlet that worked with other portlets through inter-portlet communication is written to use the Cooperative Portlet API, also known as the Property broker. The Property broker was an IBM extension to the JSR 168 specification. JSR286 introduced the portlet event model, which superseded the Property Broker. JSR286 portlets and JSR 168 portlets can

interoperate if certain conditions are met. For more information, see the *Interoperability between JSR 286 portlet events and JSR 168 cooperative portlets* section.

A JSR 286 portlet that needs to perform the following tasks:

- Generate a render link to another Portal page (causes page navigation).
- Optionally control the mode or window state of the targeted portlet.
- Optionally control the Portal state.
- Optionally control the render parameters of the targeted portlet.
- Optionally control the locale for the generated request.

CF05 Starting with Portal 8.5 CF05, use the WebSphere Portal-defined public render parameters for all of these use cases, and more. These render parameters make various aspects of the current request context available as normal public render parameters in a WebSphere Portal specified namespace.

To read Navigational State values as render parameters:

These special public render parameters can be accessed through the normal JSR286 API `renderResponse.getParameter(<parameter qualified name>)`.

Where `<parameter qualified name>` is the concatenation of the NAMESPACE_URI and one of the render parameter names, as defined in the following section. For example, `http://www.ibm.com/xmlns/prod/websphere/portal/publicparams` selection.

To set Navigational State values by using render parameters:

They can be set by the normal JSR286 portlet API `baseURL.setParameter(<parameter qualified name>, <value>)`

By setting these specific render parameters by using the normal JSR286 APIs, the system encodes the correct navigational state settings into the Navigational State document on the new URL. This setting makes the new values available on the next request that results from rendering and clicking that new URL.

Notes:

- This support requires the use of stateful URLs because the render parameters are only carried in the encoded Navigation State document.
- All these values are available as public constants in the `com.ibm.portal.PublicRenderParameters` class, which is a public API within the `wp.model.api` component. For complete details, see the WebSphere Portal javadoc *IBM WebSphere Portal Express, Version 8.5.0.0 SPI Specification.*

The namespace of the WebSphere Portal Express specified public render parameter is `http://www.ibm.com/xmlns/prod/websphere/portal/publicparams` (available as NAMESPACE_URI). The following items are the special public render parameters that WebSphere Portal Express supports:

- **`selection (NAME_SELECTION)`**
- **`uri (NAME_URI)`**
- **`parameters (NAME_PARAMETERS)`**
- **`locale (NAME_LOCALE)`**
- **`themeTemplate (NAME_THEME_TEMPLATE)`**

- **labelMappings (NAME_LABEL_MAPPINGS)**
- **screenTemplate (NAME_SCREEN_TEMPLATE)**
- **themePolicy (NAME_THEME_POLICY)**
- **solo (NAME_SOLO)**
- **showTools (NAME_SHOW_TOOLS)**
- **hiddenContent (NAME_HIDDEN_CONTENT)**
- **hiddenPages (NAME_HIDDEN_PAGES)**
- **statePartition (NAME_STATE_PARTITION)**
- **path-info (NAME_PATH_INFO)**
- **focus (NAME_FOCUS)**
- **deviceClass (NAME_DEVICE_CLASS)**
- **digest (NAME_DIGEST)**
- **pageMode (NAME_PAGE_MODE)**
- **editMode (NAME_PAGE_EDIT_MODE)**
- **infoMode (NAME_PAGE_INFO_MODE)**
- **helpMode (NAME_PAGE_HELP_MODE)**

There also exists a Portal URL Generation Convenience API. This convenience API supports render URLs only. No state changes (action URLs) can be generated by using this API. The following items are the key classes in this API:

- `com.ibm.portal.portlet.service.url.PortalURLGenerationService`
- `com.ibm.portal.portlet.service.url.PortalURLWriter`

Per the javadoc of the *com.ibm.portal.portlet.service.url* package, a programmer:

1. Obtains an instance of a `PortletServiceHome` by JNDI lookup.
2. On that home interface, calls `getPortletService(PortalURLGenerationService.class)`.
3. On that service, calls `getPortalURLWriter(request, response)`
4. Uses the methods on the `PortalURLWriter` object to create and manipulate the render URL.

An equivalent set of JSP tags for the URL Generation Convenience API exists.

```
<%@ taglib uri="http://www.ibm.com/xmlns/prod/websphere/portal/v8/portlet/ibm-portlet-ext"
```

For more information, see *JSP tags for standard portlets*. An example of creating a render URL by using the Convenience API tag is`<portlet-ext:portalRenderURL>`.

Friendly URL

A Friendly URL also known as a Friendly URL Name is a human-readable name for a Portal page. It is set as an attribute of the page, and each page can have at most one Friendly URL.

Notes:

- If a page has a correctly configured Friendly URL, then WebSphere Portal Express guarantees that any request which renders the page has the Friendly URL. The request does not have the Friendly URL only if the Friendly URL enforcement is explicitly turned off by using the `friendly.redirect.enabled` configuration setting.

- Friendly URLs do not guarantee that they are free of Navigational State. If it is necessary to remove Navigational State, further techniques are available. You might also want to explore *Vanity URLs* or *URL Mappings* for these use cases.

To use the Friendly URL API, you need to get a `FriendlyURLFactory` instance. Depending on the type of code you are writing, you can get a `FriendlyURLFactory` instance in one of two ways:

If you are writing a portlet:

- Start with the `com.ibm.portal.resolver.friendly.service.PortletFriendlySelectionServiceHo` class. You need to obtain an instance of that class in 2 steps:
  1. Do a JNDI lookup by using the constant `PortletFriendlySelectionServiceHome.JNDI_NAME`. This JNDI lookup returns an instance of a `PortletServiceHome`.
  2. On the `PortletServiceHome` object, call `getPortletService(PortletFriendlySelectionServiceHome.class)`. The call returns a `PortletFriendlySelectionServiceHome` instance.
- For efficiency, do the JNDI lookup and `getPortletService` call within the `init(PortletConfig)` method of your portlet, and save the `PortletFriendlySelectionsServiceHome` instance to be reused. It is not necessary to do these calls on a per-request basis.
- On a per-request call, such as the `doView(RenderRequest,RenderResponse)` method of the portlet, use the saved `PortletFriendlySelectionServiceHome` to call `getPortletFriendlySelectionService(request,response)`. The call returns a `PortletFriendlySelectionService` instance.

  Note: This instance must be disposed by calling `dispose()` before it goes out of scope.
- On the returned `PortletFriendlySelectionService` instance, call `getURLFactory()`.

If you are writing theme code:

- Start with the `com.ibm.portal.resolver.friendly.service.PortalFriendlySelectionServiceHom` class. Do JNDI lookup by using the constant `PortalFriendlySelectionServiceHome.JNDI_NAME`. This lookup returns an instance of a `PortalFriendlySelectionServiceHome`.
- For efficiency, do the JNDI lookup within the constructor or other initialization method of your class, and save the `PortalFriendlySelectionServiceHome` instance to be reused. It is not necessary to do this call on a per-request basis.
- On a per-request call, use the saved `PortalFriendlySelectionServiceHome` to call `getPortalFriendlySelectionsService(HttpServletRequest,HttpServletResponse)` The call returns a `PortalFriendlySelectionService` instance.

  Note: This instance must be disposed by calling `dispose()` before it goes out of scope.
- On the returned `PortalFriendlySelectionService` instance, call `getURLFactory()`.

After you have a FriendlyURLFactory, you can call one of the newURL() methods to obtain a FriendlyURL instance. A FriendlyURL instance can be set up using the set* methods, which are written to the response by using the writer(Writer) method. The Writer is obtained from the response, and then is disposed by calling dispose().

Vanity URL

A Vanity URL is a simple, easy to remember URL that a user can enter by hand. Vanity URLs are managed by the webmaster with the WebSphere Portal Express administrative tools, such as the toolbar, administrative portlets, or XMLAccess scripting. However, sometimes when you render a response, it is necessary to produce a Vanity URL link.

The following section describes how to use the Vanity URL API to obtain a VanityURLNode, which can be used to render a Vanity URL link. Depending on the type of code package you are developing, use one of the 3 different ways to access and use the Vanity URL API.

Note: Be careful to select the correct SPI package for the code that is being developed.
The necessary interface documentation is in the WebSphere Portal Express SPI javadoc.

If you are writing a portlet:

- Locate the com.ibm.portal.portlet.service.model package.
- Start with the VanityURLModelProvider interface. You need to obtain an instance of a class, which implements that interface, in 2 steps:
  1. Obtain an instance of a PortletServiceHome by JNDI lookup by using the name constant VanityURLModelProvider.JNDI_NAME.
  2. On that PortletServiceHome object, call getPortletService(VanityURLModelProvider.class).
- Do the JNDI lookup and call getPortletService in the init() method of the portlet. Save the returned VanityURLModelProvider instance in a static field in your portlet class. This provider instance can be reused across requests.
- For each request, for example in the call to the doView() method of the portlet, call getVanityURLModel on the saved VanityURLModelProvider instance, passing the current portlet request and response. Models must not be reused across requests.
- On the VanityURLModel object, call getLocator().
- On the VanityURLModelLocator, you can call any of the findBy methods, which return either a single VanityURLNode or an IterableListModel<VanityURLNode> list.

If you are writing theme code:

- Locate the com.ibm.portal.model package.

  Note: Many of the class names in this set of instructions are identical to the class names for DataSource usage of the Vanity URL API, but the package name is different.
- Obtain an instance of a class, which implements VanityURLModelProvider, in 2 steps:

1. Obtain an instance of a VanityURLModelHome by JNDI lookup by using the name constant VanityURLModelHome.JNDI_NAME. See the com.ibm.portal.model.VanityURLModelHome javadoc for example code.
2. On that VanityURLModelHome object, call getVanityURLModelProvider().

- Do the JNDI lookup and call getVanityURLModelProvider() in the constructor or an init() method of your class. Save the returned VanityURLModelProvider instance in a static field in your class. This provider instance can be reused across requests.
- For each request, call getVanityURLModel on the saved VanityURLModelProvider instance, passing the current servlet request and response. Models must not be reused across requests.
- On the VanityURLModel object, call getLocator().
- On the VanityURLModelLocator, you can call any of the findBy methods, which return either a single VanityURLNode or an IterableListModel<VanityURLNode> list.

If you are writing a DataSource:

- Locate the com.ibm.portal.cor.service package.

  Note: Many of the class names in this set of instructions are identical to the class names for the theme code usage of the Vanity URL API, but the package name is different.
- Obtain an instance of a class, which implements VanityURLModelProvider, in 2 steps:
  1. Obtain an instance of a VanityURLModelHome by JNDI lookup by using the name constant VanityURLModelHome.JNDI_NAME. See the com.ibm.portal.cor.service.VanityURLModelHome javadoc for example code.
  2. On that VanityURLModelHome object, call getVanityURLModelProvider().
- Do the JNDI lookup and call getVanityURLModelProvider() in the constructor or an init method of your class, and save the returned VanityURLModelProvider instance in a static field in your class. This provider instance can be reused across requests.
- In a per-request method of the DataSource, call modelProvider.getVanityURLModel(com.ibm.content.operations.registry.api.Co

  Programming detail for a DataSource: In most cases, a DataSource is allocated by a dedicated factory. The factory is registered with the resolver framework to handle requests for a specific URI. When a request for that URI is received, the factory is started by the resolver framework and is passed a com.ibm.content.operations.registry.api.Context object. Make sure to understand the difference between this COR Context object and the javax.naming.Context class that is used by JNDI lookup. The factory allocates a new DataSource instance (or might retrieve an existing one from a pool) and calls reset() on the DataSource, passing the COR Context object. This object and the other parameters to reset(), must be saved in instance fields within the DataSource. They can be used on subsequent method calls until either dispose() is called or reset() is called again.

Use the COR Context object on the call to getVanityURLModel(com.ibm.content.operations.registry.api.Context).

- On the VanityURLModel object, call getLocator().
- On the VanityURLModelLocator, you can call any of the findBy methods, which return either a single VanityURLNode or an IterableListModel<VanityURLNode> list.

After you have a VanityURLNode instance through any of the appropriate ways, you can call the various methods of that interface to build a rendered representation of the VanityURL:

- VanityURLnode.isSecure() tells whether the target protocol is http or https.
- VanityURLnode.isProtected() tells whether the target URL mapping is to /portal or /myportal.
- VanityURLnode.getHost() returns the host name that is associated with this Vanity URL.
- VanityURLnode.getPort() returns the port number that is associated with this Vanity URL.
- VanityURLnode.getVpld() returns the Virtual Portal OID for this Vanity URL.
- VanityURLnode.getPath() returns the path string for this Vanity URL. This path string is what is thought of as the Vanity URL string.

Create, update, or delete Vanity URLs.

If you are writing code, which is intended to create, update, or delete Vanity URLs, rather than read and render them, use the following APIs:

- Locate the com.ibm.portal.model.controller package.
- You need to obtain an instance of a class, which implements VanityURLModelControllerHome through JNDI lookup by using the name constant VanityURLModelControllerHome.JNDI_Name. See the com.ibm.portal.model.controller.VanityURLModelControllerHome javadoc for example code.
- On that VanityURLModelControllerHome object, call getVanityURLModelControllerProvider().
- Do the JNDI lookup and call getVanityURLModelControllerProvider() in the constructor or an init method of your class, and save the returned VanityURLModelControllerProvider instance in a static field in your class. This provider instance can be reused across requests.
- You might also need to obtain a VanityURLModelProvider, by using the same code as described in the writing Theme code.
- In a per-request method,
  - On the VanityURLModelProvider, call getVanityURLModel(request,response).
  - On the VanityURLModelControllerProvider object, call createVanityURLModelController(model) passing the model that is returned from the call to getVanityURLModel.
- The subsequent calls to create the new Vanity URL and set its attributes are documented in the javadoc for VanityURLModelController and ModifiableVanityURLNode.

A JSR 286 portlet that needs to generate an action URL to a second specific portlet, or any other use case that is not listed here.
Navigational State API

Piece-of-Content URL

A Piece-of-Content or POC URL is a URL that targets a DataSource or a ResolutionService within the Resolver framework. Given such a DataSource or ResolutionService, the PoC URL API assists the programmer to create a URL that results in the Resolver framework to start the correct DataSource or ResolutionService.

To work with Piece-of-Content URLs, obtain an instance of a `com.ibm.portal.resolver.acessors.url.PocURLFactory`. Much like other URL APIs, the code for obtaining an instance of a URLFactory depends on whether you are writing a portlet, theme code, or if you are already running a code within the resolver framework.

If you are writing a portlet:

- In the Portal javadoc, locate the `com.ibm.portal.resolver.servicer.service` package.
- Obtain a `PortletPocServiceHome` instance in 2 steps:
  1. Do a JNDI lookup for the `PorletServiceHome` instance by using the name constant `PortletPocServiceHome.JNDI_NAME`. See the javadoc for `PortletPocServiceHome` for sample code.
  2. On the returned `PortletServiceHome` object, call `getPortletService(PortletPocServiceHome.class)`.
- Do the JNDI lookup and call `getPortletService` in the `init(PortletContig)` method of your portlet, and save the resulting `PortletPocServiceHome` instance in a class instance variable. It is not necessary to redo these calls on every request that is serviced by the portlet.
- In a per-request method, such as `render(RenderRequest,RenderResponse)`, call the appropriate `PortletPocServiceHome.getPortletPocService()` method based on the specific method that is implemented and the type of request and response that are passed. There are versions of this method for Action, Event, Render, Resource and plain PortletRequest and Response.

  Note: It is necessary to call `dispose()` on the returned service instance before it goes out of scope.
- On the `PortletPocService` or appropriate subclass, call `getURLFactory()`.

If you are writing theme code:

- In the Portal javadoc, locate the `com.ibm.portal.resolver.service` package.
- Do a JNDI lookup for the `PortalPocServiceHome`, by using the name constant `PortalPocServiceHome.JNDI_NAME`.
- Save the resulting `PortalPocServiceHome` instance in a class instance field. It is not necessary to do the JNDI lookup for every request.
- In per-request method of your class, on the saved `PortalPocServiceHome` instance call `getPortalPocService` passing the `HttpServletRequest` and `HttpServletResponse`.

Note: It is necessary to call `dispose()` on the returned service instance before it goes out of scope.

- On the resulting `PortalPocService` instance, call `getURLFactory()`.

If you are writing code that runs as a DataSource or ResolutionService:

- In the Portal javadoc, locate the `com.ibm.portal.resolver.service` package.
- Do a JNDI lookup for the `CorPocServiceHome`, by using the name constant `CorPocServiceHome.JNDI_NAME`.
- Save the resulting `CorPocServiceHome` instance in a class instance field. It is not necessary to do the JNDI lookup for every request.
- In per-request method of your class, on the saved `CorPocServiceHome` instance call `getCorPocService(com.ibm.content.operations.registry.api.Context)`.

  Note: It is necessary to call `dispose()` on the returned service instance before it goes out of scope.

  Programming detail for a DataSource: In most cases, a DataSource is allocated by a dedicated factory. The factory is registered with the resolver framework to handle requests for a specific URI. When a request for that URI is received, the factory is started by the resolver framework and is passed a `com.ibm.content.operations.registry.api.Context` object. Make sure to understand the difference between this `COR Context` object and the `javax.naming.Context` class that is used by JNDI lookup. The factory allocates a new DataSource instance (or might retrieve an existing one from a pool) and calls `reset()` on the DataSource, passing the `COR Context` object. This object and the other parameters to `reset()`, must be saved in instance fields within the DataSource. They can be used on subsequent method calls until either `dispose()` is called or `reset()` is called again. Use the `COR Context` object on the call to `getCorPocService (com.ibm.content.operations.registry.api.Context)`.
- On the resulting `CorPocService` instance, call `getURLFactory()`.

After you obtain a `PocURLFactory`:

- Call one of the `newURL` methods, to instantiate the desired type of `PocURL`.
- On the resulting `PocURL` instance, call the `setXXX` methods to define your URL attributes.
- Stream the URL to the response and dispose of the `PocURL` instance by calling `writeDispose()`. The Writer instance for use by `writeDispose` must be obtained from the response argument to the method.

"URL generation by using the Navigational State SPI" on page 2846
The Navigational State API is used to read and modify the Navigational State document within a Portal URL. The Navigational State document is the seemingly random string of characters that appears in WebSphere Portal URLs. This string is a compressed, encoded XML document that contains a large amount of information that supports various Portal functions, including bookmark-ability of Portal pages and back button support.

Related concepts:

"Portlet communication" on page 3069
IBM WebSphere Portal Express supports multiple ways for portlets to exchange or

share information.

"Developing themes and skins" on page 2520
You can create themes using modules to contribute to separate areas of pages to provide flexibility, enhance the user experience, and maximize performance. To optimize themes on your website, use the theme optimization module framework. The framework separates feature-specific logic and capabilities from the theme code.

"Interoperability between JSR 286 portlet events and JSR 168 cooperative portlets" on page 3087
By concept, cooperative portlets are similar to JSR 286 portlet events. Both concepts describe publish/subscribe communication patterns that are based on typed information that is published and received by portlets and propagated via communication links.

"JSP tags for standard portlets" on page 3133
The standard portlet API defines several tags that can be used in portlet JSPs to access the portlet request and response and to generate URLs.

Related tasks:

Package javax.portlet

IBM WebSphere Portal, Version 8.5.0.0 SPI Specification

URL generation by using the Navigational State SPI

The Navigational State API is used to read and modify the Navigational State document within a Portal URL. The Navigational State document is the seemingly random string of characters that appears in WebSphere Portal URLs. This string is a compressed, encoded XML document that contains a large amount of information that supports various Portal functions, including bookmark-ability of Portal pages and back button support.

For details about the back button behavior and the support of bookmarks, see to the topic *Back button behavior*.

You can use the Navigational State SPI to read, create, and modify URLs that carry navigational state. The `com.ibm.portal.state` package is the main package of the Navigational State SPI. It holds the service interfaces and the interfaces that make up the navigational state object model.

The SPI offers two services (obtainable through JNDI) to create such URLs. Which service you choose depends on whether you are writing Portal-level code or portlet code, for example within a theme.

Portal-level code: Portal State Manager Service
A portal service that is used to realize use cases that go beyond what the portal tags provide. Use it to create URLs within portal artifacts such as themes, skins, and custom JSP tags. You can also use it in artifacts that are removed from the request processing, such as Enterprise JavaBeans. However, the Portal State Manager Service is not meant to be used for portlets. The Portal State Manager service classes are in package `com.ibm.portal.state.service`. A `PortalStateManagerService` instance is obtained with a JNDI look-up that uses a name constant from the `PortalStateManagerServiceHome` interface.

Portlet code: Portlet State Manager Service
The counterpart service that supports JSR168 and JSR286 compliant portlets. Use the Portlet State Manager Service to create URLs within

portlets that cannot be created by using the Standard Portlet APIs. A `PortalStateManagerService` instance is obtained with a `PortalStateManagerServiceHome` instance.

Tip: A `PortletStateManagerService` instance is obtained with a `com.ibm.portal.portlet.service.PortletServiceHome` instance.

Note: CF05 Consider the new Portal-defined public render parameters support for these use cases, instead of the Navigational State SPI. The render parameters provide most of the same functions in a much simpler way.

For more information about all SPI interfaces, see the Javadoc.

"Object Model"
Learn about the main object models used in the navigational state SPI.

"Accessor SPI" on page 2849
The Accessor SPI provides typed access to the state document model. It allows the programmer to query and modify navigational state information. The Accessor SPI is part of the package `com.ibm.portal.state.accessors.*`.

"URL generation services" on page 2851
Learn about the services that are used to create URLs in the navigational state SPI.

Related concepts:

"Back button behavior" on page 1506
Learn how the behavior of the web browser Back button can affect portal navigation.

Object Model

Learn about the main object models used in the navigational state SPI.

Navigational state

The Navigational state object model is a hierarchy of state information, which is modeled as a DOM-like document containing untyped state information represented as strings because a typical page contains many URLs that require the navigational state to be serialized multiple times per request. The string-based memory allows for efficiently serializing navigational states into URLs, because it avoids processing time and CPU consuming object to string conversions during the serialization process.

The Navigational state is modeled via the following interfaces:

- `com.ibm.portal.state.StateHolder`
- `com.ibm.portal.state.StateHolderController`
- `com.ibm.portal.state.dom.DocumentModel`
- `com.ibm.portal.state.dom.DocumentController`

The `StateHolder` interface provides read access to the navigational state information by exposing a `getModel()` method that returns an interface to the untyped document model, namely the `DocumentModel` interface. The `DocumentModel` interface provides a set of methods to inspect both the hierarchical structure of the state document as well as single nodes being part of that document.

In addition, the `StateHolder` interface offers a `newState()` method which can be used to create a clone of that particular StateHolder instance. This is an important method because, typically, the current navigational state has to be cloned before

modifying it according to the specific semantics of the generated URL. For example if the programmer wants to create a URL that points to a particular portal page, the page selection information in the state document model has to be changed; however, this modification must not take effect on all URLs of a page.

Beyond that, the `StateHolderController` interface also allows you to modify state information by offering a `getController()` method that delivers a controller interface to the untyped state document called `DocumentController`. The `DocumentController` interface provides a means to modify the hierarchical structure of the state document, which includes creating document nodes, inserting them into the node hierarchy, and removing nodes.

Engine URLs

URLs are modeled via the `com.ibm.portal.state.EngineURL` interface. An `EngineURL` represents a URL that contains navigational state. The initial `StateHolder` an `EngineURL` refers to is specified when requesting a new `EngineURL` instance from the appropriate URL factory; see "URL generation services" on page 2851 for additional information. Typically it is a copy of the request-specific base state.

The `EngineURL` interface provides the following methods:

`StateHolderController getState()`
Returns a read-write interface to the navigational state carried by the URL.

`void setProtected(Boolean flag)`
Specifies whether the URL should point to the public or protected area.

`void setSecure(Boolean flag)`
Specifies whether a secure https connection is required.

`Writer writeCopy(Writer out)`
Streams the URL to the given writer. Maintains the state of the URL. For example this method can be used to write the URL multiple times.

`Writer writeDispose(Writer out)`
Streams the URL to the given writer and finally releases the state of the URL. The `EngineURL` object must not be accessed again after invoking this method.

`void dispose()`
The `dispose()` method is inherited from the `Disposable` interface. It must be invoked to indicate that the `EngineURL` object is no longer needed. The `EngineURL` object must not be accessed again after invoking this method. Alternatively, you can invoke the `writeDispose()` method, which calls `dispose()`.

The crucial method is the `getState()` method, which returns the state holder object this particular `EngineURL` instance refers to.

Note: The method returns a controller interface (`StateHolderController`) that allows the programmer to modify the state of this `EngineURL`. See "Accessor SPI" on page 2849 for additional information about modifying the state.

Resource URLs

URLs that address generic resources such as, but not limited to, files, icons, and voice grammars cannot contain navigational states and are therefore modeled

using a separate interface called com.ibm.portal.state.DisposableURL. The DisposableURL interface almost offers the same methods as EngineURL but does not provide a getState() method.

Resource URLs can also be created using the URL generation Services that are offered along with the Navigational State SPI. See "URL generation services" on page 2851 for additional information.

Accessor SPI

The Accessor SPI provides typed access to the state document model. It allows the programmer to query and modify navigational state information. The Accessor SPI is part of the package com.ibm.portal.state.accessors.*.

The Accessor SPI is an abstract layer that surrounds the access to particular nodes in the hierarchical document model. For more information about the hierarchical document model, see "Object Model" on page 2847. For each state aspect such as, but not limited to, page selection, expansion states, and portlet states, the SPI offers an accessor factory. The accessor factory provides read-only and read-write accessor controls that are designed for the particular state aspect they refer to. The accessors read from or write to the respective positions in the state document model and does the required type conversions.

The navigational state information is in the state document model and is available when the accessor factory is used. After the node is located, the accessor factory passes a node reference to the accessor or accessor controller during. The accessor and accessor controller are independent from the state document model structure; you can reuse accessors even if the information is moved to another node in the state document.

The SelectionAccessorFactory offers the following interfaces, as shown in the example:

SelectionAccessor getSelectionAccessor(StateHolder)
: This method returns a SelectionAccessor interface that allows for reading page selection information from the StateHolder.

SelectionAccessorController getSelectionAccessorController(StateHolderController)
: This method returns a SelectionAccessorController interface that allows programmers to modify page selection information. The controller uses the StateHolderController interface to modify the navigational state accordingly.

The flyweight pattern, where the StateHolder or StateHolderController is used as an argument, is commonly used in the accessor factory interfaces. The navigational state the accessor operates on cannot be the base state that is retrieved from the request URL; typically, it is the state clone created for a particular EngineURL. Call getState() on the EngineURL object to obtain the URL-specific state holder. The following example shows how to make a created EngineURL point to a certain portal page (for example, the "Stock Market" page) by using the SelectionAccessorController:

```
final EngineURL url = ...;
final SelectionAccessorFactory selectionFct = ...;

final SelectionAccessorController selectionCtrl =
    selectionFct.getSelectionAccessorController(url.getState());

try {
```

```
    selectionCtrl.setSelection("wps.StockMarket");
} catch (StateException e) {
    // include error handling here
} finally {
    selectionCtrl.dispose();
}
```

Using the SelectionAccessorController to create a page link

The base Accessor interface is derived from the com.ibm.portal.Disposable interface. Start the dispose() method to indicate when the accessor is no longer required. Using the dispose() method allows the accessor factory to store the accessors and accessor controllers in object pools to achieve better performance (due to less initialization and garbage collection overhead).

The Navigational State SPI offers the following accessor factories, each covering a certain state aspect:

SelectionAccessorFactory
: The SelectionAccessorFactory provides accessors to read and write portal page selection information. To create a URL that points to another page, the SelectionAccessorController needs to be requested from the factory to include the new selection into the state, the created EngineURL is associated with.

PortletAccessorFactory
: The PortletAccessorFactory provides accessors to read and write portlet-related navigational state information, which includes portlet mode, window state, and render parameters. In particular the PortletAccessorController can be used to change the navigational state of a portlet (for example, the portlet mode).

PortletTargetAccessorFactory
: The PortletTargetAccessorFactory provides accessors to read and write portlet action-related information. In particular the PortletAccessorController can be used to declare a portlet as the target of an action. This action allows the programmer to create URLs that trigger portlet actions.

SoloAccessorFactory
: The SoloAccessorFactory provides accessors to read and write the so-called Solo state. If the portal is in Solo state, it renders only one particular portlet of the current portal page; all navigation controls and toolbars are hidden. The SoloAccessorController can be used to create URLs that activate/deactivate the Solo state for a particular portlet.

ThemeTemplateAccessorFactory
: The ThemeTemplateAccessorFactory supports reading and writing theme template information. In particular the ThemeTemplateAccessorController can be used to create URLs that switch to a certain theme template.

LocaleAccessorFactory
: The LocaleAccessorFactory provides accessors to read and write locale information. The LocaleAccessorController can be used to set a special locale into the navigational state and thus into a URL.

  Note: A locale that is retrieved from such a URL takes precedence over user preferred locales or locales that are defined on your browser.

ExpansionStatesAccessorFactory
The ExpansionStatesAccessorFactory provides accessors to read and write expansion states information; for example, to determine whether a navigation node in a navigation tree control is expanded or collapsed. The ExpansionStatesAccessorController is typically used to generate URLs that toggle the expansion state of a navigation node.

ShowToolsAccessorFactory
The ShowToolsAccessorFactory provides accessors to read and write tool-related information. The ShowToolsAccessorController is typically used to create a URL that blends in the tool icons for portlet windows that offer functions such as moving/deleting the respective portlet.

StatePartitionAccessorFactory
The StatePartitionAccessorFactory provides accessors to read and write state partition identifiers. The StatePartitionAccessorController can be used to include a state partition identifier into the navigational state. A new state partition identifier is included into URLs that open new browser windows or iFrames.

EngineActionAccessorFactory
The EngineActionAccessorFactory provides controllers that are used to create engine action URLs. The EngineActionAccessorController, in particular, allows you to set action parameters.

Note: The EngineActionAccessorFactory does not offer a read-only accessor because the portal manages the engine actions.

URL generation services

Learn about the services that are used to create URLs in the navigational state SPI.

The Navigational State SPI offers the following two services that create URLs that are accessible through the Java Naming and Directory Interface (JNDI):

- The com.ibm.portal.state.service.PortalStateManagerService is a portal service that provides services beyond what the portal tags provide. Use the PortalStateManagerService to create URLs within portal artifacts. You can create URLs within portal artifacts such as themes and skins (for example custom JSP tags). And within the server-side artifacts that are removed from the request processing (for example Enterprise JavaBeans). This service is outside the portlet context.
- The com.ibm.portal.portlet.service.PortletStateManagerService supports JSR 168 and JSR 286 compliant portlets. Use the PortletStateManagerService to create URLs within portlets that cannot be created with the Standard Portlet API. This service is for portlets only.

Both services use a common interface that is called com.ibm.portal.state.service.StateManagerService, which provides the functions that are common to both services. You can reuse URL generation code in themes, skins, and portlets if they use the common interface. For example, you can use a custom JSP tag, which was originally designed for use in themes, in a portlet-specific JSP because only the service lookup is different.

Getting access to the PortalStateManagerService

Access the PortalStateManagerService through a JNDI lookup with the lookup name "portal:service/state/PortalStateManager". The lookup returns a com.ibm.portal.state.service.PortalStateManagerServiceHome interface that

provides the following two get commands that retrieve com.ibm.portal.state.service.PortalStateManagerService:

PortalStateManagerService getPortalStateManagerService(HttpServletRequest request, HttpServletResponse response)
The returned service is suitable for creating URLs in themes, skins, and custom JSP tags. All server-related information that is needed to generate the URLs (such as host name, protocol, server port, context path) is retrieved from the servlet request.

PortalStateManagerService getPortalStateManagerService(ServerContext ctx, Locale locale, Profile profile, boolean isProtected, boolean isSecure)
This method returns a PortalStateManagerService used for "offline" use cases such as creating URLs in environments where the servlet request is not available (for example, in an Enterprise JavaBeans). Using this method requires that the server-related information (such as host name, protocol, server port, context path) is provided in the ServerContext argument. The com.ibm.portal.state.accessors.url.ServerContext interface is the required abstraction for this kind of information. If you want to create URLs that point to a certain virtual portal, the provided ServerContext object needs to address that virtual portal. The ServerContext object addresses that virtual portal either through the context path or by host name. The two Boolean arguments specify whether the URLs must point to the protected area by default (isProtected) and whether the created URLs must be served with a secure connection by default (isSecure). To support creating resource URLs, the arguments locale and profile must be provided. The locale argument is needed to create URLs that address locale-specific resources such as language-dependent icons. The profile argument represents the client profile and in particular allows for checking the capabilities of the client.

The lifetime of a PortalStateManagerService object depends on whether you requested a "request-specific" service or "offline" service. The request-specific service has request scope; it is only used for the duration of one servlet request. For all subsequent servlet requests, request a new service instance from the home interface. The lifetime of the offline PortalStateManagerService corresponds with the lifetime of the ServerContext object.

The obtained PortalStateManagerServiceHome object is valid for the lifetime of the portal. You must do the JNDI lookup only once and store the retrieved home object accordingly, for example in an instance or static variable.

The following example shows how to get access to the service:

```
/** the JNDI name to retrieve the PortalStateManagerServiceHome object */
private static final String JNDI_NAME =
  "portal:service/state/PortalStateManager";

/** PortalStateManagerServiceHome object to retrieve the service from */
private static PortalStateManagerServiceHome serviceHome;

public void useRequestService(final HttpServletRequest request,
                              final HttpServletResponse response) {
  try {
    // get the request-specific service from our home interface
    final StateManagerService service
      = getServiceHome().getPortalStateManagerService(request, response);
    // use the service
    // ...
    // indicate that we do not need it any longer
```

```
    service.dispose();
  } catch (Exception e) {
    // error handling
  }
}

public void useOfflineService(final ServerContext ctx,
                              final Locale locale,
                              final Profile profile,
                              final boolean prot,
              final boolean secure) {
  try {
    // get the offline service from our home interface
    final StateManagerService service
      = getServiceHome().getPortalStateManagerService(ctx, locale, profile, prot, secure);
    // use the service
    // ...
    // indicate that we do not need it any longer
    service.dispose();
  } catch (Exception e) {
    // error handling
  }
}

/**
 * Looks up the PortalStateManagerServiceHome being valid
 * for the lifetime of the portal.
 */
private static PortalStateManagerServiceHome getServiceHome() {
  if (serviceHome == null) {
    try {
      final Context ctx = new InitialContext();
      serviceHome =
        (PortalStateManagerServiceHome) ctx.lookup(JNDI_NAME);
    } catch (Exception e) {
      // error handling
    }
  }
  return serviceHome;
}
```

Note: The `PortalStateManagerService` interface is derived from the generic `StateManagerService` interface. As the `StateManagerService` extends the `com.ibm.portal.Disposable` interface, you must indicate when you do not need to access the service any longer by starting the offered `dispose()` method on it. Dispose the request-specific `PortalStateManagerService` by the end of the processed servlet request.

Getting access to the `PortletStateManagerService`

Access the `PortletStateManagerService` through JNDI with the lookup name "`portletservice/com.ibm.portal.state.service.PortletStateManagerService`". The lookup returns a generic `com.ibm.portal.portlet.service.PortletServiceHome` interface, which offers a `getPortletService(Class)` method to get the `PortletStateManagerService`. The retrieved `PortletStateManagerService` instance is valid for the lifetime of the portal. You must do the service retrieval in the method of the portlet and store it in a portlet instance variable. The following example shows the required lookup code:

```
public class MyPortlet extends GenericPortlet {

  /** The JNDI name which is needed to lookup the service */
  private static final String JNDI_NAME =
    "portletservice/com.ibm.portal.state.service.PortletStateManagerService";
```

```
/** portlet state manager service */
protected PortletStateManagerService service;

/**
 * @see javax.portlet.GenericPortlet#init()
 */
public void init() throws PortletException {
  super.init();
  try {
    // lookup the portlet state manager service
    final Context ctx = new InitialContext();
    final PortletServiceHome serviceHome = (PortletServiceHome)
      ctx.lookup(JNDI_NAME);
    service = (PortletStateManagerService)
      serviceHome.getPortletService(PortletStateManagerService.class);
  } catch (NameNotFoundException e) {
    throw new PortletException(e);
  } catch (NamingException e) {
    throw new PortletException(e);
  }
}
```

You can use the actual service within the render method of the portlet to include URLs into the markup. Or use the service in the helper methods that serve the mandatory portlet modes such as `doView`, `doEdit`, and `doHelp`) to include URLs into the markup. Or use the service in the `processAction` method to send a redirect to a certain URL. The `PortletStateManagerService` interface shows the following two methods:

PortletStateManager getPortletStateManager(PortletRequest request, PortletResponse response)

This method returns a `PortletStateManager` object, which you can use during action processing and rendering (for example in the `processAction` method, `doView` method, `doEdit` method, etc.). The `PortletStateManager` interface adds more methods to extend the generic `StateManagerService` interface. For example, you can directly read the current request-specific navigational state of the portlet (portlet mode, window state, and render parameters).

PortletStateManagerController getPortletStateManagerController(ActionRequest request, Action response)

This method returns a `PortletStateManagerController` that extends the `PortletStateManager` interface. The `PortletStateManagerController` provides more methods that allow for modifying the navigational state of the portlet for the current request. And is therefore only accessible during action processing (for example, in the `processAction` method of the portlet).

Both the `PortletStateManager` and the `PortletStateManagerController` have request scope, which means that you must not store references to them across requests. Instead, you must retrieve the service from the `PortletServiceHome` object. To indicate that the retrieved `PortletStateManager` or `PortletStateManagerController` instance is no longer accessed in the scope of a request, you must start the dispose method. The dispose method is inherited from the Disposable interface on it. The following example shows you how to retrieve the service from the `PortletServiceHome` object:

```
/**
 * @see javax.portlet.GenericPortlet#doView(RenderRequest, RenderResponse)
 */
protected void doView(
```

```
  final RenderRequest request, final RenderResponse response)
  throws PortletException, IOException {
  response.setContentType(request.getResponseContentType());
  final PrintWriter writer = response.getWriter();
  try {
    // get the request-specific portlet state manager
    final PortletStateManager mgr = service.
      getPortletStateManager(request, response);
    // do something (create URLs etc.)
    // ...
    // indicate that we do not need the portlet state manager any longer
    mgr.dispose();
  } catch (StateException e) {
    throw new PortletException(e);
  }
}
```

The base interface `StateManagerService`

The `PortletStateManager` and the `PortalStateManagerService` are derived from the `com.ibm.portal.state.service.StateManagerService` interface, which offers functionality that is common to both URL generation services. The use cases that are common to both services refer to the creation of EngineURLs that carry navigational state and resource URLs. Therefore, the `StateManagerService` interface must be sufficient to implement most of the use cases. The interface provides the following two methods:

URLFactory getURLFactory()
: This method returns a `com.ibm.portal.state.URLFactory` object that offers several methods to create various URLs. For further details on the `URLFactory`.

AccessorFactory getAccessorFactory(Class accessorFactoryClass)
: This method provides access to the various accessor factories. For more information, see "Accessor SPI" on page 2849. You must pass the respective accessor factory interface in as a method argument to retrieve a certain accessor factory the `Class` object.

  Note: When you use the `PortletStateManagerService`, the set of accessor factories is restricted to the `SelectionAccessorFactory`, `PortletAccessorFactory`, `PortletTargetAccessorFactory`, `SoloAccessorFactory`, `ThemeTemplateAccessorFactory`, `LocaleAccessorFactory`, `StatePartitionAccessorFactory`, and `ExpansionStatesAccessorFactory`.

Typically, you request an `EngineURL` object from the `URLFactory` and then modify the navigational state according to the required URL semantics. To do this step, call the `getState()` method of the `EngineURL` to get the `StateHolderController` object that is required to modify the navigational state through the Accessor SPI. The following code snippet exemplarily illustrates this typical usage pattern. The following example shows how to call the `getState()` method of the `EngineURL`:

```
protected EngineURL createPageLink(final ObjectID pageID)
  throws StateException {
  // get the URL factory from the state manager service
  final URLFactory urlFactory = service.getURLFactory();
  try {
    // get a EngineURL from the factory; maintain navigational state
    final EngineURL url = urlFactory.newURL(null);
    final SelectionAccessorFactory selectionFct = (SelectionAccessorFactory)
      service.getAccessorFactory(SelectionAccessorFactory.class);
    // get a selection controller that operates on the URL-specific state
```

```
    final SelectionAccessorController selectionCtrl =
      selectionFct.getSelectionAccessorController(url.getState());
    try {
      // modify page selection and return URL
      selectionCtrl.setSelection(pageID);
      return url;
    } finally {
      selectionCtrl.dispose();
    }
  } finally {
    urlFactory.dispose();
  }
}
```

In the previous code sample, the `newURL(Constants.Clone)` method of the `URLFactory` is used. However, the `URLFactory` offers several more convenience methods to create `EngineURLs` and resource URLs. The following list provides a complete description of the `URLFactory` interface:

`EngineURL newURL(Constants.Clone type)`

Choose this method to create a `EngineURL` because it is suitable for all prevalent use cases. The method provides an `EngineURL`, which refers to the `StateHolder` representing the navigational state of the current request. The type argument specifies how the request-specific `StateHolder` must be cloned for the URL to be created. There are the following four pre-defined clone constants:

- The `SMART_COPY` indicates that a shallow `StateHolder` copy must be created. The copy records the state modifications that are applied for this particular `EngineURL` only, instead of copying all nodes in the document model to construct the clone. The `SMART_COPY` represents the default; therefore passing in null is equivalent to `SMART_COPY`. This clone method also allows the generation of relative "delta" URLs.
- The `DEEP_COPY` results in a complete copy of the request-specific `StateHolder`, for example each node and the node hierarchy is cloned. The deep copy prevents the generation of delta URLs.
- The `EMPTY_COPY` indicates that the contents of the request-specific `StateHolder` must be cleared, for example the created `EngineURL` is based on an empty state. Any interaction with such a URL results in the lost the navigational state of previous interactions.

If the `URLFactory` does not have access to the current request, a new empty `StateHolder` is created internally. In that case, the type argument does not take any effect.

`EngineURL newURL(StateHolder state, Constants.Clone type)`

This second `getURL` method requires that the `StateHolder` the `EngineURL` is based on, is passed explicitly. Accordingly, the type argument refers to this particular `StateHolder`. Use this method when the URL to be created must encode a `StateHolder` that was defined programmatically. Clicking this URL results in the lost the navigational state of previous interactions with the portal.

`EngineURL newURL(URLContext ctx, Constants.Clone type)`

With this variant, you can specify whether the created URL must be absolute, server-relative, or relative. This action can be achieved by passing an `URLContext` interface. The interface must be implemented accordingly; for example if the URL must be absolute, the `isAbsolute()` method must return true whereas `isRelative()` and `isServerRelative()` must return false. Because this method does not require an explicit `StateHolder`

argument, the `EngineURL` to be created encodes the `StateHolder` retrieved from the request. To reduce markup size, it is strongly recommended to pass in an `URLContext` that allows for relative URLs. If the `URLFactory` does not have access to the current request, a new empty `StateHolder` is created internally. In that case, the type argument does not take any effect.

`EngineURL newURL(URLContext ctx, Constants.Clone type)`
This method is the counterpart of the previous method and takes a `StateHolder` defined as an explicit argument.

`DisposableURL newResourceURL( String name, PortalResources.Type type)`
This method creates a URL that points to a generic resource. The file name and resource type identify the resource. The resource lookup takes request-specific information such as the current locale, the client device, and the markup name into account (if the request is available). The `com.ibm.portal.state.accessors.url.PortalResources` interface provides several useful constants for the resource type that represents resource types such as files, sounds, icons, voice grammars.

`DisposableURL newResourceURL( String name, PortalResources.Type type, PortalResources.State state)`
This method creates a URL that points to a generic resource. The file name and resource type identify the resource. It can also contain a resource state that is used to further distinguish the lookup of the resource. The resource lookup takes additional information from the request into account (if available). This is the current locale, the client device, markup that is chosen for the client, and the theme name.

Note: An absolute URL is a complete URL containing protocol, host name, and port. If there is a server-relative URL, the browser implies the current protocol, host name, and port. If there is a relative URL, the browser appends the URL to either the current request URL or to the value of the HTML base tag (if any). Server-relative and relative URLs cannot be enforced. In a protocol switch from "http" to "https", for example, the generated URL is absolute in any case.

Changing the host name of absolute URLs

You can change the host name of absolute URLs for security reasons. For example, you can do this change if the DOM of an application, such as a portlet, runs within an iframe and you do not want the JavaScript code within that iframe to be able to access the DOM of the HTML document. If all URLs within that iframe are absolute URLs and the host name of these URLs is different from which the document originates, then the iframe has access to itself only. In other words, it cannot manipulate or access the rest of the document DOM. To ensure this action, create all URLs within the portlet by using the Navigational State SPI with absolute URLs. The portlet must then define a virtual host name, which must be rerouted by a proxy to the portal server again. In addition, the portlet must ensure that all requests in which the required absolute URLs are to be generated contain a special request header. The request header tells the portal, the name of the virtual host name. The headers are as follows:

com.ibm.lotus.openajax.virtualhost
Use this header for setting the host name of every generated absolute URL to the value of this request header.

com.ibm.lotus.openajax.virtualport
Use this header for setting the port of every generated absolute URL to the value of this request header.

Model SPI overview

Models provide information that is needed by WebSphere Portal Express to perform tasks such as content aggregation or building navigation to browse the aggregated content. The information that is aggregated is represented through models that can be accessed programmatically by using the Model SPI (read-only). The information of a model is usually persistent (stored in a database) but can also be transient (computed and stored only in memory). Models can be represented by using a tree structure (nodes have a parent-child relationship), a list structure, or a selection structure (a selected element in a tree structure).

The following models can be obtained by using the Model SPI:

Content Model
Describes the topology in which the content is structured. The content model is a tree structure that is composed of content nodes. Types of content nodes include pages, labels, internal URLs, and external URLs.

Navigation Model
Describes the topology of the navigation visible to a specific user, which is composed of navigation nodes. Navigation nodes are implied by the structure of the content model. A navigation node references content that is represented by content nodes.

Navigation Selection Model
Describes the selected node in the navigation.

Content Metadata Model
Provides access to metadata of nodes of the Content Model. The metadata are aggregated by using the hierarchy that the content model exposes.

Language List
A list of supported languages.

Layout Model
Describes the layout of a page, which is composed of layout nodes. Layout nodes can be containers, which affect the layout of the page (rows and columns), or controls, which affect the content (portlets) of a page.

Markup List
A list of supported markup languages.

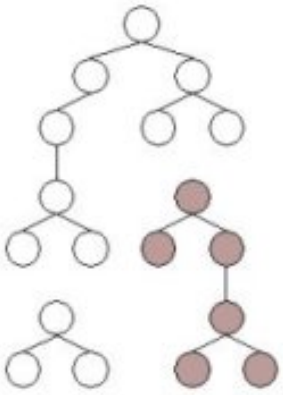
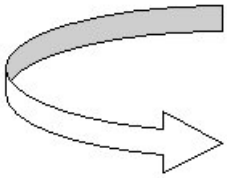
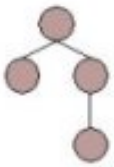
Skin List
A list of skins.

Theme List
A list of themes.

PortletModel, AdminPortletModel
Describes portlets and their associated configuration data.

The following table shows how a model is requested for specific use from the model source, and the target model that is returned.

Table 452. Representations of a model source, request and return from a model, and the model target, with the resulting filtered model

Model source	The specific model is requested from the source, and the source returns the specific model	Model target that is returned and filtered for a specific use
		

Create a specific model from source information

The interfaces of the public object model provide read only access to resources. Manipulation is possible only through the Controller SPI .

The models and their elements are all described through interfaces that are contained in the package `com.ibm.portal` and its sub packages. The more common interfaces are located higher in the package hierarchy. For example, `Identifiable`, an interface present on almost all resources, is located directly under `com.ibm.portal`.

Model scope

The models present information based on access control. Therefore, models are scoped to a specific user, and requests to retrieve model information return only the resources to which the user has access.

The concept of virtual portals scopes some models to the virtual portal in which a user operates. At the moment, this scoping concept applies to the content model, the navigation model, and the navigation selection model. These models scope their resources to the virtual portal in which a user operates.

The main package, com.ibm.portal

The `com.ibm.portal` package holds interfaces that are commonly used throughout the object model. This document describes important interfaces and classes that are found in the object model. It does not describe every class. For the complete information about all interfaces, read the Javadoc documentation.

Most resources carry an identifier. You can use it to address or locate them. This identifier is defined with the interface Identifiable, by which you can retrieve the ID of an element. An object ID uniquely identifies an element in an installation - and beyond (the ID is also called GUID - globally unique ID). An object ID can

optionally have a unique name assigned (a name that can exist only once per installation) to make it easier to address specific elements.

Do not use `toString` on object IDs - a human readable representation is returned that cannot be parsed back into an ObjectID object. For conversion between an ObjectID object and its string representation, the Identification interface is used (see package `com.ibm.portal.identification`).

A common operation is to search for resources that have a specific object ID. To perform this search in a general way, the concept of a Locator is introduced: A locator is provided by a model and allows searching for elements of the model in specific ways. To obtain such a locator object, you must use the getLocator method of an object that implements the LocatorProvider interface.

You can use the generic locator interface to locate a resource by its object ID (findByObjectID) or by its unique name (findByUniqueName). Some models provide specialized locators that extend the generic locator to provide more search functions such as search by title or some other criterion.

On a generic level, models are either lists or trees. Therefore, a TreeModel and a ListModel interface exist in the main package. In a tree model, you can perform the following tasks.

- Obtain the root node of a tree model (getRoot)
- Query the children of a node of the model (hasChildren)
- Obtain the children of a node of the model (getChildren)
- Obtain the parent of a node of the model (getParent)

With these methods, it is possible to explore a tree model. You must obtain the input arguments that represent nodes of the model from the model itself.

Combined with the locator concept discussed previously, a tree model becomes a searchable tree model: SearchableTreeModel is a tree model that also extends from LocatorProvider.

Some models also take on the form of a list, for example the list of markups or the list of languages that WebSphere Portal Express supports. The generic way to directly access the elements of the list is through the iterator provided by it (iterator method). When a method returns a list model, it uses the interface ListModel. However, in some situations it might also return a PagedListModel, which extends from ListModel and additionally provides a paged iterator. It makes it possible to obtain the elements of the list model in chunks, which can give better performance than obtaining each element individually.

List models become searchable the same way by which tree models do: SearchableListModel is a list model that also extends from LocatorProvider.

A widespread interface on elements of models is Localized. This interface provides a title and description of an element. Title and description are properties that are locale-dependant. To obtain a title or description call getTitle(Locale) or getDescription(Locale), depending on the case.

Note: An element returns a title or description only if such information exists in the exact locale that is passed in to the methods mentioned. No fallback mechanism is implemented inside of these methods. A suitable fallback mechanism

must be employed by invokers of the methods of Localized, or the default that is provided by LocalizedStringResolver (see package `com.ibm.portal.model`) can be used.

"Sub packages of the Model SPI"
Sub packages provide information on installed resources, hold Identification interface, and define the navigational model and the content that is represented in the Portal. The sub packages also represent portlets and their configuration data in the portal and the interconnections between the portlets.

"Obtain a model from the portal" on page 2867
Portal models can be obtained by using three different ways, depending on where the code that uses them is located.

"Obtaining the object ID for a page or portlet" on page 2868
There are several use cases when a portlet needs to obtain the object ID used to uniquely identify a portlet or a page. For example, the object ID of a page definition is required for a portlet to start a dynamic instance of that page.

"Filtering the content model" on page 2869
By applying filters to the content model, you can exclude parts of the page hierarchy from the content model. Filtering is performed based on request data and metadata assigned to the pages.

"Model SPI samples" on page 2869
The Model SPI can be used in portlets, themes, and skins. The models can be used with authenticated users and also with the anonymous user.

"Remote Model SPI REST service" on page 2871
The Remote Model SPI gives you access to portal models through REST services. It allows you to obtain and modify portal resources that are exposed by some of the models of the model SPI remotely, that is from clients that are outside the JVM of the server. This is achieved by means of REST services.

Related concepts:

"Controller SPI " on page 2883
You can use the Controller SPI for portal administration. It allows you to modify portal resources. It enhances the read-only portal Model SPI by adding writable aspects.

Sub packages of the Model SPI

Sub packages provide information on installed resources, hold Identification interface, and define the navigational model and the content that is represented in the Portal. The sub packages also represent portlets and their configuration data in the portal and the interconnections between the portlets.

The Model SPI includes the following sub packages.

- "The sub package com.ibm.portal.admin" on page 2862
- "The sub package com.ibm.portal.content" on page 2862
- "The sub package com.ibm.portal.identification" on page 2864
- "The sub package com.ibm.portal.navigation" on page 2864
- "The sub package com.ibm.portal.portletmodel" on page 2865
- "The sub package com.ibm.portal.wire" on page 2866

See "Model SPI overview" on page 2858 for a description of the main package in the Model SPI: `com.ibm.portal`.

The sub package com.ibm.portal.admin

WebSphere Portal provides several list models that contain information on installed resources, such as supported languages, supported markups, skins, and themes. For each of these resources, a specific ListModel exists that provides information on these resources. This package defines several list models:

- Language list model
- Markup list model
- Skin list model
- Theme list model
- Device class model

Each list model contains elements that implement interfaces that coincide with their names: The language list model has Language elements. The markup list model has Markup elements. The skin list model has Skin elements. The theme list model has Theme elements, and the device class model has device class elements.

The LanguageList contains all languages that are supported by the portal. The MarkupList contains all markups that the portal generally supports. From these list models, a resource might support only a selection. The SkinList and the ThemeList hold the themes and skins that are installed for the portal. The DeviceClassModel contains the device classes that the portal generally supports.

The sub package com.ibm.portal.content

Elements of this package define how content is represented in the portal. The two main models of this package are the ContentModel and the LayoutModel. The content model defines a tree structure for content elements (such as pages and labels); it is used to group these content elements logically.

Note:
The topology of the content model currently dictates the topology of the navigation model. For more information, see section *The sub package com.ibm.portal.navigation*. Theoretically, the topologies of content model and navigation model are allowed to diverge.

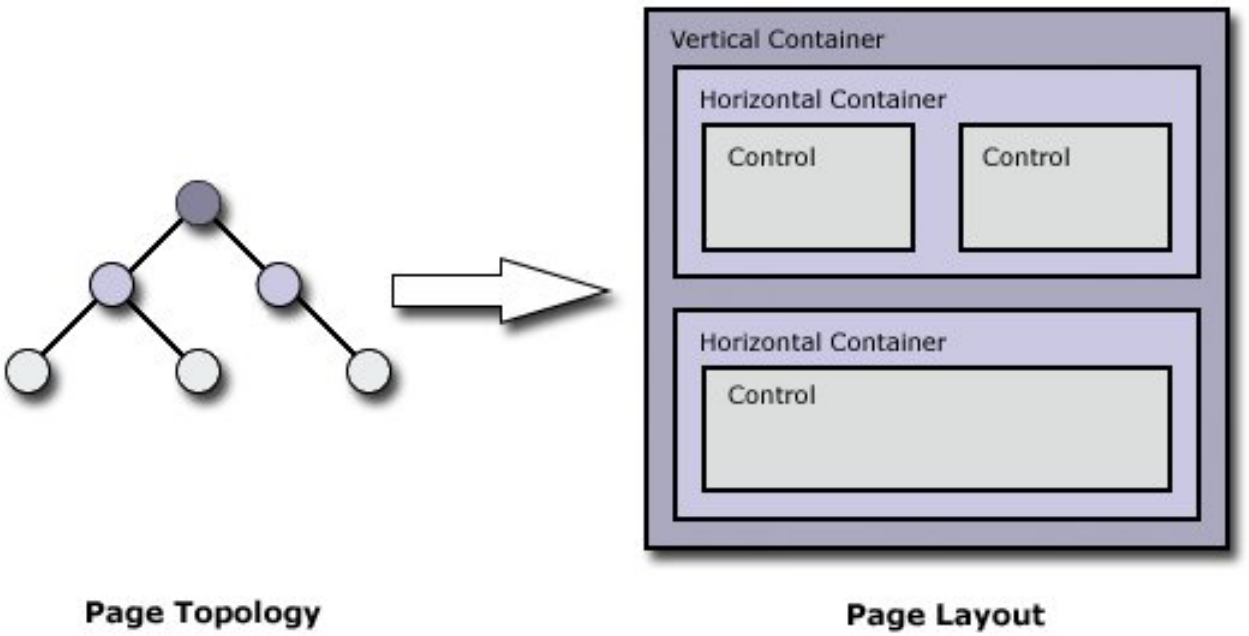
Currently, the elements of the content model can be pages, labels, or URLs. For each type, a specific ContentNode interface exists: ContentPage, ContentLabel, and ContentURL. Each content node has different information that is associated with it.

Each content node can provide the type of content node it represents; if this type is ContentNodeType.PAGE, a model exists that represents the layout of that page. It can be obtained with the getLayoutModel method of the content model. The layout model holds LayoutNode elements that describe the layout and contents of the page. Each layout node can return metrics in form of the LayoutMetrics interface. It can describe attributes of the node such as width or orientation (horizontal/vertical).

The elements of the layout model are either LayoutContainers that define rows and columns or LayoutControls, which represent portlets. This information is used when a page is rendered.

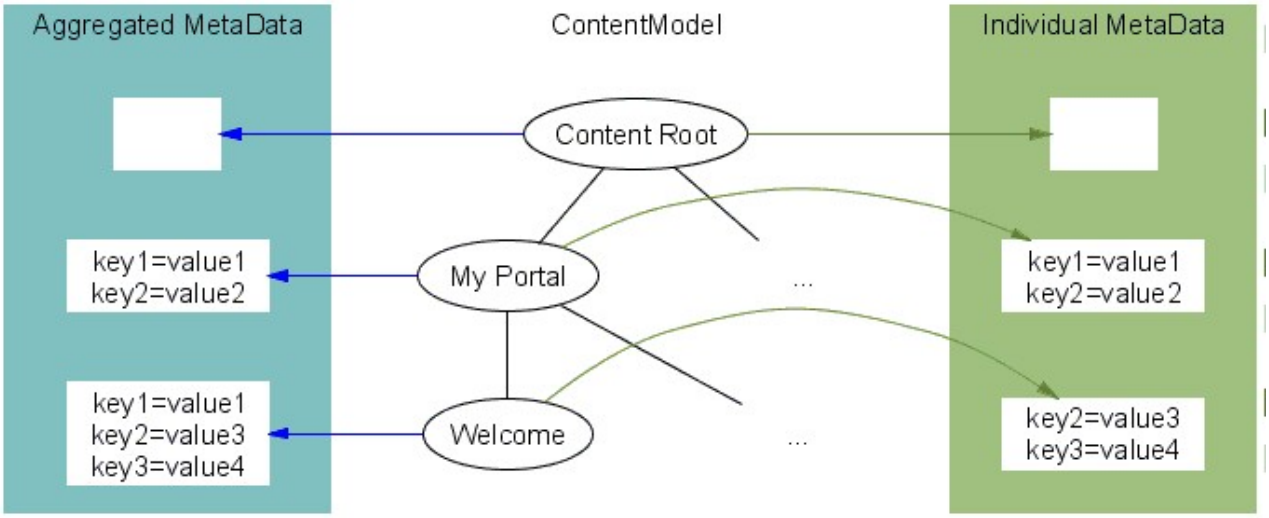
The following figure shows how page layout information and page content is represented. The figure depicts a page with three portlets. The surrounding vertical and horizontal containers define the layout while the controls hold the portlets that

provide the actual content presented to the user.



Content nodes can provide metadata through the com.ibm.portal.MetaDataProvider interface. Metadata can be used to associate arbitrary information with a content node. The content metadata model provides a view on metadata of the nodes in the content model. It aggregates the metadata of the individual nodes into one metadata object by using the hierarchy of the nodes as described in the content model.

The following figure shows the relationship between the individual metadata that are shown by content nodes and the aggregated view that the content metadata model provides:



The metadata of individual content nodes are shown in XML Access scripts by using the <parameter> tag.

The sub package com.ibm.portal.identification

This package holds the Identification interface, which allows the conversion between ObjectID objects and their string representation. This Identification interface is required where an object ID is to be passed as a parameter that can be only a String (for inclusion into URLs, for example). An implementation of this interface can be retrieved by using a JNDI lookup, as in the following example.

```
try
{
   Context ctx = new InitialContext();
   Name ctxName = new CompositeName("portal:services/Identification");
   Identification identification = (Identification) ctx.lookup(ctxName);
   String serializedOID = identification.serialize(aOID);

      ...

   ObjectID anotherOID = identification.deserialize(serializedOID);
}
catch (SerializationException sx)
{
   // some error handling code here
}
catch (NamingException nx)
{
   // some error handling code here
}
```

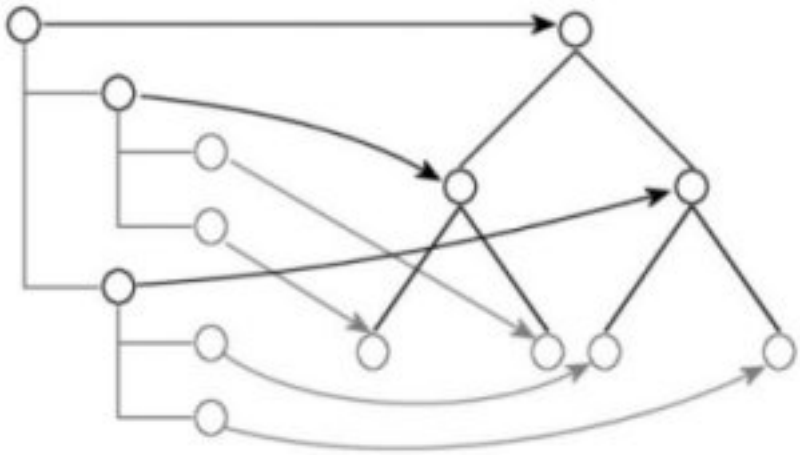
The sub package com.ibm.portal.navigation

This package defines the navigation model (NavigationModel) which represents a view on content model that is used for navigation. As described in section *The sub*

package com.ibm.portal.content, the topology of the navigation model currently corresponds with the content model. Each node in the content model has an equivalent navigation node at the same hierarchical level.

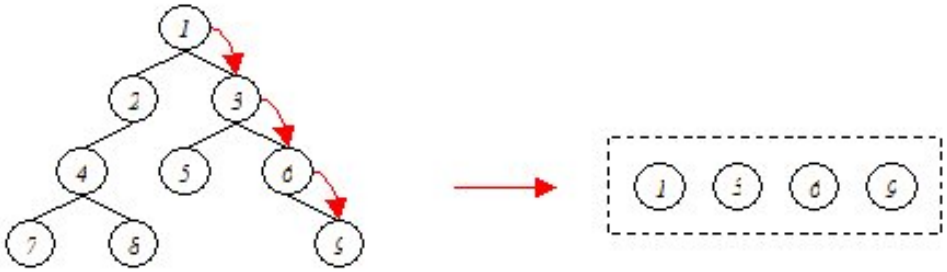
The elements of the navigation model are NavigationNodes. Each such node can reference a ContentNode. The navigation nodes have a title that is provided through the Localized interface. The following figure shows the connection between the navigation model and the content model.

The figure shows how nodes in the navigation model can reference a node in the content model.



When a user is going through the portal, the currently selected navigation node is important to render the current page. The NavigationSelectionModel reflects the current selection and represents a list that defines a path through the navigation model.

The figure shows how the navigation selection model defines a path through the navigation model.



The last node of this path is always the currently selected navigation node. Its referenced content node represents what is aggregated for the user to see (normally a page that is represented through a ContentPage object).

The sub package com.ibm.portal.portletmodel

The PortletModel represents portlets and their configuration data in the portal. It contains the following elements:

WebApplication
Represents a deployed WAR file in the portal.

Portlet
The programmer in the deployment descriptor of a portlet web application. Each portlet web application contains one or more portlets.

PortletDefinition
Represents administrator configuration for a portlet. Multiple portlet definitions can be associated with the same portlet so that the same portlet code can be started with different administrative settings. With the administrative UI of the portal, you can create new portlet definitions as copies of existing ones.

PortletEntity
Represents user configuration for a portlet. It is normally created by placing a portlet definition on a page; the same portlet definition can be added to multiple pages, resulting in multiple portlet entities. There can be two levels of user configuration (shared and personalized configuration) that are stored in the form of separate portlet entities.

PortletWindow
Represents a particular view on a portlet. It normally corresponds to a LayoutControl in the layout model. The same PortletWindow can be associated with different PorltetEntities for different users, if they personalize the portlet independently.

CommunicationEndpoint
Represents an endpoint that is available for communication that is defined for the portlet. Such endpoints can be of type CommunicationSource or CommunicationTarget and are available through the CommunicationEndpointProvider. This provider can be retrieved by the getEndpointProvider method on the PortletDefinition. For JSR 286 portlets that were developed for eventing, PublishingEventDefinition[s] and ProcessingEventDefinition[s] represent the events that a single portlet can process or publish as defined in the `portlet.xml`. The PublishingEventDefinition is a CommunicationSource and the ProcessingEventDefinition is a CommunicationTarget for JSR286 portlets.

The sub package com.ibm.portal.wire

The WireModel represents the interconnections between JSR 168 cooperative portlets or JSR 286 eventing portlets. A WireModel can be obtained from a LayoutModel by using the getWireModel method of the layout model. It contains only interconnections that originate from the page that the parent layout model represents. It contains the following element:

Wire Represents the interconnection between a CommunicationSource of the PortletDefinition of a PortletWindow on a source page and the CommunicationTarget of the PortletDefinition of a PortletWindow on a target page. Examples of interconnections are:

- Between a JSR 168 cooperative portlet with an output property and a JSR 168 cooperative portlet with a target action defined.
- Between a JSR 286 portlet that defines a publishing event and a JSR 286 portlet that defines a processing event.

Obtain a model from the portal

Portal models can be obtained by using three different ways, depending on where the code that uses them is located.

The different ways to obtain Portal models are as follows:

JNDI lookup

A JNDI lookup can be done for code that is in a request/response cycle of the portal, for example in a JSP. The lookup results in a home interface from which a model provider can be obtained. Interfaces for the models that are mentioned previously can be found in `com.ibm.portal.model`. To avoid calling the JNDI lookup too often, store the provider object. The following example, which is only applicable to non-portlet code that runs inside of a portal, shows how the content model is obtained by using a JNDI lookup.

```
try
{
   Context ctx = new InitialContext();
b   ContentModelHome home = (ContentModelHome)
                  ctx.lookup("portal:service/model/ContentModel");
   if (home != null) {
      ContentModelProvider provider = home.getContentModelProvider();
      ContentModel model =
                    provider.getContentModel(aRequest, aResponse);
    ...
   }
}
catch (NamingException nx)
{
    // some error handling code here
}
```

Figure 36. Obtaining a content model by using JNDI lookup.

Portlet service (standard portlet API)

Standard portlets can access models through specific portlet services. These portlet services are in `com.ibm.portal.portlet.service.model`. The following example shows how the navigation model is obtained by a standard portlet.

```
PortletServiceHome psh;
javax.naming.Context ctx = new javax.naming.InitialContext();
boolean serviceAvailable = false;
try {
   psh = (PortletServiceHome)
      ctx.lookup("portletservice/com.ibm.portal.portlet.service.model.NavigationModelProvider");
      serviceAvailable = true;
}
catch(javax.naming.NameNotFoundException ex)
{
    ... error handling...
}
...
if (serviceAvailable) {
   NavigationModelProvider provider = (NavigationModelProvider)
                           psh.getPortletService(NavigationModelProvider.class);
   NavigationModel model = provider.getNavigationModel(aRequest, aResponse);
   ...
}
```

Figure 37. Obtaining the navigation model.

Note the following limitations for obtaining a portal model:

1. WSRP portlets must not use the Model SPI.
2. Model access is only possible after the portal has initialized the request appropriately. Access is possible inside of code that is started through the portal servlet. Models cannot be accessed in servlet filters.

Obtaining the object ID for a page or portlet

There are several use cases when a portlet needs to obtain the object ID used to uniquely identify a portlet or a page. For example, the object ID of a page definition is required for a portlet to start a dynamic instance of that page.

You can use the lookup() method of the JNDI Context class to obtain the object ID for a portlet or a page, passing the unique name of the page or portlet. As an alternative for portlets, you can obtain the object ID by using JNDI lookup() and passing a combination of the portlet application ID and the portlet name.

- For standard portlets, the portlet application ID is the value of the ID attribute of the <portlet-app/> element in the `portlet.xml`. However, this attribute is not required by the Java Portlet Specification. If the ID attribute is not set, the portlet WAR file name is used as the portlet application ID.
- For IBM portlets, the portlet application ID is the UID attribute of the <portlet-app/> element. This attribute is required.

The following example shows both ways to find an object ID.

```
// initialization
Context ctx = new InitialContext();

// portal:uniquename prefix is required for unique name lookup
Name uniqueName = new CompositeName("portal:uniquename");

// portal:config/ prefix required for portlet definition lookup
Name portletName = new CompositeName("portal:config/portletdefinition");

// the unique name assigned to the page is example.org.page
uniqueName.add(org.example.page);

ObjectID oidForName = (ObjectID) ctx.lookup(uniqueName);

// appID and portletName have already been set programmatically
portletName.add(appID);
portletName.add(portletName);

ObjectID portletDefOID = (ObjectID) ctx.lookup(portletName);
```

The `Name` used for the lookup() method is created from a `CompositeName`, which is prepopulated with the required portal prefixes enclosed in quotation marks. This technique is used to avoid having to escape special characters in the prefix.

For this example, the following packages are required at a minimum for the import statements:

- `javax.naming.CompositeName`
- `javax.naming.Context`
- `javax.naming.Name`
- `com.ibm.portal.ObjectID`

Filtering the content model

By applying filters to the content model, you can exclude parts of the page hierarchy from the content model. Filtering is performed based on request data and metadata assigned to the pages.

About this task

There are two ways you can filter the content model:

- You can filter on the device class that is specified in the HTTP request. This filtering can be used to provide a unified content model for both desktop and mobile users. For pages that are specific to mobile users, you assign only the device class for the mobile client, such as a smartphone or tablet. These pages are then visible only to users who access the portal with one of the specified devices.
- You can filter on the markup language that pages support. This approach is useful when building mobile websites for older devices that use markup languages like WML or Compact HTML

Tip: When filtering for mobile devices that support HTML, filter only by device class.

Procedure

1. Enable content model filtering in the portal. By default, filtering is not enabled. This step must be done only once.
   a. Log in to the WebSphere Integrated Solutions Console.
   b. Click **Resources** > **Resource Environment** > **Resource Environment Providers**.
   c. Click **WP ConfigService**.
   d. Under **Additional Properties**, click **Custom Properties**.
   e. Click **New**, and enter the appropriate property name, depending on the type of filtering that you want to perform.

      Filter by device class
      Add **`content.topology.filter.deviceclass`**, and set the string value to the `true`.

      Filter by markup language
      Add **`content.topology.filter.markup`**, and set the string value to the `true`.
2. Save your changes, and restart the server.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Model SPI samples

The Model SPI can be used in portlets, themes, and skins. The models can be used with authenticated users and also with the anonymous user.

The following samples focus on JSPs, but the code can also be used in Java source files.

Displaying the portal selection path (breadcrumb trail)

The following example shows how to render a breadcrumb trail that shows the current selection path.

```
<%@ taglib uri="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0/portal-fmt"
    prefix="portal-fmt" %>
<%@ taglib uri="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0/portal-navigation"
    prefix="portal-navigation" %>
<%@ page import="com.ibm.portal.model.NavigationSelectionModelHome" %>
<%@ page import="com.ibm.portal.model.NavigationSelectionModelProvider" %>
<%@ page import="com.ibm.portal.navigation.NavigationSelectionModel" %>
<%@ page import="com.ibm.portal.navigation.NavigationNode" %>
<%@ page import="com.ibm.portal.ModelException" %>
<%@ page import="java.util.Iterator" %>
<%@ page import="javax.naming.InitialContext" %>
<%@ page import="javax.naming.Context" %>
<%@ page import="javax.naming.NamingException" %>

<%
try{
    Context ctx = new InitialContext();
    NavigationSelectionModelHome home = (NavigationSelectionModelHome)
        ctx.lookup("portal:service/model/NavigationSelectionModel");
    if (home != null) {
        NavigationSelectionModelProvider provider =
            home.getNavigationSelectionModelProvider();
        NavigationSelectionModel model =
            provider.getNavigationSelectionModel(request, response);
        for (java.util.Iterator i = model.iterator(); i.hasNext(); )
        {
            NavigationNode node = (NavigationNode) i.next();
            if (i.hasNext()) {
%>

        <a href="<portal-navigation:navigationUrl type='link' varname='<%=node%>'/>">
            <portal-fmt:title varname='<%=node%>'/>
        </a>
        &gt;
<%
            }
            else
            {
%>
        <portal-fmt:title varname='<%=node%>'/>
<%
            }
        }
    }
}
catch (ModelException mx) {
%>

<p><span style="color:#ff0000">A model exception occured</span></p>

<%
}
catch (NamingException nx) {
%>

<p><span style="color:#ff0000">A naming exception occured</span></p>

<%
}
%>
```

This example uses a JNDI lookup to obtain the navigation selection model. The model is then iterated and for each node a title is produced by using the <portal:title/> tag. Until the last node is reached, the <portal:navigationUrl/> tag is used to generate links to the referenced pages.

Getting page layout information

This example shows portlet code that displays layout information of a page as retrieved from the layout model of that page. The portlet outputs the layout structure of the page it is located. Two methods of the portlet are shown in the following example. The first, showTableModel, is the entry point for rendering a table that shows the layout of a page. The page is identified through the selected

navigation node of the navigation selection model. The other method, printLayoutElement, recursively traverses the layout model and outputs markup according to the element it meets.

```
/**
 * Displays a table model of the layout of a page.
 *
 * @param aWriter the writer where the output goes to
 * @param aLayoutModel the layout model to show
 * @param aPage the page for which to show the layout model\
 * @param aRequest the portlet request
 * @throws ModelException if an exception occurs
 */
private void showLayoutModel(final PrintWriter aWriter,
        final LayoutModel aLayoutModel, final ContentPage aPage,
        final RenderRequest aRequest)
      throws ModelException {
   aWriter.print("<h3>Layout for page ");
   aWriter.print(getTitle(aPage));
   aWriter.println("</h3>");
   // invoke the recursive traversal of the layout model;
   // start with the root node
   printLayoutNode(aWriter, aLayoutModel,
                 (LayoutNode) aLayoutModel.getRoot(), aRequest);
   }

/**
 * Outputs a single element of the layout model to the output writer.
 *
 * @param writer the writer where the output goes to
 * @param model the layout model to use
 * @param node the current node of the layout model
 * @param request the portlet request
 * @throws ModelException if an exception occurs
 */
private void printLayoutNode(final PrintWriter writer, final LayoutModel model,
        final LayoutNode node, final RenderRequest request) throws ModelException {
   if (node != null) {
      if (node instanceof LayoutControl) {
         // output control information
         writer.print("<b>Control</b> (");
         writer.print(getTitle((LayoutControl) node));
         writer.println(")");
      }
      else {
         writer.println("<table border=\"1\">");
         // get the layout metrics
         // (needed to find out the orientation of containers)
         final LayoutMetrics metrics = node.getLayoutMetrics();
         final Object info = metrics.getValue(LayoutMetrics.ORIENTATION);
         writer.print("<tr><td><b>Container</b> (");
         writer.print(info);
         writer.println(")");
         if (info == Orientation.HORIZONTAL) {
            writer.println("<table>");
            writer.println("<tr>");
            // recurse for the horizontal container
            if (model.hasChildren(node)) {
               for (Iterator i = model.getChildren(node); i.hasNext();) {
                  writer.println("<td valign=\"top\">");
                  printLayoutNode(writer, model, (LayoutNode) i.next(), request);
                  writer.println("</td>");
               }
            }
            writer.println("</tr>");
            writer.println("</table>");
         }
         else {
            // recurse for the vertical container
            if (model.hasChildren(node)) {
               writer.println("<table>");
               for (Iterator i = model.getChildren(node); i.hasNext();) {
                  writer.println("<tr><td>");
                  printLayoutNode(writer, model, (LayoutNode) i.next(), request);
                  writer.println("</td></tr>");
               }
               writer.println("</table>");
            }
         }
         writer.println("</td></tr></table>");
      }
   }
}
```

Remote Model SPI REST service

The Remote Model SPI gives you access to portal models through REST services. It allows you to obtain and modify portal resources that are exposed by some of the

models of the model SPI remotely, that is from clients that are outside the JVM of the server. This is achieved by means of REST services.

The Remote Model SPI supports the following models:

ContentModel
This allows you to obtain and modify the content topology and the properties of content nodes such as pages, labels, and content URLs.

NavigationModel
This allows you to obtain the navigation topology only, as the navigation model is implied by the structure of the content model.

LayoutModel
This allows you to obtain and modify the layout of a page, that is the topology of layout elements of a page, and the properties of layout elements, such as layout containers and layout controls.

PortletModel
This allows you to obtain, create, update, and delete portlets.

Note that the Remote Model SPI currently does not support the following models:

- LanguageList
- MarkupList
- SkinList
- ThemeList

The Remote Model SPI uses feeds in the Atom Syndication Format in conjunction with the Atom Threading Extension to expose model resources, and the HTTP-based Atom Publishing Protocol (APP) to modify portal resources. With the Remote Model SPI as a REST service, you must use the following HTTP verbs:

- To obtain model information: `HTTP GET`
- To modify existing model information: `HTTP PUT`
- To create model resources: `HTTP POST`
- To delete model resources: `HTTP DELETE`

"Feeds for REST services"
When you access a REST service to get information or to modify a portal resource, the response and in some cases also the request works by means of a feed. A feed contains information about one or more portal resources in a specific format as exposed by portal models. Learn how you obtain feeds for portal resources and what the format of such feeds is.

Feeds for REST services

When you access a REST service to get information or to modify a portal resource, the response and in some cases also the request works by means of a feed. A feed contains information about one or more portal resources in a specific format as exposed by portal models. Learn how you obtain feeds for portal resources and what the format of such feeds is.

You can request a feed by sending an HTTP request to a specific URL. A requested feed can contain links to dependent resources that can be used to request those resources iteratively. In order to modify resources, you can modify such a feed or create a new feed and send it to a specific URL by using an HTTP request.

Example: A feed that exposes full information on a layout container of a content page:

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:feed xmlns:atom="http://www.w3.org/2005/Atom"
           xml:base="/wps/mycontenthandler/"
           xmlns:thr="http://purl.org/syndication/thread/1.0"
           xmlns:xhtml="http://www.w3.org/1999/xhtml"
           xmlns:base="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0/ibm-portal-composite-base"
           xmlns:model="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model-elements"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xmlns:portal="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model">
  <atom:author>
    <atom:name>IBM WebSphere Portal/6.0.1</atom:name>
  </atom:author>
  <atom:title>IBM WebSphere Portal Model Feed</atom:title>
  <atom:id>lm:oid:6_E0HNTD421GG2502HO17LUG1G35</atom:id>
  <atom:link href="." rel="self" type="application/atom+xml"/>
  <atom:updated>2007-03-04T01:26:47.937-08:00</atom:updated>
  <atom:entry>
    <atom:title>7_E0</atom:title>
    <atom:id>lm:oid:7_E0@oid:6_E0</atom:id>
    <atom:updated>2007-03-04T01:26:46.655-08:00</atom:updated>
    <atom:content type="application/xml">
      <model:layout-container>
        <model:metadata name="ORIENTATION">
          <base:value xsi:type="base:String" value="Horizontal"/>
        </model:metadata>
        <model:templatename>UnlayeredContainer-H</model:templatename>
        <model:deletable>true</model:deletable>
        <model:deletableflag>true</model:deletableflag>
        <model:modifiable>true</model:modifiable>
        <model:modifiableflag>true</model:modifiableflag>
        <model:containerchild>true</model:containerchild>
        <model:controlchild>false</model:controlchild>
      </model:layout-container>
    </atom:content>
    <atom:link portal:uri="lm:oid:7_E0@oid:6_E0" href="/wps/myportal/c0/04_SB8K8...AXCmmeA!/"
               type="text/html"/>
    <atom:link href="?uri=lm:oid:7_E0@oid:6_E0&verb=download&levels=2&rep=compact"
               rel="replies" type="application/atom+xml"/>
  </atom:entry>
  <model:allportletsallowed>true</model:allportletsallowed>
</atom:feed>
```

"Syntax for addressing portal resources"
To obtain information about a portal resource or to modify a portal resource, you need to obtain a feed for this resource, or you send a request to a certain URL that can contain a feed. Find out how you can construct URLs to which you can send HTTP requests related to the REST service.

"Elements of a model feed" on page 2877
These sections show sample feeds for the different models.

"Use cases for modifying resources" on page 2880
Portal Model REST services allow you to create new resources, modify, move and delete existing resources. View some common use cases of resource modifications.

Syntax for addressing portal resources:

To obtain information about a portal resource or to modify a portal resource, you need to obtain a feed for this resource, or you send a request to a certain URL that can contain a feed. Find out how you can construct URLs to which you can send HTTP requests related to the REST service.

For example, a URL can address a specific resource or can contain certain query parameters that allow you to control the extent and the way by which information on resources is exposed, or they allow you to control the way by which resources are modified.

In order to obtain a feed that contains certain portal resources, you send an HTTP GET request to the following URL:

`/wps/[my]contenthandler[/vpmapping]?(uri=model-uri)+(&mode=verb)?(&name=value)*`

The meanings of the syntax elements are as follows:

contenthandler
The name of the servlet for unauthenticated access. No J2EE security context is available.

mycontenthandler
The name of the servlet for authenticated access. There is a J2EE security context. Refer to the security topics for more details.

vpmapping
The optional name of the URL mapping to a virtual portal. If no mapping is given, the default virtual portal is assumed, otherwise the ID of the addressed virtual portal is associated with the current thread.

uri=*model-uri*
The identification of the addressed resource, as described in Model schemas for addressing resources. The `uri` parameter can appear only once.

mode=*verb*
An optional specification for the mode of access. By default `mode=download` is assumed.

name=value
An arbitrary set of parameters. These include additional query parameters. For details refer to the following topics.

"Model schemas for addressing resources"
Addressing a resource includes specifying the model to which the resource is associated. To do this, you specify a schema.

"Additional query parameters" on page 2875
In order to specify the extent and the contents of the requested feed, you may use additional query parameters. All additional query parameters are optional.

Model schemas for addressing resources:

Addressing a resource includes specifying the model to which the resource is associated. To do this, you specify a schema.

There is a schema defined for each model that the remote model SPI supports:

- For addressing a resource in the content model: `cm`
- For addressing a resource in the navigation model: `nm`
- For addressing a resource in the layout model: `lm`
- For addressing a resource in the portlet model: `pm`

Furthermore, you need to specify the resource of the model itself, as described in the following resource addressing specification:

```
model-uri = "cm:" page-oid |
            "nm:" navigationnode-oid |
            "lm:" [layoutnode-oid "@"] page-oid |
            "pm:" portlet-oid ["@" page-oid
```

The meanings of the syntax elements are as follows:

page-oid
This is the portal object ID of a content page. When you address portlet model resources, this is required for portlet windows, but it must not occur for any other portlet model resource identification.

navigationnode-oid
This is the object ID of a navigation model node.

layoutnode-oid
This is the object ID of a layout model node.

portlet-oid
This is the object ID of a layout control, portlet window, portlet entity, or portlet definition.

oid This is the serialized string that represents a portal object ID. This is URI-escaped with UTF-8 encoding. The character `@` is also escaped by using `%40`. You can also use unique names instead of object IDs. Unique names also have to be URI-escaped, and the `@` character must be escaped. Note that every object ID has the defined scheme `'oid:'`.

Note that the model schemes are mandatory for ambiguity reasons. For example content and navigation nodes currently have got the same object IDs.

Example 1: To obtain a feed of the root node of content model, that is the content node with the unique name `wps.content.root`, send an `HTTP GET` request to the following URL:

```
/wps/mycontenthandler?uri=cm:oid:wps.content.root
```

Example 2: To obtain a feed of the layout node with the object id `'_7_0830M4HTFF0SHFCQ_2BV' on the content page with the object id '_6_0830M4HTFF0SHFCQ_4D'`, send an `HTTP GET` request to the following URL:

```
/wps/mycontenthandler?uri=lm:oid:_7_0830M4HTFF0SHFCQ_2BV@oid:_6_0830M4HTFF0SHFCQ_4D
```

Additional query parameters:

In order to specify the extent and the contents of the requested feed, you may use additional query parameters. All additional query parameters are optional.

Levels

You can request a feed of a resource that is maintained in a tree model. If you do this, you can parameterize the URL,by the number of levels that the response should contain as follows:

```
[ &levels = levelcount ]
```

levelcount is an integer value that is greater than zero. Values have the following meaning:

0 This is not a valid value and therefore returns `400: Bad Request`.

1 This represents the model node itself. As a client, you can obtain a subtree of a model by requesting the root of the required subtree and any levels beneath it.

2 This represents the requested node including its direct children.

>2 These are entries for the requested node and its children up to the specified depth. The maximum path length is limited by the maximum depth of the requested (sub-) model and level count.

all If you specify this value, the obtained feed includes all descendants of the requested root node. As this can be an expensive call, a client should use this only for the layout model or for testing purposes.

(absent)
This is equivalent to an `all` levels parameter.

The `level` parameter for the portlet model:
The PortletModel is not a tree model, therefore the `level` parameter has a different meaning in the context of the PortletModel. You can set it to the following values:

0 This is not a valid value and therefore returns `400: Bad Request`.

<0 This returns all parents on the hierarchy higher than the given object ID with the depths of the value specified.

(absent)
This returns an ATOM feed that contains the addressed node of the PortletModel. This is equivalent for a levels parameter with the value `-1`.

1 This is equivalent to a value of `-1`.

-all This returns all levels up to the root web application.

The parameters `level` and `mode` are mutually exclusive if the value for `mode` is `view`.

Example: A URL that explicitly specifies that the node itself and all its direct children should be contained in the returned response:

```
/wps/mycontenthandler?uri=nm:oid:wps.content.root&levels=2
```

Representation mode

If the feed always transports all available information, it will be large. Therefore there is a way to limit the size of the feed. You can use the query parameter `rep` to specify the volume of information that is transported. You use the parameter as follows:

```
[ &rep = compact | full | empty ]
```

You can set the parameter to the following values:

Compact
This is the default value for the generated links. For performance reasons, this reduces the volume of returned information to a subset of the most important items. The meaning of the `compact` representation mode is defined separately and different for each model.

full The full representation mode exposes all available information.

empty
The empty representation mode returns no response body. However, the response headers, especially the HTTP status code, are the same as if you

use the compact or full representation modes. For example, you can use this mode if a client wants to modify the resource, but does not evaluate the response for performance reasons.

(absent)
If you omit the parameter, the full representation mode is used.

Extension parameter

For special use cases, for example in the context of federation, there is a parameter that allows you to manipulate the ATOM alternate link, also known as the view link. The value specified for this parameter must be a URI. If you specify this parameter, the Remote Model SPI performs a Piece Of Content (POC) lookup through the interface `com.ibm.portal.resolver.LookupService` with the view mode and the given URI. At this time the Remote Model SPI uses the default lookup service. Use the parameter as follows:

`[ &extension=`*`uri`* `]`

Explicit MetaData

Some metadata names are hidden, as they are not exposed in the MetaData iterator; for example, this is the case for all names that start with `com.ibm.portal`. These names also do not show up in a feed.

In the `compact` representation mode no metadata may be exposed, although some certain MetaData are required. Therefore, in order to expose them in a feed, you need to explicitly request them. To do this, use the `&mdname` parameter as follows:

`[ &mdname=`*`string`* `]`

You can use the parameter `mdname` multiple times in the same URL.

Example: This URL specifies explicitly that the otherwise hidden metadata `com.ibm.portal.Hidden` is exposed in the returned response:

`/wps/mycontenthandler?uri=nm:oid:wps.content.root&mdname=com.ibm.portal.Hidden`

Elements of a model feed:

These sections show sample feeds for the different models.

For more information on what information are exposed by the feeds refer to the section about mapping of feed elements to elements of the Model SPI.

Content Model feeds

In order to obtain information on the content model, you need to send an HTTP GET request to the content handler with the model schema `cm`.

Example: Content Model feed for the Content Model root node:

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:feed xmlns:atom="http://www.w3.org/2005/Atom"
           xml:base="http://wps128.boeblingen.de.ibm.com:10040/wps/mycontenthandler"
           xmlns:thr="http://purl.org/syndication/thread/1.0"
           xmlns:xhtml="http://www.w3.org/1999/xhtml"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xmlns:portal="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model"
           xmlns:model="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model-elements"
           xmlns:base="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0/ibm-portal-composite-base">
```

```
    <atom:author>
           <atom:name>IBM WebSphere Portal/6.1</atom:name>
    </atom:author>
    <atom:title>IBM WebSphere Portal Model Feed</atom:title>
    <atom:id>cm:oid:6_00000000000000000000000000A0</atom:id>
    <atom:link href="?uri=cm:oid:6_00000000000000000000000000A0&mode=download&levels=1&rep=compact"
               rel="self" type="application/atom+xml"/>
    <atom:updated>2008-02-26T06:36:02.239Z</atom:updated>
    <atom:entry>
        <atom:title>Content Root</atom:title>
        <atom:id portal:uniquename="wps.content.root">cm:oid:6_00000000000000000000000000A0</atom:id>
        <atom:published>2002-05-31T22:00:00.000Z</atom:published>
        <atom:updated>2008-02-26T06:36:02.239Z</atom:updated>
        <atom:content type="application/xml">
            <model:content-label>
                <model:title>
                    <base:nls-string xml:lang="de">[G'Content Root13:48, 3 Mar 2008 (
                       W. Europe Standard Time)ﾂ¦]</base:nls-string>
                    <base:nls-string xml:lang="en">Content Root</base:nls-string>
                </model:title>
                <model:active>true</model:active>
                <model:supportedMarkup>wml</model:supportedMarkup>
                <model:supportedMarkup>html</model:supportedMarkup>
            </model:content-label>
        </atom:content>
        <atom:link href="?uri=cm:oid:6_00000000000000000000000000A0&mode=download&levels=2&rep=compact"
                   rel="replies" type="application/atom+xml"/>
        <atom:link portal:rel="contextMenu"
                   portal:uri="wp.operations:page:oid:6_00000000000000000000000000A0"
                   portal:uniquename="wps.content.root"
                   href="?uri=wp.operations:page:oid:6_00000000000000000000000000A0&mode=download"
                   rel="related" type="application/vnd.mozilla.xul+xml"/>
    </atom:entry>
</atom:feed>
```

Layout Model feeds

In order to obtain information on the layout model, send an HTTP GET request to the content handler with the model schema `lm`.

Example: Layout Model feed for a Layout Model root node of a specific content node:

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:feed xmlns:atom="http://www.w3.org/2005/Atom"
           xml:base="http://wps128.boeblingen.de.ibm.com:10040/wps/mycontenthandler"
           xmlns:thr="http://purl.org/syndication/thread/1.0"
           xmlns:xhtml="http://www.w3.org/1999/xhtml"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xmlns:portal="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model"
           xmlns:model="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model-elements"
           xmlns:base="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0/ibm-portal-composite-base">
   <atom:author>
           <atom:name>IBM WebSphere Portal/6.1</atom:name>
   </atom:author>
   <atom:title>IBM WebSphere Portal Model Feed</atom:title>
   <atom:id>lm:oid:6_CGAH47L00GD87023IKNDD41G65</atom:id>
   <atom:link href="?uri=lm:oid:6_CGAH47L00GD87023IKNDD41G65&mode=download&levels=1&rep=compact"
              rel="self" type="application/atom+xml"/>
   <atom:updated>2008-02-24T03:30:17.059Z</atom:updated>

   <atom:entry>
       <atom:title>7_CGAH47L00GD87023IKNDD41G67</atom:title>
       <atom:id>lm:oid:7_CGAH47L00GD87023IKNDD41G670oid:6_CGAH47L00GD87023IKNDD41G65</atom:id>
       <atom:published>2008-02-24T03:30:16.817Z</atom:published>
       <atom:updated>2008-02-24T03:30:16.862Z</atom:updated>
       <atom:content type="application/xml">
           <model:layout-container>
               <model:metadata name="ORIENTATION">
                   <base:value xsi:type="base:String" value="Horizontal"/>
               </model:metadata>
               <model:templateName>UnlayeredContainer-H</model:templateName>
               <model:deletable>true</model:deletable>
               <model:deletableFlag>true</model:deletableFlag>
```

```
                <model:modifiable>true</model:modifiable>
                <model:modifiableFlag>true</model:modifiableFlag>
                <model:containerChild>true</model:containerChild>
                <model:controlChild>false</model:controlChild>
            </model:layout-container>
        </atom:content>
        <atom:link portal:uri="lm:oid:7_CGAH47L00GD87023IKNDD41G67@oid:6_CGAH47L00GD87023IKNDD41G65"
                   href="/wps/mypoc/!ut/p/lm/oid:7_CGAH47L00GD87023IKNDD41G67@oid:6_CGAH47L00GD87023IKNDD41G65?uri
                   =lm%3aoid%3a7_CGAH47L00GD87023IKNDD41G67%40oid%3a6_CGAH47L00GD87023IKNDD41G65&mode=view"
                   type="text/html"/>
        <atom:link href="?uri=lm:oid:7_CGAH47L00GD87023IKNDD41G67@oid:6_CGAH47L00GD87023IKNDD41G65&mode=download&levels=2&rep=compact"
                   rel="replies" type="application/atom+xml"/>
    </atom:entry>

    <atom:entry>
        ....
    </atom:entry>

    <model:allPortletsAllowed>true</model:allPortletsAllowed>
    <atom:link portal:uri="wm:oid:6_CGAH47L00GD87023IKNDD41G65"
               portal:uniquename="ibm.portal.Home.Web20Introduction"
               href="?uri=wm:oid:6_CGAH47L00GD87023IKNDD41G65&mode=download&rep=compact"
               rel="related" type="application/atom+xml"/>
</atom:feed>
```

Navigation Model feeds

In order to obtain information on the Navigation Model, send an HTTP GET request to the content handler with the model schema `nm`.

Example: Navigation Model feed for the Navigation Model root node;

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:feed xmlns:atom="http://www.w3.org/2005/Atom"
           xml:base="http://wps128.boeblingen.de.ibm.com:10040/wps/mycontenthandler"
           xmlns:thr="http://purl.org/syndication/thread/1.0"
           xmlns:xhtml="http://www.w3.org/1999/xhtml"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xmlns:portal="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model"
           xmlns:model="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model-elements"
           xmlns:base="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0/ibm-portal-composite-base">
    <atom:author>
           <atom:name>IBM WebSphere Portal/6.1</atom:name>
    </atom:author>
    <atom:title>IBM WebSphere Portal Model Feed</atom:title>
    <atom:id>nm:oid:6_000000000000000000000000A0</atom:id>
    <atom:link href="?uri=nm:oid:6_000000000000000000000000A0&mode=download&levels=1&rep=compact"
               rel="self" type="application/atom+xml"/>
    <atom:updated>2008-03-02T19:02:43.754Z</atom:updated>
    <atom:entry>
        <atom:title>Content Root</atom:title>
        <atom:id portal:uniquename="wps.content.root">nm:oid:6_000000000000000000000000A0</atom:id>
        <atom:published>2002-05-31T22:00:00.000Z</atom:published>
        <atom:updated>2008-03-02T19:02:43.754Z</atom:updated>
        <atom:content type="application/xml">
            <model:navigation-node>
                <model:title>
                    <base:nls-string xml:lang="de">[G'Content Root~~~~~⌂ı|]</base:nls-string>
                    <base:nls-string xml:lang="en">Content Root</base:nls-string>
                </model:title>
            </model:navigation-node>
        </atom:content>
        <atom:link portal:rel="label" portal:uri="cm:oid:6_000000000000000000000000A0"
                   portal:uniquename="wps.content.root"
                   href="?uri=cm:oid:6_000000000000000000000000A0&mode=download&levels=1&rep=compact"
                   rel="related" type="application/atom+xml"/>
        <atom:link href="?uri=nm:oid:6_000000000000000000000000A0&mode=download&levels=2&rep=compact"
                   rel="replies" type="application/atom+xml"/>
    <atom:link portal:rel="contextMenu" portal:uri="wp.operations:page:oid:6_000000000000000000000000A0"
               portal:uniquename="wps.content.root"
```

```
            href="?uri=wp.operations:page:oid:6_00000000000000000000000000A0&mode=download"
            rel="related" type="application/vnd.mozilla.xul+xml"/>
    </atom:entry>
</atom:feed>
```

Use cases for modifying resources:

Portal Model REST services allow you to create new resources, modify, move and delete existing resources. View some common use cases of resource modifications.

The information exposed in Portal Model feeds is equivalent to the information exposed by the respective portal Model SPI. For information about how the elements of the feeds are mapped to the elements of the Model SPI refer to the information that follows here about use cases for modifying resources. Before you work with feed elements, make sure to familiarize yourself with the Model SPI.

Depending on the type of the modification, use the following HTTP verbs:

HTTP PUT
: Use this to update existing resources. This includes moving resources within model topologies,

HTTP POST
: Use this to create new resources.

HTTP DELETE
: Use this to delete existing resources

"Setting titles and descriptions of resources"
Portal Model REST services allow you to set titles and descriptions of resources.

"Creating Resources" on page 2881
Portal Model REST services allow you to create new resources.

"Putting a portlet on a page" on page 2882
Portal Model REST services allow you to put portlets on pages.

"Deleting resources" on page 2882
Portal Model REST services allow you to delete resources.

Setting titles and descriptions of resources:

Portal Model REST services allow you to set titles and descriptions of resources.

To set titles and descriptions of a resource, take the following steps:

1. Obtain the URI of resource for which you want to modify titles and descriptions.
2. Perform an HTTP PUT request to the URI that you obtained by the previous step, with a feed in the message body that contains titles and descriptions that you want to set.
3. Check for response `200 OK` from the server.

You can use the `update` parameter with either of the `merge` or `replace` modes:

`update=merge`
: Use the `merge` mode if you want to modify only the titles or descriptions of the resource. In this case, you need to send only the titles or descriptions that you want to modify.

`update=replace`
: Use the `replace` mode if you want to replace the complete entry of the resource. In this case, you need to send the complete resource.

Example: Feed to modify titles and description for a content node:

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:feed xmlns:atom="http://www.w3.org/2005/Atom"
           xml:base="http://wps128.boeblingen.de.ibm.com:10040/wps/mycontenthandler"
           xmlns:thr="http://purl.org/syndication/thread/1.0"
           xmlns:xhtml="http://www.w3.org/1999/xhtml"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xmlns:portal="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model"
           xmlns:model="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model-elements"
           xmlns:base="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0/ibm-portal-composite-base">
    <atom:author>
        <atom:name>IBM WebSphere Portal/6.1</atom:name>
    </atom:author>
    <atom:title>IBM WebSphere Portal Model Feed</atom:title>
    <atom:id>cm:oid:6_000000000000000000000000A0</atom:id>
    <atom:link href="?uri=cm:oid:6_000000000000000000000000A0&mode=download&levels=1&rep=compact"
               rel="self" type="application/atom+xml"/>
    <atom:updated>2008-02-26T06:36:02.239Z</atom:updated>
    <atom:entry>
        <atom:title>Content Root</atom:title>
        <atom:id portal:uniquename="wps.content.root">cm:oid:6_000000000000000000000000A0</atom:id>
        <atom:published>2002-05-31T22:00:00.000Z</atom:published>
        <atom:updated>2008-02-26T06:36:02.239Z</atom:updated>
        <atom:content type="application/xml">
            <model:content-label>
                <model:title>
                    <base:nls-string xml:lang="de">Mein neuer Titel</base:nls-string>
                    <base:nls-string xml:lang="en">My new title</base:nls-string>
                </model:title>
                <model:description>
                    <base:nls-string xml:lang="de">Meine neue Beschreibung</base:nls-string>
                    <base:nls-string xml:lang="en">My new description</base:nls-string>
                </model:description>
            </model:content-label>
        </atom:content>
    </atom:entry>
</atom:feed>
```

Creating Resources:

Portal Model REST services allow you to create new resources.

Example 1: Creating a page; note that this page has no layout or content defined:

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:feed xmlns:base="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0/ibm-portal-composite-base"
           xmlns:creation-context="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.2/portal-creation-context"
           xmlns:model="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model-elements"
           xmlns:portal="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model"
           xmlns:thr="http://purl.org/syndication/thread/1.0"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xmlns:atom="http://www.w3.org/2005/Atom">
    <atom:author>
        <atom:name>IBM WebSphere Portal/8.0</atom:name>
    </atom:author>
    <atom:title>IBM WebSphere Portal Model Feed</atom:title>
    <atom:id/>
    <atom:link href="." rel="self" type="application/atom+xml"/>
    <atom:updated>2007-07-04T13:28:37.721+0200</atom:updated>
    <atom:entry>
        <atom:title>test page - testpage_1183548505593</atom:title>
        <atom:id portal:uniquename="testpage_1183548505593">cid:testpage_1183548505593</atom:id>
        <atom:updated>2007-07-04T13:28:37.731+0200</atom:updated>
        <atom:content type="application/xml">
            <portal:content-page>
                <model:supported-markup>html</model:supported-markup>
                <model:supported-markup>wml</model:supported-markup>
                <model:title>
                    <base:nls-string xml:lang="en">test page - testpage_1183548505593</base:nls-string>
                </model:title>
```

```
            </portal:content-page>
        </atom:content>
        <thr:in-reply-to href="cm:oid:6_00000000000000000000000A0" ref="cm:oid:6_00000000000000000000000A0"
                         type="application/atom+xml"/>
    </atom:entry>
</atom:feed>
```

Example 2: Putting a layout container on a page:

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:feed xmlns:base="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0/ibm-portal-composite-base"
           xmlns:creation-context="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.2/portal-creation-context"
           xmlns:model="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model-elements"
           xmlns:portal="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model"
           xmlns:thr="http://purl.org/syndication/thread/1.0"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xmlns:atom="http://www.w3.org/2005/Atom">
  <atom:author>
    <atom:name>IBM WebSphere Portal/8.0</atom:name>
  </atom:author>
  <atom:title>IBM WebSphere Portal Model Feed</atom:title>
  <atom:id/>
  <atom:link href="." rel="self" type="application/atom+xml"/>
  <atom:updated>2007-07-04T13:28:56.358+0200</atom:updated>
  <atom:entry>
    <atom:title>lm:oid:newid_layout_0_1183548510931@oid:testpage_1183548505593</atom:title>
    <atom:id portal:uniquename="testpage_1183548505593_lmroot">lm:oid:testpage_1183548505593</atom:id>
    <atom:updated>2007-07-04T13:28:56.358+0200</atom:updated>
    <atom:content type="application/xml">
      <portal:layout-container/>
    </atom:content>
  </atom:entry>
</atom:feed>
```

Putting a portlet on a page:

Portal Model REST services allow you to put portlets on pages.

Example: Putting a portlet on a page:

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:feed xmlns:base="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0/ibm-portal-composite-base"
           xmlns:creation-context="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.2/portal-creation-context"
           xmlns:model="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model-elements"
           xmlns:portal="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model"
           xmlns:thr="http://purl.org/syndication/thread/1.0"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xmlns:atom="http://www.w3.org/2005/Atom">
    <atom:author>
        <atom:name>IBM WebSphere Portal/6.0.1</atom:name>
    </atom:author>
    <atom:title>IBM WebSphere Portal Model Feed</atom:title>
    <atom:id/>
    <atom:link href="." rel="self" type="application/atom+xml"/>
    <atom:updated>2007-07-04T13:29:14.444+0200</atom:updated>
    <atom:entry>
        <atom:title>lm:oid:newid_layout_5_1183548514616@oid:testpage_1183548505593</atom:title>
        <atom:id portal:uniquename="testpage_1183548505593_control">lm:oid:testpage_1183548505593</atom:id>
        <atom:updated>2007-07-04T13:29:14.444+0200</atom:updated>
        <atom:content type="application/xml">
            <portal:layout-control creation-context:portlet-definition="pm:oid:wps.p.Welcome"/>
        </atom:content>
        <thr:in-reply-to href="lm:oid:testpage_1183548505593_lmcontainer4@oid:testpage_1183548505593"
                         ref="lm:oid:testpage_1183548505593_lmcontainer4@oid:testpage_1183548505593"
                         type="application/atom+xml"/>
    </atom:entry>
</atom:feed>
```

Deleting resources:

Portal Model REST services allow you to delete resources.

To delete a resource, send an HTTP DELETE request to the resource that you want to delete.

Controller SPI

You can use the Controller SPI for portal administration. It allows you to modify portal resources. It enhances the read-only portal Model SPI by adding writable aspects.

The interfaces of the programming model for portal resources that are published under the topic *Model SPI overview* offered only read-only methods. The Controller SPI extends them by adding a set of new interfaces. These interfaces are derived from the read-only portal models and interfaces and map to them, but they also extend them with methods for modifying the resources that they represent. This way the Controller SPI allows you to modify portal resources to a certain extent.

Controller SPI overview

The Controller SPI provides controllers for portal resources. You can use these controllers to modify portal resources that are exposed by particular models of the Model SPI. Controllers offer methods to modify the topology and properties of the model and of its nodes. They expose the same interfaces as the corresponding read-only model, and they instantly reflect modifications that you apply to the controller.

Note: While the modifications come into effect immediately for the controller, they not reflected in the persistence layer until you commit the controller and the changes that you made by it.

The resources that are exposed by the controller can be modified through specific interfaces which match their read-only counterparts. For an example, refer to the following class list:

Classes from the read-only model . . .	**. . . are reflected in the Controller SPI by the following classes**
`ContentNode`	*`Modifiable`*`ContentNode`
• `MarkupCapable`	• *`Modifiable`*`MarkupCapable`
• `Localized`	• *`Modifiable`*`Localized`
• `Identifiable`	• *`Modifiable`*`Identifiable`
• `ActiveFlag`	• *`Modifiable`*`ActiveFlag`

Further benefits of the Controller SPI are as follows:

- Controller instances work as workspaces where you make your modifications. You can try your modifications and assess them in a preview mode. When your changes meet your requirements, you apply them to the portal by a commit step.
- You can make and apply your changes to a running portal environment. You do not need to restart the portal for the changes to take effect.

The Controller SPI provides the following controllers:

Content Model Controller

This allows you to modify the content topology and the properties of content nodes such as pages, labels, and content URLs.

Note: If you modify the content topology, this might also change the navigation of your portal for your users.

Layout Model Controller
This allows you to modify the layout of a page, such as the topology of layout elements of a page, and the properties of layout elements such as layout containers and layout controls.

Portlet Model Controller
This allows you to create, update and delete portlets.

Note: At this time there is no controller for the following models:

- `NavigationModel`. This is by implication of the structure of the content model.
- `NavigationSelectionModel`. This is computed from the navigational state per request.
- `LanguageList`, `MarkupList`, `SkinList`, and `ThemeList`.

A controller is based on the corresponding read-only model. This means when you first create the controller on the basis of a read-only model, both the controller and the model expose the same information. You can then use the controller to create, update, or delete information exposed through it. These changes will be reflected in the controller immediately. To persist changes that you made to the underlying read-only model, you need to commit the controller.

In particular, a controller offers methods to do all of the following:

- Provide modifiable instances of existing resources. These modifiable instances exist for each modifiable resource property, and they allow for these properties to be modified.
- Create and delete model resources.
- Obtain dependent controllers. This is optional. For example, a `ContentModelController` offers a method to obtain a `LayoutModelController`.
- Persist the modifications.

Note: Before you use the Controller SPI, be sure to familiarize yourself with the read-only models. Refer to *Model SPI overview*.

Scope of the Controller SPI

A controller instance is based on a read-only model instance. Therefore it has the same scope and lifetime as the corresponding read-only model. Consequently, the following equivalences apply:

- If the underlying model is scoped to a particular user, then so is the controller.
- If the underlying model is scoped to a request, then so is the controller.
- If the underlying model is scoped to a virtual portal, then so is the controller.

"Packages of the Controller SPI" on page 2885
The portal provides the SPI Controller in several separate packages.

"Working with controllers" on page 2886
When you modify a portal resource with the Controller SPI, you go through a set of steps.

"Making modifications by using the Controller SPI" on page 2888
The Controller SPI allows you to modify portal resources, the topology of your portal, and properties.

"Confirming modifications" on page 2901
Modifiable interfaces and Controller interfaces provide methods for confirming modifications. You can use these confirm methods to check whether modifications might be prevented by portal internal constraints.

"Hints and tips for using the Controller SPI" on page 2902
These are some hints and tips for using the Controller SPI.

Related concepts:

"Model SPI overview" on page 2858
Models provide information that is needed by WebSphere Portal Express to perform tasks such as content aggregation or building navigation to browse the aggregated content. The information that is aggregated is represented through models that can be accessed programmatically by using the Model SPI (read-only). The information of a model is usually persistent (stored in a database) but can also be transient (computed and stored only in memory). Models can be represented by using a tree structure (nodes have a parent-child relationship), a list structure, or a selection structure (a selected element in a tree structure).

Related information:

IBM WebSphere Portal 8 API and SPI Reference

Packages of the Controller SPI

The portal provides the SPI Controller in several separate packages.

These packages are as follows:

- `com.ibm.portal`. This package holds the following interfaces:
  - Base interfaces for the Controller SPI, for example `Modifiable` and `Controller`.
  - Modifiable interfaces, such as `ModifiableActiveFlag`, `ModifiableLocalized`, `ModifiableMetaData`.
- `com.ibm.portal.admin`. This package holds modifiable interfaces, for example the `ModifiableMarkupCapable` and the `LanguageListController`.
- `com.ibm.portal.content`. This package holds the following interfaces:
  - Interfaces for content and layout model controller modifiable interfaces for content and layout nodes. For example, these can be `ModifiableContentPage` or `ModifiableLayoutContainer`.
  - Modifiable interfaces for modifiable aspects of content and layout nodes, for example `ModifiableBookmarkableFlag`
  - Creation contexts for creating content pages and layout nodes, for example `LayoutContainerCreationContext`.
- `com.ibm.portal.model.controller`. This package holds home and provider interfaces to obtain controllers, including a builder factory for `CreationContext` instances.
- `com.ibm.portal.model.controller.exceptions`. This package holds controller specific exceptions.
- `com.ibm.portal.portlet`. This package holds the following interfaces:
  - Interfaces for the portlet model controller.
  - Modifiable interfaces for portlet definitions, entities and preferences, for example `ModifiablePortletPreferences` and `ModifiablePortletEntity`.
  - Portlet creation and cloning context interfaces.

Note: A controller interface usually resides in the same package as its corresponding read-only Model SPI interface.

Working with controllers

When you modify a portal resource with the Controller SPI, you go through a set of steps.

About this task

For example, these steps can be as follows:

Procedure

1. Obtain the appropriate controller.
2. Obtain a modifiable instance of the resource that you want to modify from the controller.
3. Apply your modifications as required to the modifiable instance Commit the controller to persist the modifications.
4. Optionally repeat steps 2 and 3 as required.
5. Commit the controller to persist the modifications. This step saves and applies the modifications to your portal.

What to do next

Note: You can modify multiple resources with one controller before you commit the modifications.

For more detailed information about each of these steps refer to the following topics.

"Obtaining a controller for working with resources"
To modify, create, or delete portal resources by using the Controller SPI, you first need to create a controller.

"Committing and persisting your modifications" on page 2887
To persist the modifications that you applied to the controller, you commit the controller.

Obtaining a controller for working with resources

To modify, create, or delete portal resources by using the Controller SPI, you first need to create a controller.

About this task

You do this by using a JNDI based lookup for the correct "home" interface, that is, the corresponding read-only interface.

Note: The provider lookup for a controller home is possible from servlet level code and portlets.

The following controllers are available via JNDI:

ContentModelController
To obtain a ContentModelController, perform a lookup for the string `ContentModelControllerHome.CONTENT_MODEL_CONTROLLER_JNDI_NAME`.

PortletModelController
To obtain a ContentModelController, perform a lookup for the string `PortletModelControllerHome.PORTLET_MODEL_CONTROLLER_JNDI_NAME`.

The **LayoutModelController** cannot be obtained via a JNDI lookup. You obtain it through its associated **ContentModelController**.

Example - Obtaining a content model controller:

```
ContentModelController result = null;
final Context ctx = new InitialContext();
final ContentModelControllerHome home = (ContentModelControllerHome)
     ctx.lookup(ContentModelControllerHome.CONTENT_MODEL_CONTROLLER_JNDI_NAME);
if (home != null) {
     result = home.getContentModelControllerProvider().createContentModelController(aContentModel)
}
```

Note: To obtain a `ContentModelController`, you must pass an existing content model to the `createContentModelController` method of the `ContentModelControllerProvider`.

Example 2 - Obtaining a layout model controller for a specific page:

```
// locate the page for which you want to create a LayoutModelController
final Locator locator = cmController.getLocator();
final ContentPage page = (ContentPage) locator.findByUniqueName("MyPage");

// create a LayoutModelController
final LayoutModelController lmController = cmController.getLayoutModelController(page);
```

Related concepts:

"Obtain a model from the portal" on page 2867
Portal models can be obtained by using three different ways, depending on where the code that uses them is located.

Committing and persisting your modifications

To persist the modifications that you applied to the controller, you commit the controller.

About this task

You can commit only the `ContentModelController` and the `PortletModelController` as only these two implement the `Committable` interface. You cannot commit a `LayoutModelController`. Committing the `LayoutModelController` is included when you commit the `ContentModelController` from which you obtained the `LayoutModelController`. Committing the `ContentModelController` can include committing more than one `LayoutModelController`.

Notes:

1. After you have successfully committed a controller, you must not use it any more. In particular, do not invoke the `commit()` method again at a later stage.
2. After you have used a controller, you always need to dispose it.
3. If you do not want to persist the changes made by using the controller, dispose it without invoking `commit()` on it.

The following example shows how you commit a controller and then dispose it:

```
// commit the controller
try {
    controller.commit();
} finally {
    controller.dispose();
}
```

Making modifications by using the Controller SPI

The Controller SPI allows you to modify portal resources, the topology of your portal, and properties.

About this task

The following topics describe how you use the Controller SPI for different types of modifications. Some topics provide example code snippets. For these snippets to work, they need to be completed with preceding and subsequent code. Examples are shown in the following.

Preceding code: Each snippet must be preceded at least by the following code:

```
ContentModelController cmController= null;
final Context ctx = new InitialContext();
final ContentModelControllerHome home = (ContentModelControllerHome)
      ctx.lookup(ContentModelControllerHome.CONTENT_MODEL_CONTROLLER_JNDI_NAME);
if (home != null) {
      cmController = home.getContentModelControllerProvider().
           createContentModelController(aContentModel);
}
```

Depending on the modification that you want to make, you might require additional preceding code statements. For more details refer to “Obtaining a controller for working with resources” on page 2886.

Subsequent code: After you have completed your modifications, you commit the controller so that the modification take effect. After that you cannot use or commit the controller any more, but you dispose it. To commit your modifications and dispose the controller, add the following statements to your code:

```
// commit the controller
try {
    controller.commit();
} finally {
    controller.dispose();
}
```

For more details about this refer to “Committing and persisting your modifications” on page 2887.

“Modifying portal resources and topologies”
The Controller SPI allows you to modify portal resources and the topology of your portal in different ways.

“Modifying properties” on page 2893
The Controller SPI enables the modification of properties resources.

Related tasks:

“Obtaining a controller for working with resources” on page 2886
To modify, create, or delete portal resources by using the Controller SPI, you first need to create a controller.

“Committing and persisting your modifications” on page 2887
To persist the modifications that you applied to the controller, you commit the controller.

Modifying portal resources and topologies

The Controller SPI allows you to modify portal resources and the topology of your portal in different ways.

About this task

You can make the following modifications:

- Create new resources
- Insert new resources into the portal topology
- Move existing resources
- Delete existing resources.

Depending on the resource that you create or modify, you can perform different types of modifications. For more detailed information about each of these refer to the following topics.

"Creating resources"
You create resources by using methods of controllers. Each controller type enables the specific resources for its type of model.

"Moving or Inserting Nodes" on page 2891
To move existing nodes or insert new nodes, use the `insert()` method of the controller.

"Deleting Nodes" on page 2893
To delete nodes, use the `delete()` method of the controller.

Creating resources:

You create resources by using methods of controllers. Each controller type enables the specific resources for its type of model.

About this task

The following list shows the existing controllers and the resources that you can create by using them:

- `ContentModelController`. This enables the creation of resources of the following types:
  - `ContentPage`
  - `ContentLabel`
  - The following two types of content URL:
    - `ExternalContentURL`
    - `InternalContentURL`
- `LayoutModelController`. This enables the creation of resources of the following types:
  - `LayoutContainer`
  - `LayoutControl`
- `PortletModelController`. This enables the creation of resources of the following types:
  - `PortletDefinition`
  - `PortletEntity`

To create a resource, proceed by the following steps:

Procedure

1. Optional: Obtain a creation context. Take the following two steps:
   a. Obtain the creation context builder factory. For details about how to do this step, refer to "Obtaining creation contexts" on page 2890.

b. Obtain a creation context from the creation context builder. For details about how to do this step, refer to "Obtaining creation contexts."

2. Obtain a controller.
3. Create a resource. To do this step, use the `create(Class, CreationContext)` method of the controller and provide both arguments.

Results

1. The created resource is not part of the topology, unless you insert it into the topology by using the `insert()` method of the controller.
2. For content nodes, that is content pages, content labels, and content URLs, you must set a supported markup. Otherwise, they do not show in the read-only model.

Example

Example 1 - Creating a resource by using the creation context builder: This example creates a `ContentPage` that you can later insert into the content model.

```
// obtain creation context builder
final CreationContextBuilderFactory builder = CreationContextBuilderFactory.getInstance();

// obtain creation context
final CreationContext creationContext = builder.newIdentifiableCreationContext(objectID);

// create resource (which is not yet part of the topology of the controller!)
final Modifiable modifiable = controller.create(ContentPage.class, creationContext);
```

Example 2 - Cloning an existing portlet definition: To clone an existing portlet definition, create a portlet definition by using a portlet model controller and specify a `PortletDefinitionCloningContext`.

```
// obtain portlet definition cloning context; includes obtaining a portlet definition
final PortletDefinitionCloningContext context = ... (portletDefinition)

// create portlet definition
pmController.create(PortletDefinition.class, context);
```

The result of the `create()` method is a `Modifiable` instance of the created resource that you can cast to the type of the created resource as shown in the following example.

Example 3 - Casting a created resource to its type:

```
final ModifiableContentPage modifiableContentPage = (ModifiableContentPage)
      controller.create(ContentPage.class, creationContext);
```

"Obtaining creation contexts"
You need a creation context to define immutable properties of a resource that you create. You can use the creation context builder factory to generate multiple such creation contexts without having to implement those interfaces directly.

Obtaining creation contexts:

You need a creation context to define immutable properties of a resource that you create. You can use the creation context builder factory to generate multiple such creation contexts without having to implement those interfaces directly.

About this task

The creation context builder factory can do both of the following:

- Generate single creation contexts, that is contexts that contain only one or more immutable properties. Examples:

  - A creation context for an object ID contains only the object ID property.
  - A layout control creation context contains two properties, portlet definition and portlet entity.
- Combine several creation contexts into one in order to define multiple immutable properties.

Note: You can only combine creation contexts that have not already been combined by using the creation context builder.

Example

Example 1 - Obtaining a simple creation context:

```
// obtain creation context builder
final CreationContextBuilderFactory builder = CreationContextBuilderFactory.getInstance();

// obtain creation context
final CreationContext creationContext = builder.newIdentifiableCreationContext(objectID);
```

Example 2 - Obtaining a combined creation context:

```
// obtain creation context builder
final CreationContextBuilderFactory builder = CreationContextBuilderFactory.getInstance();

// obtain combined creation context
final CreationContext creationContext = builder.combine(new CreationContext[]
     {builder.newContentPageCreationContext(true), builder.newIdentifiableCreationContext(objectID)});
```

What to do next

The following list describes creation contexts that you can create by using the creation context builder factory:

ContentPageCreationContext
Use this creation context to define whether a page that you create is private. This applies to the resource type `ContentPage` on the `ContentModelController`.

DerivedContentPageCreationContext
Use this creation context to define the derivation parent of a page that you want to create. This applies to the resource type `ContentPage` on the `ContentModelController`.

Note: This creation context derives from the `ContentPageCreationContext`.

IdentifiableCreationContext
Use this creation context to define an object ID for a resource that you want to create. This applies all resource types: `ContentModelController`, `LayoutModelController`, `PortletModelController`.

PortletDefinitionCloningContext
Use this creation context to define the portlet definition ID, and optionally, the domain for the portlet definition that you want to clone. This applies to the resource type `PortletDefinition` on the `PortletModelController`.

PortletEntityCreationContext
Use this creation context to define the parent ID, and optionally, the domain for the portlet entity that you want to create. This applies to the resource type `PortletEntity` on the `PortletModelController`.

Moving or Inserting Nodes:

To move existing nodes or insert new nodes, use the `insert()` method of the controller.

About this task

1. When you create a node, you must insert it into the topology of the model, for example, to make it visible and accessible for users.
2. Operations that you defined on the controller itself, such as `hasChildren()` or `getLayoutModelController()`, work only after you insert the node.

Procedure

1. Obtain an appropriate controller.
2. Identify the target location in the topology. You must have that information in a later step.
3. Identify the resource that you want to move or insert as follows:
   - For moving a resource, locate an existing node with the appropriate locators of the controller.
   - For inserting a resource, create a modifiable instance that represents a new resource.
4. Insert the resource into the topology of the controller by using its `insert()` method and by specifying its target location. This resource can be one of the following list:
   - A parent node. In this case, the resource is inserted as the last child of the parent node.
   - A parent node and a sibling node. In this case, the resource is inserted as the child of the parent node in a position immediately before the sibling node.
   - No location information at all. In this case, the resource is inserted as a new root.

     Note: This insertion is not possible for all controllers. For example, you cannot insert a new root node into a `ContentModelController` that has a root node already.
5. Persist your modifications by using the `commit()` method of the controller.

Example

Example 1 - Moving an existing portlet to a different container (error handling omitted):

```
// obtain locator of LayoutModelController
final Locator locator = lmController.getLocator();

// locate portlet and target containers
final LayoutControl control = (LayoutControl) locator.findByUniqueName("MyControl");
final LayoutContainer parentContainer = (LayoutContainer) locator.findByUniqueName("MyParentContainer");
final LayoutContainer nextContainer = (LayoutContainer) locator.findByUniqueName("MyNextContainer");

// move portlet under parentContainer just before nextContainer
lmController.insert(control, parentContainer, nextContainer);
```

"Placing a portlet on a page"
To put a portlet on a page, use the `insert()` method of the controller.

Placing a portlet on a page:

To put a portlet on a page, use the `insert()` method of the controller.

Procedure

1. Obtain a layout model controller for the page the portlet is to be put on.
2. Obtain a `LayoutControlCreationContext`; this includes obtaining the portlet definition ID, and optionally the portlet entity ID.

3. Create a layout control by using the context from the previous step and by using the `create()` method of the layout model controller.
4. Insert the created layout control into the layout model controller.
5. Persist your modifications by using the `commit()` method of the controller.

Example

Example 1 - Placing a portlet on a page:

```
// obtain layout model controller
final LayoutModelController lmController = cmController.getLayoutModelController(page);

// obtain layout control creation context
final LayoutControlCreationContext context   = ... (portletDefinition, null);
//final LayoutControlCreationContext context = ... (portletDefinition, portletEntity);

// create layout control
final Modifiable control = lmController.create(LayoutControl.class, context);

// insert control into the topology of the layout model controller (given a container and a sibling)
lmController.insert(control, container, sibling);

// commit the content model controller
cmController.commit();
```

Deleting Nodes:

To delete nodes, use the `delete()` method of the controller.

Procedure

1. Obtain an appropriate controller.
2. Locate the resource that you want to delete. Use an appropriate locator of the controller, or search the model by iterating until the required node is found.
3. Delete the resource by using the `delete()` method of the controller.
4. Persist your modifications by using the `commit()` method of the controller.

Example

Example 1 - Deleting a node (error handling omitted):

```
// obtain locator of ContentModelController
final Locator locator = cmController.getLocator();

// locate page to delete
final ContentPage page = (ContentPage) locator.findByUniqueName("MyPage");

// delete the page
cmController.delete(page);

// commit the controller
cmController.commit();
```

Modifying properties

The Controller SPI enables the modification of properties resources.

Procedure

1. Obtain the appropriate controller for the model of which you want to modify a resource.
2. Obtain a modifiable instance of the resource from that controller.
3. Start the appropriate methods of the modifiable instance to run the modifications.
4. Persist the modifications by using the `commit()` method of the controller.

Setting titles and descriptions:

You can set the titles and descriptions of a resource.

Procedure

1. Obtain a modifiable instance of the resource for which you want to set titles or descriptions.
2. Check whether the resource implements the `ModifiableLocalized` interface. To check, use the operator `instanceof`. If the resource does not implement the `ModifiableLocalized` interface, you cannot modify it.
3. Use the appropriate methods to set titles and descriptions. For example, if you want to set a title, use the `setTitle` method.

   Note: You cannot set a description in a particular locale without having a title that is set in that same locale.

Example

Modifying titles and descriptions:

```
// obtain modifiable instance of a model node
final Modifiable modifiable = controller.getModifiableNode(node);

// check if the resource implements ModifiableLocalized interface
if (modifiable instanceof ModifiableLocalized) {

    // set title and description
    ((ModifiableLocalized) modifiable).setTitle(Locale.GERMAN, "Titel");
    ((ModifiableLocalized) modifiable).setDescription(Locale.GERMAN, "Beschreibung");
}
```

Setting unique names:

You can set unique names for resources.

About this task

To set the unique name for a resource, proceed by the following steps:

Procedure

1. Obtain a modifiable instance of the resource for which you want to set the unique name.
2. Check whether the resource implements the `ModifiableIdentifiable` interface. To do this, use the operator `instanceof`. If the resource does not implement the `ModifiableIdentifiable` interface, you cannot modify it.
3. Obtain a modifiable instance of the resource object ID, that is `ModifiableObjectID`. To do this, use the `getModifiableObjectID`.
4. Set the unique name by using the `setUniqueName()` method of the `ModifiableObjectID`.

Example

Example - Setting unique names:

```
// obtain modifiable instance of a model node
final Modifiable modifiable = controller.getModifiableNode(node);

// obtain modifiable instance of the resource's object id; note that modifiable
// instances of all model nodes implement the ModifiableIdentifiable interface
final ModifiableObjectID modifiableObjectID = ((ModifiableIdentifiable)
modifiable).getModifiableObjectID();

// set unique name
modifiableObjectID.setUniqueName("MyUniqueName");
```

Setting metadata:

You can set metadata on all modifiable instances that implement the `ModifiableMetaDataProvider` interface.

About this task

To set metadata for a resource, proceed by the following steps:

Procedure

1. Obtain a modifiable instance of the resource for which you want to set metadata.
2. Check whether the resource implements the `ModifiableMetaDataProvider` interface. To do this, use the operator `instanceof`. If the resource does not implement the `ModifiableMetaDataProvider` interface, you cannot modify it.
3. Obtain a modifiable instance of the metadata of the resource.
4. Set the metadata by using the appropriate methods. For example, if you want to set metadata, use the `setValue` method.

Example

Example - Setting metadata for a resource:

```
// obtain modifiable instance of a model node
final Modifiable modifiable = controller.getModifiableNode(node);

// check if the resource implements ModifiableMetaDataProvider interface
```

```
if (modifiable instanceof ModifiableMetaDataProvider) {

    // obtain modifiable intance of the resource's meta data
    final ModifiableMetaData modifiableMetaData = ((ModifiableMetaDataProvider)
        modifiable).getModifiableMetaData();

    // set meta data
    modifiableMetaData.setValue("MyKey", "MyValue");
}
```

Setting supported markups:

You can set supported markups on modifiable instances that implement the `ModifiableMarkupCapable` interface.

About this task

For example, these can be the following:

- `ModifiableContentPage`
- `ModifiableContentLabel`
- `ModifiableContentURL`

Note: For content nodes, that is content pages, content labels, and content URLs, you need to set a supported markup. Otherwise they will not show in the read-only model.
To set the supported markups for a resource, proceed as follows:

Procedure

1. Obtain a modifiable instance of the resource for which you want to set supported markups.
2. Use the appropriate methods of the `ModifiableMarkupCapable` interface to set supported markups. For example, if you want to set a markup, use the `addMarkup` method.

Example

Example - Setting a supported markup on a layout control (error handling omitted):

```
// obtain markup list
final MarkupList markupList = ...;

// obtain modifiable instance of a layout control
final Modifiable modifiable = lmController.getModifiableNode(control);

// obtain markup object
Markup markup = markupList.getByName("html");

// set markup
((ModifiableMarkupCapable) modifiable).addMarkup(markup);
```

What to do next

For more detail about the markup list in the first line of the example and how to obtain it refer to “Obtain a model from the portal” on page 2867.

Setting the orientation for layout containers:

You can set the orientation for modifiable instances of `LayoutContainer` nodes.

About this task

To set the orientation for a layout container, proceed by the following steps

Procedure

1. Obtain a modifiable instance of a layout container.
2. Obtain the modifiable layout metrics instance.
3. Set the orientation by using the `setValue()` method.

Example

Example - Setting orientation of an existing layout container:

```
//  get modifiable instance of layout container
final ModifiableLayoutNode modifiable = (ModifiableLayoutNode) lmController.getModifiableNode(cont

//  get modifiable layout metrics and set the orientation
ModifiableLayoutMetrics modifiableLayoutMetrics = modifiable.getModifiableLayoutMetrics();
modifiableLayoutMetrics.setValue(LayoutMetrics.ORIENTATION, Orientation.HORIZONTAL);
```

What to do next

Note: If you do not set the orientation of a container, it is automatically set when you insert the container into the topology of the controller by the following rules:

- If a new root container is inserted into an empty topology, horizontal orientation is used for the new root container.
- If a new root container is inserted into a topology that is not empty, the new root container is set to the opposite orientation of the existing root container. For example, if the existing root container had horizontal orientation, the new root container will have vertical orientation.
- If a container is inserted as the child of an existing parent container, the child container is set to the opposite orientation of the parent container.

Setting portlet preferences:

You can set the portlet preferences for portlet definitions and portlet entities.

"Setting portlet preferences for portlet definitions"
You can set the portlet preferences for portlet definitions.

"Setting portlet preferences for portlet entities" on page 2898
You can set the portlet preferences for portlet entities.

Related concepts:

Preference layers and portlet modes
Get an overview of portlet configuration preference layers and portlet modes to best implement the different layers on which preferences for portlet views can be configured.

Setting portlet preferences for portlet definitions:

You can set the portlet preferences for portlet definitions.

About this task

To set the portlet preferences for a portlet definition, proceed by the following steps:

Procedure

1. Obtain a modifiable instance of a portlet definition.
2. Obtain the modifiable portlet preferences layer.
3. Set the preferences by using the appropriate methods of the `ModifiablePortletPreferences` interface.

Example

Example - Setting portlet preferences on a portlet definition:

```
// obtain portlet model controller
final PortletModelController pmController = getPortletModelController(portletModel);

// obtain modifiable instance of a portlet definition
final Modifiable modifiable = pmController.getModifiableNode(portletDefinition);

// obtain modifiable preferences layer
final ModifiablePortletPreferences preferences = ((ModifiablePortletDefinition)
      modifiable).getModifiablePortletPreferencesLayer();

// set a single value
preferences.setStringValue("name", "value");
```

Setting portlet preferences for portlet entities:

You can set the portlet preferences for portlet entities.

About this task

To set the portlet preferences for a portlet entity, proceed by the following steps:

Procedure

1. Obtain a modifiable instance of a portlet entity.
2. Obtain the modifiable portlet preferences layer.
3. Set the preferences by using the appropriate methods of the `ModifiablePortletPreferences` interface.

Example

Example - Setting portlet preferences on a portlet entity:

```
// obtain portlet model controller
final PortletModelController pmController = getPortletModelController(portletModel);

// obtain modifiable instance of a portlet entity
final Modifiable modifiable = pmController.getModifiableNode(portletEntity);

// obtain modifiable preferences layer
final ModifiablePortletPreferences preferences = ((ModifiablePortletEntity)
    modifiable).getModifiablePortletPreferencesLayer();

// set a single value
preferences.setStringValue("name", "value");
```

Setting flags:

The Controller SPI allows you to set flags for resources. For example, you can set a flag for a page so that portal users can bookmark the page.

About this task

You set flags for resources either on modifiable instances of those resources or on controllers managing the resources. The following flags are available:

Table 453. Flags that you can modify

Flag	Flag is associated to resource	Flag is set on
ActiveFlag	ContentNode (resources that implement the ActiveFlag interface)	ModifiableContentNode
BookmarkableFlag	ContentNode (resources that implement the BookmarkableFlag interface)	ModifiableContentNode
ShareableFlag	ContentNode (resources that implement the ShareableFlag interface)	ModifiableContentNode
AllPortletsAllowedFlag	ContentPage	LayoutModelController
DeletableFlag	LayoutNode	LayoutModelController
ModifiableFlag	LayoutNode	LayoutModelController

Notes:

1. ContentNode is the super class of ContentPage, ContentLabel, ContentURL, and InternalContentURL.
2. LayoutNode is the super class of LayoutContainer and LayoutControl.

In case of layered resources, for example derived content pages, you set flags only on the layer on which you work. To set flags, use the appropriate method. For example, to set the ActiveFlag, use the setActiveFlag method on the ModifiableContentNode instance. For more details about how the flags are aggregated in derivation scenarios refer to the Model SPI documentation.

Example

Example 1 - Setting the DeletableFlag for a layout container:

```
// set modifiable flag on the layout model controller
lmController.setDeletableFlag(container, true);
```

Example 2 - Setting the BookmarkableFlag for a content page:

```
 // obtain modifiable instance of an existing content page
    final Modifiable modifiable = cmController.getModifiableNode(page);
// set modifiable flag on the modifiable instance
    ((ModifiableBookmarkableFlag) modifiable).setBookmarkableFlag(true);
```

Related concepts:

"Model SPI overview" on page 2858
Models provide information that is needed by WebSphere Portal Express to perform tasks such as content aggregation or building navigation to browse the aggregated content. The information that is aggregated is represented through models that can be accessed programmatically by using the Model SPI (read-only). The information of a model is usually persistent (stored in a database) but can also be transient (computed and stored only in memory). Models can be represented by using a tree structure (nodes have a parent-child relationship), a list structure, or a selection structure (a selected element in a tree structure).

Setting themes:

The Controller SPI enables the setting of themes on modifiable instances that implement the ThemeSetter interface, for example ContentPage and ContentLabel.

Procedure

1. Obtain a modifiable instance of the resource for which you want to set a theme.
2. Obtain a `Theme` object from the `ThemeList` model.
3. Check whether the resource implements the `ThemeSetter` interface. Use the operator `instanceof`. If the resource does not implement the `ThemeSetter` interface, you cannot set the theme.
4. Use the `setTheme()` method to set the theme.

Example

Setting a theme for a content page (error handling omitted):

```
final Modifiable modifiable = cmController.getModifiableNode(page);

// obtain theme to set from com.ibm.portal.admin.ThemeList
final Theme theme = ...

// set theme
if (modifiable instanceof ThemeSetter) {
    ((ThemeSetter) modifiable).setTheme(theme);
}
```

Setting URLs:

You can set URLs to point to external content (`ExternalContentURL`) or to internal content, that is to nodes in the portal (`InternalContentURL`).

About this task

"Setting external content URLs"
Learn how to set URLs that point to external content (`ExternalContentURL`).

"Setting internal content URLs" on page 2901
Learn how to set URLs that point to portal internal content (`InternalContentURL`).

Setting external content URLs:

Learn how to set URLs that point to external content (`ExternalContentURL`).

Procedure

1. Obtain a modifiable instance of an external content URL.
2. Obtain the markup for which you want to set the URL.
3. Set the required URL for the modifiable instance of the external content URL.

Example

```
// obtain modifiable instance of an existing ContentURL
final Modifiable modifiable = cmController.getModifiableNode(contentUrl);

// identify target of url
final String target = ...

// identify markup
final Markup markup = ...

// set URL
((ModifiableContentURL) modifiable).setURL(target, markup);
```

Setting internal content URLs:

Learn how to set URLs that point to portal internal content (`InternalContentURL`).

Procedure

1. Obtain a modifiable instance of an internal content URL.
2. Identify the content node to which you want to assign the URL.
3. Obtain the markup for which you want to set the URL.
4. Set the required URL for the modifiable instance of the internal content URL.

Example

```
// obtain modifiable instance of an existing InternalContentURL
final Modifiable modifiable = cmController.getModifiableNode(internalContentUrl);

// identify target of url
final Identifiable identifiable = ...

// identify markup
final Markup markup = ...

// set URL
((ModifiableInternalContentURL) modifiable).setTarget(identifiable, markup);
```

Confirming modifications

Modifiable interfaces and Controller interfaces provide methods for confirming modifications. You can use these confirm methods to check whether modifications might be prevented by portal internal constraints.

About this task

The confirm methods do not perform a modification, but they only indicate whether a modification can be performed or not. For example, if an administrator has locked the layout of a page, you cannot move portlets on that page by using the Controller SPI or by any other method.

For every method of the Controller SPI that modifies portal resources, the Controller SPI also provides a corresponding method to confirm the respective modification. All confirm methods start with the prefix `confirm` followed by the name of the method which is to be confirmed.

Example

Example - Confirming placement of a portlet by confirming the movement of the control of that portlet:

```
// check if the control may be moved to the specified layout container
final boolean result = lmController.confirmInsert(control, container, null);

// check result
if (result == true) {
    // control may be moved to specified layout container
} else {
    // control may not be moved to specified layout container
}
```

Note: There is no guarantee that a modification will indeed be successful after the related `confirm`*xxx* method returns `true`. This is due to the fact that the underlying data structure might have changed between the confirmation and the actual modification.

Hints and tips for using the Controller SPI

These are some hints and tips for using the Controller SPI.

- Reusing unique names: If you use a controller to directly or indirectly remove a portal resource that has a unique name assigned to it and if you want to reuse that unique name for another resource later, you need to use a new controller to do so, unless you used the same controller for creating the resource and removing it again.
- Reusing an existing object ID is limited: A resource that reuses an object ID must be of the same type as the resource that used this object ID previously.

User and group management

The Portal User Management Architecture (PUMA) System programming interface (SPI) provides interfaces for accessing the profiles of a portal User or Group.

PUMA SPI is used to find, create, modify, and delete users and groups. Also, profile information about the currently logged in user can be retrieved. The User and Group interfaces that are returned by the SPI inherit the getObjectID() method from the com.ibm.portal.Identifiable interface of the Model SPI. This method is used to obtain the ObjectID that uniquely identifies a resource, in this case a user or group in the portal user registry. The following provider objects are used to access the User and Group objects.

PumaProfile
Contains methods that provide read-only access to the User and Group attributes and identifiers. You can use this interface to get the User object for the current user.

PumaLocator
Contains methods for looking up User and Group objects. You can use this interface to obtain a List of Group objects for all of the groups in which the current user is a member. Beginning with Version 7.0 of WebSphere Portal Express, paging is supported. Which means that the result set is split up into subsets (pages) and a special iterator can be used to access the pages.

PumaController
Contains methods for creating and deleting Users and Groups and for modifying the User and Group profiles and membership.

PumaEnvironment
Contains methods to retrieve virtual principals, access general properties for user management, and a method to bypass access control for the user and group management layer.

Before the portlet can use these provider objects, it must first retrieve the appropriate home interface, depending on the type of application.

Standard portlet
com.ibm.portal.um.portletservice.PumaHome

IBM portlet
com.ibm.portal.um.portletservice.legacy.PumaHome

Portal application (for example, theme or skin)
com.ibm.portal.um.PumaHome

Examples of how these interfaces are retrieved are provided in the accompanying Javadoc documentation. The following example shows how a standard portlet would obtain the identifier of a User as a String.

```
PortletServiceHome psh;
try{
   javax.naming.Context ctx = new javax.naming.InitialContext();
   psh = (PortletServiceHome)
         ctx.lookup("portletservice/com.ibm.portal.um.portletservice.PumaHome");
   if (psh != null){
      PumaHome service = (PumaHome) psh.getPortletService(PumaHome.class);
      PumaProfile pp = service.getProfile(request);
      User user = pp.getCurrentUser();
      String user_objid = pp.getIdentifier(user);
   }
}
catch (PumaException pe){
   // ... error handling ...
} catch(javax.naming.NameNotFoundException ex) {
   // ... error handling ...
} catch(javax.naming.NamingException ex) {
   // ... error handling ...
}
```

Note: Because PumaProfile, PumaLocator and PumaController store the current user, you must not store these objects in a session or anywhere else. But you can retrieve them from PumaHome each time you use them. PumaHome, however, can be stored.

The following sample shows how a standard portlet would do a standard search for reading and writing attributes:

```
List<User> usersStartingWithA = pumaLocator.findUsersByAttribute("uid", "a*");
// if no value for ibm-primaryEmail attribute is set, then set it
List<String> requestedAttributes = new ArrayList<String>(2);
requestedAttributes.add("uid");
requestedAttributes.add("primaryEmail");
for(User user: usersStartingWithA) {
Map<String, Object> attributes =
pumaProfile.getAttributes(user, requestedAttributes);
if (attributes.get("primaryEmail")==null
|| "".equals(attributes.get("ibm-primaryEmail"))){
pumaController.setAttributes(user, Collections.singletonMap(
"ibm-primaryEmail", attributes.get("uid")+"@ibm.com"));
}
}
```

The following sample shows how to do a paged search:

```
// create a properties map that requests 10 results per Page
Map<String, Object> pageProperties = new HashMap<String, Object>(2);
pageProperties.put(PumaLocator.RESULTS_PER_PAGE, 10);
PagingIterator<User> pageIter = pumaLocator.findUsersByAttribute(
"uid", "a*", pageProperties);
List<User> buffer = new ArrayList<User>(10);
do {
pageIter.getNextPage(buffer); //=> always has a first page
if (pageIter.getCurrentPageNumber()==0)
{
System.out.println("Total results: "+pageIter.getNumberOfTotalResults());
System.out.println("Total pages: "+pageIter.getNumberOfPages());
}
for (User aUser: buffer)
{ display(aUser);
}
```

```
} while (pageIter.hasNextPage());
// now jump to page 5
// => will throw NoSuchPageException if NumberOfPages < 6
List<User> result = pageIter.getPage(null, 5);
```

"Remote REST service for PUMA"
The remote PUMA SPI gives you access to user profiles through REST services. It provides a remote interface for user and group management for the configured Portal user repository. It is based on the REST (REpresentational State Transfer) architecture model.

Remote REST service for PUMA

The remote PUMA SPI gives you access to user profiles through REST services. It provides a remote interface for user and group management for the configured Portal user repository. It is based on the REST (REpresentational State Transfer) architecture model.

The remote REST service for PUMA allows you to remotely perform the following tasks related to user and group data based on the HTTP protocol:

- Access, retrieve, and modify information about available user or group attributes and their metadata
- Search for users or groups based on attributes or group membership
- Create, update, or delete user or group profiles and group membership relations.

The first part of this documentation describes the basic interface that lists all possible operations, including their parameters and a minimum schema to describe the resources that are involved. The second part explains specific details for the current implementation of this interface for this version of WebSphere Portal Express.

"Structure of the remote REST service for PUMA"
The interface provided by the remote REST service for PUMA defines single operations that are characterized by a particular URI path, the HTTP method, the expected input or output, and a list of query parameters. With regards to the input or output format, the interface only describes a common baseline for the payload, which can be wrapped or represented individually by different implementations of the service.

"How the portal implements the remote PUMA REST service" on page 2915
The previous topics describe the general part of the remote REST service for PUMA Interface. This interface can be implemented by different providers that can be based on different backend systems or user repositories and provide their own input and output formats. The implementation of the interface described here is the one based on WebSphere Portal Express Version 8.5.

Structure of the remote REST service for PUMA

The interface provided by the remote REST service for PUMA defines single operations that are characterized by a particular URI path, the HTTP method, the expected input or output, and a list of query parameters. With regards to the input or output format, the interface only describes a common baseline for the payload, which can be wrapped or represented individually by different implementations of the service.

All URI paths that the interface defines start with the prefix `/um`. This can be considered as the identifying part of the service. Implementations of the service can add additional path elements before the `/um` element, for example to represent the context root of a servlet. Therefore clients should not assume that the complete servlet path always starts with `/um`.

URI paths that start with `/um/secure` will be served only within a valid user context, that is only in an authenticated request. If the `/secure` path element is omitted, this means that the operation is performed in the context of the anonymous user. For simplicity, the interface description denotes all URI paths without the `/secure` element. However, all corresponding operations can be performed for both authenticated and anonymous users. For operations related to authenticated users you have to prefix the `/secure` element. This also implies that the particular implementation of the interface has to make sure to apply the appropriate access control checks before executing an operation.

Variable parts within URI paths or query parameters can contain special characters. These must be encoded in order to represent valid path elements or parameters. The interface defines that UTF-8 must be used by the client and that URLs returned by the server are UTF-8 encoded.

Some operations of the interface make use of HTTP PUT and DELETE operations, but many environments only allow GET or POST operations for HTTP requests for security reasons. To be compatible to such scenarios, the interface defines a general replacement mechanism for PUT and DELETE operations by POST requests:

- Request parameter `postAction` with a value of `put` or `delete`. The case is ignored.
- Request header `X-Method-Override` with a value of `put` or `delete`. The case is ignored.

The implementation must make sure that it is possible to enable or disable this tunneling of request methods as needed. For each operation, you can specify the input and output format. You can do this by using either the request parameter `mime-type` or by using the `accept` request header as defined in the HTTP specification RFC 2616. If you set the request parameter, the information from the accept header is ignored.

"Interface operations" on page 2906
View all operations of the remote REST service for PUMA including the necessary attributes and a description.

"Payload description" on page 2909
The actual data that is processed by the remote REST service for PUMA that is the attributes and their values, user or group profiles, and membership lists, is described by an XML schema document. This schema is normative for all kinds of input and output formats. Therefore, for representations that are not based on XML, such as JSON, you need to apply an appropriate transformation.

"PUMA REST service XML schema document" on page 2911
The data processed by the remote REST service for PUMA is described by an XML schema document. View the XML schema for the PUMA REST service XML schema document.

"Error codes" on page 2913
If the operations described previously fail for some reason, the remote REST service for PUMA returns a subset of the HTTP error codes. They include a message with a detailed description of the error.

"Data types for attributes" on page 2914
Data types for attributes belong to a subset of the XML Schema data type specification.

Related information:

http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html#sec14.1

Interface operations:

View all operations of the remote REST service for PUMA including the necessary attributes and a description.

The following list shows and describes all the possible URI path operations, together with the related query parameters. Note that some URI path operations apply to different HTTP methods - GET, or PUT and POST, or DELETE.

Notational convention: Parameter values are separated by vertical slashes (|). Default values are indicated with an asterisk (*).

/um/attributes/users

For the HTTP method GET:
This URI path operation returns the list of references to the attribute definitions that are available for users with references to the URI path operation `/um/attributes/users/`*`attribute name`*. This URI path operation has the following query parameter:

expandRefs=true | *false
If you set this parameter to `true`, the representations of the attribute definitions are embedded in the list of references. The default value is `false`.

/um/attributes/users/*attribute name*

For the HTTP method GET:
This URI path operation returns a representation of the attribute definition. For details refer to the topic about payload description. This URI path operation has no query parameter.

/um/attributes/groups

For the HTTP method GET:
This URI path operation returns the list of references to the attribute definitions that are available for groups with references to the URI path operation `/um/attributes/groups/`*`attribute name`*. This URI path operation has the following query parameter:

expandRefs=true|*false
If you set this parameter to `true`, the representations of the attribute definitions are embedded in the list of references. The default value is `false`.

/um/attributes/groups/*attribute name*

For the HTTP method GET:
This URI path operation returns a representation of the attribute definition. For details refer to the topic about payload description. This URI path operation has no query parameter.

/um/currentuser/profile

For the HTTP method GET:
This URI path operation returns the profile of the current user. If this operation is performed without authentication, the profile of the anonymous user is returned. This URI path operation has no query parameter.

For the HTTP methods PUT or POST:
This URI path operation updates the user profile with the profile

representation contained in the request body. Only the attributes provided in this representation are considered, therefore post only the attributes that you want to change. This URI path operation has the following query parameters:

update=*replace | merge | delete
This defines how the update is performed. Valid values are as follows:

replace
All attribute values provided in the update replace the existing ones. This is the default.

merge The new attribute values are merged with existing ones. This is only relevant for multi-value attributes.

delete The values of all attributes provided in the update are removed from the profile. Consequently, the actual attribute values given in the update are ignored.

/um/users/profiles
/um/groups/profiles

`For the HTTP method GET:`
This URI path operation returns the list of references to all user or group profiles that correspond to the search criteria and other parameters. If there are no limiting parameters, all available users or groups are returned. The list that is returned is access control filtered for the current user. This URI path operation has the following query parameters:

expandRefs=true | *false
If you set this to `true`, the complete profiles are embedded in the references. The default is `false`.

includeAttributes
The attributes contained in the profiles are limited to those specified in the comma-separated list provided by this attribute. By default, if you omit this parameter, the basic set of attributes is returned for users, or the minimum set of attributes is returned for groups.

memberOf=*unique ID of group*
The response contains only profiles of users or groups that a are member of the specified group.

showNested=true | *false
If you set the `memberOf` parameter, this parameter specifies whether nested groups are considered or not. If you set the parameter to `false`, only direct membership is evaluated. The default value is `false`.

searchAttributes
Use this query parameter to define a search string that specifies various combinations of attribute values as search criteria. All implementations have to support at least values such as *`attribute name`*`%3D`*`attribute value`*, where the attribute value can contain an asterisk (*) as a wildcard character.

identifier
The returned list will contain only the one user or group with

the specified unique identifier. The implementation decides what is to be used as the unique identifier. For example, this can be the distinguished name.

resultsPerPage
The returned list will contain only the given number of results. Additionally, links to additional result pages (first, last, next, previous) will be included in the response, if available.

sortByAttributes
Comma-separated list that specifies the sort order for the results. This is only supported for a paged search.

descending=true | *false
Determines if the sorting according to **sortByAttributes** will be descending.

Note: The parameters `memberOf`, `searchAttributes`, and `identifier` are mutually exclusive. Paged search (resultsPerPage) only works for searchAttributes.

For the HTTP method POST:
This URI path operation creates a new user or group by posting a representation of the related profile. For details refer to the topic about the payload description. This URI path operation has no query parameter.

/um/users/profiles/*unique ID of user*
/um/groups/profiles/*unique ID of group*

For the HTTP method GET:
This URI path operation returns a representation of the user or group profile. For details refer to the topic about the payload description. This URI path operation has the following query parameter:

includeAttributes
The attributes contained in the profiles are limited to those specified in the comma-separated list provided by this attribute. By default, if you omit this parameter, all attributes that hold values are returned.

For the HTTP methods PUT or POST:
This URI path operation updates the user or group profile with the profile representation contained in the request body. Only the attributes provided in this representation are considered, therefore post only the attributes that you want to change. This URI path operation has the following query parameters:

update=*replace | merge | delete
This defines how the update is performed. Valid values are as follows:

replace
All attribute values provided in the update replace the existing ones. This is the default.

merge The new attribute values are merged with existing ones. This is only relevant for multi-value attributes.

delete The values of all attributes provided in the update are removed from the profile. Consequently, the actual attribute values given in the update are ignored.

For the HTTP method DELETE:
This URI path operation deletes the user or group. This URI path operation has no query parameter.

/um/groupmembership/*unique ID of user*
/um/groupmembership/*unique ID of group*

For the HTTP method GET:
This URI path operation returns the list of references to all group profiles of the groups of which the user or the group is a member. Refer to the URI path operations `/um/users/profiles/`*`unique ID of user`* and `/um/group/profiles/`*`unique ID of group`* listed previously. This is also called the membership list; for details refer to the topic about the payload description. This URI path operation has the following query parameters:

expandRefs=true|*false
If you set this to `true`, the complete profiles are embedded in the references. The default is `false`.

includeAttributes
The attributes contained in the profiles are limited to those specified in the comma-separated list provided by this attribute. By default, if you omit this parameter, the basic set of attributes is returned for users, or the minimum set of attributes is returned for groups.

showNested=true|*false
This parameter specifies whether nested groups are considered or not. If you set the parameter to `false`, only direct membership is evaluated. The default value is `false`.

For the HTTP methods PUT or POST:
This URI path operation updates the existing membership list for the user or group by the one that is posted. This URI path operation has the following query parameters:

update=*replace|merge|delete
This defines how the update is performed. Possible values are as follows:

replace
The complete membership list will be replaced by the one posted. This is the default.

merge The user or group are added to the groups contained in the posted membership list in addition to the existing group membership relations of the user or group.

delete The user or group will be removed from all the groups contained in the posted list only.

For the HTTP method DELETE:
This URI path operation deletes the membership list for the user or group; this means that it removes the user or group from all groups. This URI path operation has no query parameter.

Payload description:

The actual data that is processed by the remote REST service for PUMA that is the attributes and their values, user or group profiles, and membership lists, is

described by an XML schema document. This schema is normative for all kinds of input and output formats. Therefore, for representations that are not based on XML, such as JSON, you need to apply an appropriate transformation.

For the complete schema document, refer to the topic that contains the PUMA REST Service XML schema document.

The following sections describe the particular parts of the XML schema in detail.

Attributes and attribute values

The `attribute` element is used to represent either an attribute definition when it is used alone, or an instance of the attribute that includes one or more values when it is used inside a user or group profile. The attribute is identified by the mandatory `name` attribute and can optionally define its type and a flag that specifies whether multiple values are allowed. By convention, the values for the type attribute must be those data types that are defined in the XSD data types specification. For more information, see the section about data types. The attribute values are represented as separate elements called `attributeValue` that can contain arbitrary character data.

```
<xs:element name="attribute">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="attributeValue" minOccurs="0" maxOccurs="unbounded"/>
         </xs:sequence>
            <xs:attribute name="name" type="xs:string" use="required"/>
            <xs:attribute name="type" type="xs:string" use="optional"/>
            <xs:attribute name="multiValued" type="xs:boolean" use="optional" default="false"/>
      </xs:complexType>
   </xs:element>
<xs:element name="attributeValue" type="xs:string">
```

The following is an example of a stand-alone attribute definition that is represented in plain XML:

```
<um:attribute xmlns:um="um" name="ibm-hobby" type="xs:string" multiValued="true"/>
```

The usage of the `attribute` element for representing attribute values is shown in the next section.

Profile data

Profiles are represented in a `profile` element that needs to define the type of the profile as restricted to the values `user` or `group` in an attribute. Wrapped inside the `profile` element there can be a sequence of arbitrary length of `attribute` elements. In this case, these `attribute` elements must contain attribute value elements.

```
<xs:element name="profile">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="attribute" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="type" type="profileType" use="required"/>
        <xs:attribute name="identifier" type="xs:string" use="optional"/>
    </xs:complexType>
</xs:element>
<xs:simpleType name="profileType">
    <xs:restriction base="xs:string">
        <xs:enumeration value="user"/>
        <xs:enumeration value="group"/>
    </xs:restriction>
</xs:simpleType>
```

The following is an example of a user profile according to this schema:

```
<um:profile type="user" identifier="uid=bob,o=defaultWIMFileBasedRealm">
    <um:attribute name="uid" type="xs:string" multiValued="false">
        <um:attributeValue>bob</um:attributeValue>
    </um:attribute>
    <um:attribute name="cn" type="xs:string" multiValued="false">
        <um:attributeValue>Bob</um:attributeValue>
    </um:attribute>
    <um:attribute name="ibm-hobby" type="xs:string" multiValued="true">
        <um:attributeValue>Running</um:attributeValue>
        <um:attributeValue>Baseball</um:attributeValue>
    </um:attribute>
</um:profile>
```

Group membership list

The groups membership list lists all groups of which a particular user or group is a member. The representation of the group membership list uses a special element `profileRef` that is used to represent a reference to a profile. This element contains a reference to the resource that is described by the profile; it can optionally also contain the complete profile element itself. This action is done by the mandatory `uri` attribute, which must contain the URL path of the profile, for example `/um/secure/groups/profiles/myGroupId`. The profile reference elements are then aggregated to a list within the `groupMembershipList` element. The list points to those groups of which the user or group that is defined by the resource URL is a member.

```
<xs:element name="profileRef">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="profile" minOccurs="0⌂" maxOccurs="1"/>
        </xs:sequence>
        <xs:attribute name="uri" type="xs:string" use="required"/>
    </xs:complexType>
</xs:element>
<xs:element name="groupMembershipList">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="profileRef" maxOccurs="unbounded" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
```

The following listing shows an example of a group membership list XML representation that is based on the schema:

```
<um:groupMembershipList>
    <um:profileRef uri="/wps/um/secure/groups/profiles/8eAeK . . . . . BQIMK5D1"/>
    <um:profileRef uri="/wps/um/secure/groups/profiles/8eAeK . . . . . FADFJABC"/>
</um:groupMembershipList>
```

Related information:

http://www.w3.org/2001/XMLSchema-datatypes

PUMA REST service XML schema document:

The data processed by the remote REST service for PUMA is described by an XML schema document. View the XML schema for the PUMA REST service XML schema document.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
 * IBM Confidential
 *
```

```
 * OCO Source Materials
 *
 * (C) Copyright IBM Corp. 2002, 2006
 *
 * The source code for this program is not published or otherwise
 * divested of its trade secrets, irrespective of what has been
 * deposited with the U.S. Copyright Office.
 *
-->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           targetNamespace="http://www.ibm.com/xmlns/prod/websphere/um.xsd"
           xmlns="http://www.ibm.com/xmlns/prod/websphere/um.xsd"
           elementFormDefault="qualified">
  <xs:element name="attribute">
    <xs:annotation>
      <xs:documentation>The element that represents an attribute definition.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="attributeValue" minOccurs="0" maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation>This element represents one value for the wrapping attribute.
              It can be sequenced in arbitrary length for multi-valued attributes.
              If the attribute element is only used to describe the attribute definition that is not
              part of a profile, there is no attributeValue element wrapped inside.</xs:documentation>
        </xs:annotation>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="name" type="xs:string" use="required">
        <xs:annotation>
          <xs:documentation>This XML attribute is used to specify the name that identifies the attribute.
          </xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="type" type="xs:string" use="optional">
        <xs:annotation>
          <xs:documentation>This XML attribute is used to describe the type of the attribute.
              The values correspond to the data types specified by the XML Schema data type definitions
              described by http://www.w3.org/2001/XMLSchema-datatypes.
              The actual attribute types are part of the server configuration and can not be changed by using
              this XML attribute, but are only used for description purposes.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="multiValued" type="xs:boolean" use="optional" default="false">
        <xs:annotation>
          <xs:documentation>This XML attribute specifies whether the attribute can have
              multiple values or can have only one value.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:element name="attributeValue" type="xs:string">
  <xs:annotation>
    <xs:documentation>This element wraps a single attribute value. The value itself is represented
        by all character data inside the element, using the corresponding string representation,
        depending on the attribute type.</xs:documentation>
  </xs:annotation>
  </xs:element>
  <xs:element name="profile">
  <xs:annotation>
    <xs:documentation>This element represents a user or group profile.</xs:documentation>
  </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="attribute" maxOccurs="unbounded">
          <xs:annotation>
            <xs:documentation>The profile can contain an arbitrary number of attributes
                and wrapped attribute value elements.</xs:documentation>
          </xs:annotation>
        </xs:element>
```

```
      </xs:sequence>
      <xs:attribute name="type" type="profileType" use="required">
        <xs:annotation>
          <xs:documentation>Denotes whether the profile represents a user or group profile.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="identifier" type="xs:string" use="optional">
        <xs:annotation>
          <xs:documentation>Denotes the unique identifier of the principal, e.g. the DN</xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:simpleType name="profileType">
    <xs:annotation>
      <xs:documentation>Specifies the list of values that can be used to define the type of a profile.
      Currently, user and group profiles are distinguished.</xs:documentation>
      </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:enumeration value="user"/>
      <xs:enumeration value="group"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:element name="profileRef">
    <xs:annotation>
      <xs:documentation>This element represents a reference to a profile.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="profile" minOccurs="0">
          <xs:annotation>
            <xs:documentation>The profile reference can contain a full representation of the profile itself.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="uri" type="xs:string" use="required">
        <xs:annotation>
          <xs:documentation>The relative URI that points to the resource that represente the profile.
          </xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:element name="groupMembershipList">
    <xs:annotation>
      <xs:documentation>This element represents a list of profile references to all groups
          of which a particular user or group is member.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="profileRef" maxOccurs="unbounded" minOccurs="0">
          <xs:annotation>
            <xs:documentation>The groupMembershipList can contain an arbitrary number
                of references to group profiles.</xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Error codes:

If the operations described previously fail for some reason, the remote REST service for PUMA returns a subset of the HTTP error codes. They include a message with a detailed description of the error.

The following list gives the error codes that the remote REST service for PUMA can return, and typical situations in which each specific error code is returned.

400 - Bad Request
The input provided in the request body does not comply to the expected format. For example, this can be invalid XML in a posted profile.

401 - Unauthorized
Access control check during request processing failed.

403 - Forbidden
The operation that the client tries to perform is not possible. For example, this can be creating a user ID that already exists, setting attribute values for attributes that are not defined, or using mutually exclusive request parameters.

404 - Not Found
The URI does not match any of the defined URI paths or a variable part of a defined URI path does not denote a resource that exists.

405 - Method Not Allowed
The request addresses a defined URI, but uses an HTTP method that is not defined for this URI.

415 - Unsupported Media Type
The format specified in the mime-type URI parameter or accept headers is not supported.

Data types for attributes:

Data types for attributes belong to a subset of the XML Schema data type specification.

You can use the following of data types with the PUMA REST API:

xs:string
String value or string representation of an arbitrary object.

xs:long
Number of type Long.

xs:integer
Number of type Integer.

xs:double
Number of type Double.

xs:dateTime
A timestamp with this format: `yyyy-MM-dd'T'HH:mm:ss`

xs:anyURI
URI path to a resource defined by the remote REST service for PUMA, for example to reference a user profile. This can be an absolute URL that includes the server name or a URI that denotes the absolute path.

xs:hexBinary
A hex encoded binary string.

Note: The Puma REST API converts the preceding data type names to conform with the XML Schema Definition (XSD) language.

Related information:

http://www.w3.org/2001/XMLSchema-datatypes

How the portal implements the remote PUMA REST service

The previous topics describe the general part of the remote REST service for PUMA Interface. This interface can be implemented by different providers that can be based on different backend systems or user repositories and provide their own input and output formats. The implementation of the interface described here is the one based on WebSphere Portal Express Version 8.5.

The implementation described here uses ATOM (refer to RFC 4287) and the ATOM Publishing Protocol as basic input and output format to wrap the resource representations described by the schema in the general part. Additionally, using the remote REST service for PUMA requires that you consider some special aspects described in this section.

"URL path segment for virtual portals" on page 2916
WebSphere Portal Express provides the concept of virtual portals that allows you to manage several separate portals within one portal installation. You can associate each of these virtual portals with a user realm and thereby limit the scope of users or groups that can access it to one realm of the underlying user repository.

"Identifiers used in the Portal Implementation" on page 2916
For the unique identifier in the URLs, the portal uses a string representation of the internal identifier that is considered to be opaque to the client. The identifier that is used for retrieval of users or groups, that is by either the `identifier` or the `memberOf` request parameters, is the unique security name that is associated to the security context in the WebSphere Application Server. The exact matching of the unique security name is part of the security configuration, but in most scenarios it matches the distinguished name (DN). The same identifier is of course also used for the `identifier` attribute in the profile element.

"Access Control Checks" on page 2916
The portal does not allow you to set permissions for attribute definitions. Therefore the remote REST service for PUMA allows requests to some operations only for authenticated users.

"Using ATOM/APP as input and output format" on page 2917
The remote REST service for PUMA uses the ATOM Publishing Protocol (APP) as the primary input and output format. It wraps the elements described by the schema document in the remote REST service for PUMA in appropriate ATOM feed or entry documents. Although this is the default input and output format, the client should specify the mime type `application/atom+xml` either in the `mime-type` request parameter or in the accept header. A more detailed description of how the APP maps to the RESTful interface and some examples are given here.

"Switch for tunneling of HTTP methods" on page 2920
The portal implementation allows you to simulate PUT and DELETE requests by tunneling, that is by using POST requests instead.

"HTTP caching" on page 2921
The remote REST service for PUMA sets the `Cache-Control` HTTP header in the response to `public` for resources that are served without authentication, and to `private` for URIs that require authentication.

"Context root and authentication mechanism" on page 2921
The remote REST service for PUMA is implemented as a servlet that runs as a separate enterprise application on the WebSphere Portal Express server.

"Lookup facility in the portal" on page 2921
For both convenience and alignment to other portal REST services, the portal remote REST service for PUMA offers a lookup facility. This is done by plugging a provider into a reusable lookup facility in the portal. This functionality allows you to retrieve particular URLs of the service by specifying an absolute URI as a parameter to the so-called lookup servlet addressed by `/wps/poc`.

URL path segment for virtual portals:

WebSphere Portal Express provides the concept of virtual portals that allows you to manage several separate portals within one portal installation. You can associate each of these virtual portals with a user realm and thereby limit the scope of users or groups that can access it to one realm of the underlying user repository.

To address this case, the portal remote REST service for PUMA allows you to specify a particular virtual portal for each operation. This operation is then limited to the user realm of that virtual portal. This applies to both of the following:

- The result of the operation. For example, search results are limited to the realm.
- Security checks. The user performing the operation must be member of the realm that is implicitly addressed.

The virtual portal information is represented by two additional path elements `/vp/`*`virtul_portal_url_mapping`* that follow the URL path element `/um` or, respectively, the `/um/secure` path element. When no particular virtual portal is specified, the implementation uses the default virtual portal. For example, if the portal installation defines a virtual portal with the URL mapping `sales` for the user realm SalesPersons, the URL `/wps/um/secure/vp/sales/users/profiles` returns the references to all user profiles in the SalesPersons user realm.

Identifiers used in the Portal Implementation:

For the unique identifier in the URLs, the portal uses a string representation of the internal identifier that is considered to be opaque to the client. The identifier that is used for retrieval of users or groups, that is by either the `identifier` or the `memberOf` request parameters, is the unique security name that is associated to the security context in the WebSphere Application Server. The exact matching of the unique security name is part of the security configuration, but in most scenarios it matches the distinguished name (DN). The same identifier is of course also used for the `identifier` attribute in the profile element.

Access Control Checks:

The portal does not allow you to set permissions for attribute definitions. Therefore the remote REST service for PUMA allows requests to some operations only for authenticated users.

The portal remote REST service for PUMA allows requests to the following operations only for authenticated users by using the `/um/secure` path element:

- `/um/attributes/users`
- `/um/attributes/users/`*`attribute name`*
- `/um/attributes/groups`
- `/um/attributes/groups/`*`attribute name`*

For all other operations, the implementation performs permission checks by using the corresponding Portal Access Control (PAC) settings for users and groups.

Using ATOM/APP as input and output format:

The remote REST service for PUMA uses the ATOM Publishing Protocol (APP) as the primary input and output format. It wraps the elements described by the schema document in the remote REST service for PUMA in appropriate ATOM feed or entry documents. Although this is the default input and output format, the client should specify the mime type `application/atom+xml` either in the `mime-type` request parameter or in the accept header. A more detailed description of how the APP maps to the RESTful interface and some examples are given here.

For details about the APP refer to the draft for the protocol on the internet.

Representing list results as ATOM feeds

All the REST service operations that return lists of resources return an ATOM feed document that represents each portal resource as a single ATOM entry. This applies to the GET cases for the operations `/um/attributes/users`, `/um/attributes/groups`, `/um/users/profiles`, and `/um/groups/profiles`. Depending on the value of the `expandRefs` parameter, the ATOM entries contain a `content` section that embeds the XML representation of the resource according to the schema defined in the interface.

Example 1: This example shows a server response to the `/um/secure/attributes/groups` URI path:

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:feed xmlns:xs="http://www.w3.org/2001/XMLSchema-datatypes"
           xmlns:um="http://www.ibm.com/xmlns/prod/websphere/um.xsd"
           xmlns:atom="http://www.w3.org/2005/Atom"
           xmlns:xhtml="http://www.w3.org/1999/xhtml">
    <atom:title>Available group attributes</atom:title>
    <atom:author>
        <atom:name>IBM WebSphere Portal</atom:name>
    </atom:author>
    <atom:link href="/wps/um/secure/attributes/groups" rel="self"/>
    <atom:id>um:secure/attributes/groups</atom:id>
    <atom:updated>2006-12-16T17:06:35.609Z</atom:updated>
    <atom:entry>
        <atom:title>cn</atom:title>
        <atom:link href="/wps/um/secure/attributes/groups/cn" rel="self"/>
        <atom:id>um:secure/attributes/groups/cn</atom:id>
        <atom:updated>2006-12-16T17:06:35.609Z</atom:updated>
    </atom:entry>
    <atom:entry>
        <atom:title>description</atom:title>
        <atom:link href="/wps/um/secure/attributes/groups/description" rel="self"/>
        <atom:id>um:secure/attributes/groups/description</atom:id>
        <atom:updated>2006-12-16T17:06:35.609Z</atom:updated>
    </atom:entry>
    <atom:entry>
        <atom:title>createTimestamp</atom:title>
        <atom:link href="/wps/um/secure/attributes/groups/createTimestamp" rel="self"/>
        <atom:id>um:secure/attributes/groups/createTimestamp</atom:id>
        <atom:updated>2006-12-16T17:06:35.609Z</atom:updated>
    </atom:entry>
    <atom:entry>
        <atom:title>modifyTimestamp</atom:title>
        <atom:link href="/wps/um/secure/attributes/groups/modifyTimestamp" rel="self"/>
        <atom:id>um:secure/attributes/groups/modifyTimestamp</atom:id>
        <atom:updated>2006-12-16T17:06:35.609Z</atom:updated>
    </atom:entry>
</atom:feed>
```

Note: The `atom:updated` does not contain any useful information, but it has to be added to comply to the ATOM specification.

Example 2: The same URI including the option to expand the references, that is /um/secure/attributes/groups?expandRefs=true, gives the following server response:

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:feed xmlns:xs="http://www.w3.org/2001/XMLSchema-datatypes"
           xmlns:um="http://www.ibm.com/xmlns/prod/websphere/um.xsd"
           xmlns:atom="http://www.w3.org/2005/Atom"
           xmlns:xhtml="http://www.w3.org/1999/xhtml">
    <atom:title>Available group attributes</atom:title>
    <atom:author>
        <atom:name>IBM WebSphere Portal</atom:name>
    </atom:author>
    <atom:link href="/wps/um/secure/attributes/groups?expandRefs=true" rel="self"/>
    <atom:id>um:secure/attributes/groups%3FexpandRefs%3Dtrue</atom:id>
    <atom:updated>2006-12-16T17:15:23.391Z</atom:updated>
    <atom:entry>
        <atom:title>cn</atom:title>
        <atom:link href="/wps/um/secure/attributes/groups/cn" rel="self"/>
        <atom:id>um:secure/attributes/groups/cn</atom:id>
        <atom:updated>2006-12-16T17:15:23.391Z</atom:updated>
        <atom:content type="application/xml">
            <um:attribute xmlns:um="um" name="cn"
                          type="xs:string" multiValued="false"/>
        </atom:content>
    </atom:entry>
    <atom:entry>
        <atom:title>description</atom:title>
        <atom:link href="/wps/um/secure/attributes/groups/description" rel="self"/>
        <atom:id>um:secure/attributes/groups/description</atom:id>
        <atom:updated>2006-12-16T17:15:23.391Z</atom:updated>
        <atom:content type="application/xml">
            <um:attribute xmlns:um="um" name="description"
                          type="xs:string" multiValued="true"/>
        </atom:content>
    </atom:entry>
    <atom:entry>
        <atom:title>createTimestamp</atom:title>
        <atom:link href="/wps/um/secure/attributes/groups/createTimestamp" rel="self"/>
        <atom:id>um:secure/attributes/groups/createTimestamp</atom:id>
        <atom:updated>2006-12-16T17:15:23.391Z</atom:updated>
        <atom:content type="application/xml">
            <um:attribute xmlns:um="um" name="createTimestamp"
                          type="xs:dateTime" multiValued="false"/>
        </atom:content>
    </atom:entry>
    <atom:entry>
        <atom:title>modifyTimestamp</atom:title>
        <atom:link href="/wps/um/secure/attributes/groups/modifyTimestamp" rel="self"/>
        <atom:id>um:secure/attributes/groups/modifyTimestamp</atom:id>
        <atom:updated>2006-12-16T17:15:23.391Z</atom:updated>
        <atom:content type="application/xml">
            <um:attribute xmlns:um="um" name="modifyTimestamp"
                          type="xs:dateTime" multiValued="false"/>
        </atom:content>
    </atom:entry>
</atom:feed>
```

Representing resources as ATOM entries

All particular resources are represented in an ATOM entry document. This applies to the GET cases for the operations /um/attributes/users/*attribute name*, /um/attributes/groups/*attribute name*, /um/currentuser/profile, /um/users/profiles/*unique_id_of_user*, /um/groups/profiles/*unique_id_of_group*, /um/groupmembership/*unique_id_of_user*, and /um/groupmembership/*unique_id_of_group*. This means that the XML structure is wrapped in the atom:content element of one atom:entry element. This way useful meta information, such as the link to the resource and an

ID can be returned together with the resource itself. Note that the group membership list is also treated as a single resource that contains the group membership information of one user or group.

The following example retrieves a user profile directly by using the ATOM based output. It shows a useful convenience feature of the implementation: as ATOM allows you to add several `atom:link` elements with certain types, the `related` link element can be used for a shortcut to the group membership URL of the same user. This shortcut is provided for user and group profiles and allows the client to directly follow this link instead of constructing the URL from static and dynamic parts.

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:entry xmlns:xs="http://www.w3.org/2001/XMLSchema-datatypes"
            xmlns:um="http://www.ibm.com/xmlns/prod/websphere/um.xsd"
            xmlns:atom="http://www.w3.org/2005/Atom">
   <atom:title>wpsadmin</atom:title>
   <atom:link href="/wps/um/secure/users/profiles/9eAe . . . K5D1" rel="self"/>
   <atom:link href="/wps/um/secure/groupmembership/9eA . . . K5D1" rel="related"/>
   <atom:id>um:secure/users/profiles/9eAeK2IIK9L59QKQ2 . . . K5D1</atom:id>
   <atom:updated>2006-12-16T17:34:04.406Z</atom:updated>
   <atom:content type="application/xml">
      <um:profile type="user" identifier="uid=wpsadmin,o=defaultWIMFileBasedRealm">
         <um:attribute name="uid" type="xs:string" multiValued="false">
            <um:attributeValue>wpsadmin</um:attributeValue>
         </um:attribute>
         <um:attribute name="cn" type="xs:string" multiValued="false">
            <um:attributeValue>wpsadmin</um:attributeValue>
         </um:attribute>
      </um:profile>
   </atom:content>
</atom:entry>
```

Result for a paged search

The result for a paged search will contain 3 openSearch elements, a link to the first and last page and if available links to previous and next page.

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:feed xmlns:atom="http://www.w3.org/2005/Atom"
xmlns:xs="http://www.w3.org/2001/XMLSchema-datatypes"
xmlns:um="http://www.ibm.com/xmlns/prod/websphere/um.xsd"
xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/"
xmlns:xhtml="http://www.w3.org/1999/xhtml">
<atom:title>User profiles</atom:title>
<atom:author>
<atom:name>IBM WebSphere Portal</atom:name>
</atom:author>
<atom:link href="/wps/um/secure/users/profiles?
resultsPerPage=2&searchAttributes=uid
%3DtestUser*&includeAttributes=uid&pageHandle=4d793b59" rel="self"/>
<atom:id>um:secure/users/profiles%3FresultsPerPage%3D2%26searchAttributes
%3Duid%
3DtestUser*%26includeAttributes%3Duid%26pageHandle%3D4d793b59</atom:id>
<atom:updated>2009-02-25T13:13:12.296Z</atom:updated>
<atom:link href="/wps/um/secure/users/profiles?
resultsPerPage=2&searchAttributes=uid
%3DtestUser*&includeAttributes=uid&pageHandle=4d793b59&page=1"
rel="first"/>
<atom:link href="/wps/um/secure/users/profiles?
resultsPerPage=2&searchAttributes=uid
%3DtestUser*&includeAttributes=uid&pageHandle=4d793b59&page=3"
rel="last"/>
<atom:link href="/wps/um/secure/users/profiles?
resultsPerPage=2&searchAttributes=uid
%3DtestUser*&includeAttributes=uid&pageHandle=4d793b59&page=2"
```

```
rel="next"/>
<opensearch:totalresults>5</opensearch:totalresults>
<opensearch:startindex>1</opensearch:startindex>
<opensearch:itemsperpage>2</opensearch:itemsperpage>
<atom:entry>
<atom:title>uid=testUser5,o=defaultWIMFileBasedRealm</atom:title>
<atom:link
href="/wps/um/secure/users/profiles/9eAeK9OCJR07M1D0JMG65BOCJMG6NHP8MM4C43EGJMKC
43D83JHC5RD66R4713" rel="self"/>
<atom:link
href="/wps/um/secure/groupmembership/9eAeK9OCJR07M1D0JMG65BOCJMG6NHP8MM4C43EGJMK
C43D83JHC5RD66R4713" rel="related"/>
<atom:id>um:secure/users/profiles/9eAe
K9OCJR07M1D0JMG65BOCJMG6NHP8MM4C43EGJMKC43D83JHC5RD66R4713</atom:id>
<atom:updated>2009-02-25T13:13:12.296Z</atom:updated>
<atom:content type="application/xml">
<um:profile type="user"
identifier="uid=testUser5,o=defaultWIMFileBasedRealm">
<um:attribute name="uid" type="xs:string" multivalued="false">
<um:attributevalue>testUser5</um:attributevalue>
</um:attribute>
</um:profile>
</atom:content>
</atom:entry>
```

POST, PUT and DELETE cases

For the operations of the PUMA SPI REST Service interface that use one of the methods POST, PUT, or DELETE, you do not need to consider anything special from a client perspective when using the APP format. You can use the plain XML format as defined in the schema; you do not need to wrap it into ATOM elements when doing POSTs or PUTs of resources to the server. The client needs to make sure that the correct namespaces and syntax are used for the input that is posted to the server, as the server validates both the payload format defined in the `um.xsd` and the basic ATOM structures. This is not very strict, but elements that are not defined in the ATOM specification are rejected. The client should therefore check the response as follows:

- Successful POST operations to feeds return a response with status code `201 Created`, the `Location` header set to the URI of the new resource, and a representation of the `atom:entry` element that has been created. This is only the case for operations `/um/users/profiles` and `/um/groups/profiles`
- Successful PUT or DELETE operations return a status code of `200 OK`.

Related information:

http://ietfreport.isoc.org/all-ids/draft-ietf-atompub-protocol-11.txt

Switch for tunneling of HTTP methods:

The portal implementation allows you to simulate PUT and DELETE requests by tunneling, that is by using POST requests instead.

By default tunneling is disabled. If you want to enable it, for example because you want to disable PUT and DELETE requests in general, but allow equivalent operations for REST services provided by the portal, set the property `x-method-override.enabled` to the value `true` in the portal configuration service WP ConfigService Resource Environment Provider in the WebSphere Integrated Solutions Console. If you set the property `x-method-override.enabled` to `true` , then the Config Service considers the `x-method-override` request header, when a request comes in. Whether or not to send this header is a decision of the HTTP

client. Although the property name refers to only one replacement, the switch actually applies to both ways of request tunneling as described in the interface.

HTTP caching:

The remote REST service for PUMA sets the `Cache-Control` HTTP header in the response to `public` for resources that are served without authentication, and to `private` for URIs that require authentication.

The expiration time is defined by the `max-age` directive in the same header and can be configured as a number in seconds within the `wp.user.restservice.maxage` environment entry of the remote REST service for PUMA enterprise application. This is called **UserProfileRESTServlet** in the WebSphere Integrated Solutions Console. It defaults to the value `3600` seconds, that is equivalent to one hour.

Context root and authentication mechanism:

The remote REST service for PUMA is implemented as a servlet that runs as a separate enterprise application on the WebSphere Portal Express server.

The starting weight of the application has to be higher than the starting weight of the portal application (usually `wps.ear`) because it needs the portal run time during startup. This is set by the corresponding install and configuration tasks. By default the REST Service application defines the context root `/wps/um`, where the `/wps` part is equivalent to the general portal context path. The `/um` path element is considered to be fix as it is part of the URI path definitions in the RESTful interface.

Using the Portal context path as a prefix for the context root is a necessary prerequisite with regards to the default authentication method that is configured for the servlet, which reuses the application specific form based authentication mechanism implemented in the portal. This way, a request that needs authentication is redirected to the appropriate portal login page. After successful authentication, the portal login again redirects to the previous URL. As this requires a specific handling of the redirect on the client side, it is recommended that clients make sure that a security context exists already before calling operations that involve protected URLs. This is usually the case when running in a portal session context. Alternatively, you can configure the servlet for different authentication methods by changing the `web.xml` descriptor appropriately, for example, for basic or SSL client certificate authentication.

Lookup facility in the portal:

For both convenience and alignment to other portal REST services, the portal remote REST service for PUMA offers a lookup facility. This is done by plugging a provider into a reusable lookup facility in the portal. This functionality allows you to retrieve particular URLs of the service by specifying an absolute URI as a parameter to the so-called lookup servlet addressed by `/wps/poc`.

The absolute URIs for the remote REST service for PUMA can be constructed from the service URL paths or be taken from the `atom:id` elements of the ATOM feeds or entries. The URIs are completely UTF-8 encoded. When constructing a URI manually, you take the URI path after the element `/um/` without a leading slash, add `um:` as URI prefix, and encode the whole expression by using UTF-8. For example, the URL `/um/secure/users/profiles?searchAttributes=uid%3Dwps*` has the URI `um:secure/users/profiles%3FsearchAttributes%3Duid%3Dwps*`. You can then call the URL `/wps/poc?uri=um:secure/users/profiles%3FsearchAttributes`

%3Duid%3Dwps* and get redirected to the corresponding REST service URL. Or, when you use the verb=lookup parameter, which is mandatory for operations other than GET, get an ATOM feed with the service description such as the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:entry xmlns:atom="http://www.w3.org/2005/Atom" xmlns:xhtml="http://www.w3.org/1999/xhtml">
    <atom:author>
        <atom:name>IBM WebSphere Portal</atom:name>
    </atom:author>
    <atom:id>um:secure/users/profiles?searchAttributes=uid=wps*</atom:id>
    <atom:link href="/wps/um/secure/users/profiles?searchAttributes=uid=wps*"/>
    <atom:title>Remote PUMA REST service URI information</atom:title>
    <atom:updated>2006-12-19T19:21:37.562Z</atom:updated>
</atom:entry>
```

If you need the service description of several REST Service URLs, you can set the uri parameter multiple times by using different URIs, and the lookup servlet will aggregate the descriptions into one ATOM feed.

Portal Access Control interfaces

Portal Access Control provides interfaces for retrieving and modifying and access control information of portal resources, such as portlets or pages.

"Portal Access Control SPI"
The Portal Access Control (PAC) System Programming Interface (SPI) lets you directly retrieve and modify access control information for portal resources.

"Portal Access Control REST API" on page 2923
The Portal Access Control REST API lets you remotely access and modify access control information for resources through the HTTP protocol.

Portal Access Control SPI

The Portal Access Control (PAC) System Programming Interface (SPI) lets you directly retrieve and modify access control information for portal resources.

You can retrieve the following main service interfaces through the AccessControlHome interface.

You can use the **runtime model** of Portal Access Control to evaluate whether a user is allowed to perform a specific operation. For example, you can evaluate whether the user who triggered the request is allowed to do that.

com.ibm.portal.ac.AccessControlHome
Portal Access Control provides interfaces for retrieving and modifying and access control information of portal resources, such as portlets or pages.

com.ibm.portal.ac.AccessControlGlobalRuntimeModel
The AccessControlGlobalRuntimeModel provides read access to the current access control permissions on a resource that is registered at Portal Access Control.

com.ibm.portal.ac.AccessControlRuntimeModel
The AccessControlRuntimeModel provides read access to the current access control permissions on one specific resource.

You can use the **configuration model** to retrieve the hierarchy of protected resources, and also to retrieve and modify role assignments and configuration data such as role blocks.

com.ibm.portal.ac.AccessControlEnvironment
The AccessControlEnvironment provides some general information about the access control configuration, for example the available role types.

com.ibm.portal.ac.ManagedProtectedResource
The ManagedProtectedResource provides read access to the access control configuration of a resource that is registered at Portal Access Control.

com.ibm.portal.ac.ManagedProtectedResourceController
The ManagedProtectedResourceController provides write access to the access control configuration of a resource that is registered at Portal Access Control.

com.ibm.portal.ac.RoleData
The RoleData provides read access to the role data of a single resource, such as role assignments.

Note: For performance reasons, make requests of the form "Is user *x* allowed to perform operation *y* on resource *z* ?" by using AccessControlRuntimeModel or AccessControlGlobalRuntimeModel, rather than by asking for explicit role assignments using the RoleData interface.

com.ibm.portal.ac.RoleDataController
The RoleDataController provides write access to the role data of a single resource, such as role assignments.

com.ibm.portal.ac.ManagedProtectedResourceModel
The ManagedProtectedResource represents the hierarchical tree model of protected resources per database domain.

Examples of how these interfaces are used are provided in the accompanying Javadoc. The following example shows how to evaluate if a principal has view permissions on a resource:

```
Identifiable resource = ... ; // some resource, for example a portlet
Principal bob = ... ; // some principal, for example Bob
Context ctx = new InitialContext();
AccessControlHome home = (AccessControlHome) ctx.lookup(AccessControlHome.JNDI_NAME);
AccessControlEnvironment environment = home.getAccessControlEnvironment();
Permission permission = environment.getPermission(resource, RoleType.USER);
AccessControlGlobalRuntimeModel globalModel = home.getAccessControlGlobalRuntimeModel();
isAllowed = globalModel.hasPermission(bob, permission);
```

Portal Access Control REST API

The Portal Access Control REST API lets you remotely access and modify access control information for resources through the HTTP protocol.

The REST API for Portal Access Control provides the following URIs:

Member Feed
`ac:member:oid:<principalID>@role:<roleTypeName>@oid:<resourceID>`

Member Collection Feed
`ac:member:<roleTypeName>@oid:<resourceID>`

Role Feed
`ac:role:<roleTypeName>@oid:<resourceID>`

Role Collection Feed
`ac:role:oid:<resourceID>`

Resource Config Feed
`ac:resourceconfig:oid:<resourceID>`

Allowed Access Feed
`ac:access:oid:<resourceID>`

Notes:

- Member Collection Feed is defined as a collection.
- Role Collection Feed is defined as a collection.
- Member Collection Feed allows the HTTP method `POST`.
- The available `<roleTypeName>` are listed in the table in the topic Roles.

Supported HTTP methods

Table 454. Supported HTTP methods for feeds

Feed Name	GET method	POST method	PUT method	DELETE method
Member	405	405	405	Removes a member from a role.
Member Collection	Returns all members of a role.	Adds a principal to the specified role.	405	405
Role	Returns a role.	405	405	405
Role Collection	Returns all roles of a resource.	405	405	405
Resource Config	Returns a resource configuration.	405	Modifies the resource configuration.	405
Allowed Access	Returns all access levels that the current user has been granted or has inherited.	405	405	405

Common response elements

The following XML framework is returned for GET requests on each of the feeds. It contains the link to itself, the title of the feed, the ID of the feed, the feed URI, and the timestamp when the feed was created. For collection feeds, the top-level element is an `atom:feed` and `opensearch` elements are included. For the other feeds, the top-level element is an `atom:entry`.

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:feed xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xhtml="http://www.w3.org/1999/xhtml"
  xmlns:atom="http://www.w3.org/2005/Atom"
  xmlns:model="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model-elements"
  xmlns:ac="http://www.ibm.com/xmlns/prod/lotus/access-control/v1.0"
  xmlns:portal="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model"
  xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/"
  xmlns:base="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0/ibm-portal-composite-base"
  xml:base="http://localhost:10040/wps/mycontenthandler/!ut/p/digest!F4eOOrdKv7QXA2o0iUT9A/T9A
                /ac/member:User@oid:ibm.portal.Home">
  <atom:author>
    <atom:name>IBM WebSphere Portal/6.1.0.3</atom:name>
  </atom:author>
  <atom:title>MemberCollection</atom:title>
  <atom:id>ac:member:User@oid:ibm.portal.Home</atom:id>
  <atom:link href="/wps/mycontenthandler/!ut/p/digest!F4eOOrdKv7QXA2o0iUT9A/ac/member:User@oid:ibm.portal.Home"
            rel="self"
            type="application/atom+xml"/>
  <opensearch:startIndex>0</opensearch:startIndex>
  <opensearch:itemsPerPage>2147483647</opensearch:itemsPerPage>
  <opensearch:totalResults>0</opensearch:totalResults>
  <atom:updated>2009-10-01T06:03:51.850Z</atom:updated>
</atom:feed>
```

Common response elements for collection feeds

For collection feeds, the feed contains a list of `atom:entry` elements. Each entry includes the following additional information: the ID of the entry, the entry's URI, additional links, and a content element that contains the actual information.

```
<atom:entry>
  <atom:id>ac:member:oid:8eAe13RO6G4CL3TGMIPDKBQ6MGHE53P02OTDI3T26M14LRSA6PDE@role:User@oid:
          6_CGAH47L00GN960I4GOF1G510I3</atom:id>
  <atom:title>MemberCollection</atom:title>
  <atom:updated>2009-10-01T05:56:58.341Z</atom:updated>
  <atom:link href="/wps/mycontenthandler/!ut/p/digest!F4eOOrdKv7QXA2o0iU-T9A
                  /ac/member:oid:8eAe13RO6G4CL3TGMIPDKBQ6MGHE53P02OTDI3T26M14LRSA6PDE@role
   :User@oid:6_CGAH47L00GN960I4GOF1G510I3" rel="edit" type="application/atom+xml"/>
  <atom:content type="application/xml">
    <ac:member ac:id="8eAe13RO6G4CL3TGMIPDKBQ6MGHE53P02OTDI3T26M14LRSA6PDE"
               ac:DN="all authenticated portal users" ac:type="virtual"/>
  </atom:content>
</atom:entry>
```

URI variables and their ranges

<roleTypeName>
: Case-insensitive name of a defined role type; for example, Manager, EDitor, usEr.

<resourceID>
: `ObjectID` or `uniqueName` of a resource; for example, `ibm.portal.Home` or `8eAe13RO6G4CL3TGMIPDKBQ6MGHE53P02OTDI3T26M14LRSA6PDE`.

<principalID>
: `ObjectID` of a Principal (user or group); for example, `8eAe13RO6G4CL3TGMIPDKBQ6MGHE53P02OTDI3T26M14LRSA6PDE`.

Member feed and the DELETE method

URI: `ac:member:oid:<principalID>@role:<roleTypeName>@oid:<resourceID>`

Parameters: None.

The HTTP `DELETE` method:

- Deletes the given principal from the given `RoleType` on given resource.
- Equivalent Portal Access Control Java API:
  `accessControlHome.getRoleDataController(resourceID)`

  .
- Returns `200` if successful.
- Returns `404` if `resourceID` cannot be resolved.
- Returns `400` if:
  - `roleTypeName` is not applicable.
  - the according role does not exist.
  - `principalID` cannot be resolved.
  - the current user does not have sufficient access rights.

Member Collection feed and the GET method

URI: `ac:member:<roleTypeName>@oid:<resourceID>`

Parameters:

start-index=*n*
Specifies the *n-th* member of the role will be the first one in the result list.

max-results=*m*
Specifies that at most *m* members will be included.

The HTTP `GET` method:

- Provides all members mapped to a single role for a given resource.
- Equivalent Portal Access Control Java API: accessControlHome.getRoleData(resourceID).getMappedPrincipals(roleType)
- Returns `200` if successful.
- Returns `404` if `resourceID` cannot be resolved.
- Returns `400` if `roleTypeName` is not applicable.

The following is the result of the `GET` method. Each entry element in the member collection contains a single member element that specifies the member identity. The edit link can be used to remove the member from the role (DELETE on member feed).

```
<atom:entry ...>
...
<atom:title>MemberCollection</atom:title>
<atom:entry>
 <atom:id>ac:member:oid:8eAe13RO6G4CL3TGMIPDKBQ6MGHE53P02OTDI3T26M14LRSA6PDE@role:User@oid:
          6_CGAH47L00GN960I4GOF1G510I3</atom:id>
 <atom:title>MemberCollection</atom:title>
 <atom:updated>2009-10-01T05:56:58.341Z</atom:updated>
 <atom:link href="/wps/mycontenthandler/!ut/p/digest!F4eOOrdKv7QXA2o0iUT9A
                  /ac/member:oid:8eAe13RO6G4CL3TGMIPDKBQ6MGHE53P02OTDI3T26M14LRSA6PDE@role:
    Privileged%25252520User@oid:6_CGAH47L00GN960I4GOF1G510I3"
    rel="edit" type="application/atom+xml"/>
 <atom:content type="application/xml">
  <ac:member ac:id="8eAe13RO6G4CL3TGMIPDKBQ6MGHE53P02OTDI3T26M14LRSA6PDE"
             ac:DN="all authenticated portal users" ac:type="virtual"/>
 </atom:content>
</atom:entry>
```

Member Collection feed and the POST method

URI: `ac:member: <roleTypeName>@oid:<resourceID>`

Parameters: None.

The HTTP `POST` method:

- Adds the principal specified in the payload to the given `RoleType` on given resource.
- Equivalent Portal Access Control Java API:
  `accessControlHome.getRoleDataController(resourceID).addPrincipalsToRole(RoleType, Collection<Principal>)`
- Returns `201` if successful.
- Returns `404` if:
  - `resourceID` cannot be resolved.
  - the specified principal cannot be resolved from the distinguished name (DN) or email.
- Returns `400` if:
  - `roleTypeName` is not applicable.
  - specified principal cannot be resolved from `oid`.
  - the current user does not have sufficient access rights.

The `POST` payload for the Member Collection feed is as follows:

UID: ac:member: <roleTypeName>@oid:<resourceID>

The payload needs to include at least the following:

```
<atom:entry xmlns:atom="http://www.w3.org/2005/Atom">
  <atom:content type="application/xml">
    <ac:member xmlns:ac="http://www.ibm.com/xmlns/prod/lotus/access-control/v1.0"
                    ac:id="9eAe6BD86PGCHPD6JMCCG1D8MMG63JD0JM4C3BDAJMK66BC46O"/>
  </atom:content>
</atom:entry>
```

As an alternative to ac:id, which is used to identify a Principal using the ObjectID, you can use the following alternative identifiers:

ac:DN="uid=wpsadmin,o=defaultWIMFileBasedRealm" ac:type="user"
: Specifies the principal using the distinguished name. The type can be user, group or virtual (for virtual principals), where user is the default. Valid Virtual Principals are the ones defined in the following Javadoc: com.ibm.portal.um.PumaEnvironment.VirtualPrincipalNames and are currently **all authenticated portal users**, **all portal user groups**, and **anonymous portal user**.

ac:email="wpsadmin@de.ibm.com"
: Specifies users by their email addresses.

Role feed and the GET method

URI: ac:role:<roleTypeName>@oid:<resourceID>

Parameters:

resolve-membership
: Specify true or false. If true, then members are included. The default is false.

The HTTP GET method:

- Returns information about a single role available for given resource.
- Role is only available if at least one principal is mapped to it, otherwise returns 404.
- Equivalent Portal Access Control Java API: accessControlHome.getRoleData(resourceID).getRole(RoleType).
- Returns 200 if successful.
- Returns 404 if:
  - resourceID cannot be resolved.
  - the role for specified roleTypeName is not available.
- Returns 400 if roleTypeName is not applicable.

The following is the result of the GET method:

```
<atom:entry ...>
  ...
  <atom:title>Role</atom:title>
  <atom:id>ac:role:User@oid:myPage</atom:id>
  <atom:link href="/wps/mypoc/!ut/p/digest!TSVlGy5DI0S5vyp5i_yTw
                  /ac/role:User@oid:myPage?mode=download&amp;resolve-membership=true"
    rel="self" type="application/atom+xml"/>
  <atom:updated>2009-10-01T12:35:05.442Z</atom:updated>
  <atom:link ac:rel="members" href="/wps/mycontenthandler/!ut/p/digest!TSVlGy5DI0S5vyp5i_yTw
           /ac/member:User@oid:6_M8768B1A00FT20I480O8A53000?
   resolve-membership=true" rel="related" type="application/atom+xml"/>
 <atom:content type="application/xml">
  <ac:role ac:type="User">
   <ac:member
     ac:id="8eAeI9EE3J5C63C8JM064RD2JMG613C2MM4C6BP4MM072JD6MI17P9E03R86H1"ac:display-name="wpsadmins"
```

```
    ac:DN="cn=wpsadmins,o=defaultWIMFileBasedRealm" ac:type="group"/>
  </ac:role>
 </atom:content>
</atom:entry>
```

Role Collection feed and the GET method

URI: `ac:role:oid:<resourceID>`

Parameters:

`filter` Specify one of the following values:

`type=<roleTypeName>` to return the specified role, if available.

`inUse` to return available roles. This is the default value.

`all` to return all applicable role types.

`start-index=`*n*

Specifies that the *n-th* member of the role is the first one in the result list.

`max-results=`*m*

Specifies that at most *m* members are included.

URI: `ac:role:oid:<resourceID>`

The HTTP `GET` method:

- Provides information about all roles available for a given resource.
- A role is only available if at least one principal is mapped to it.
- Equivalent Portal Access Control Java API:
  `accessControlHome.getRoleData(resourceID).getRoles()`
- Returns `200` if successful.
- Returns `404` if `resourceID` cannot be resolved.
- Returns `400` if the specified `roleTypeName` is not applicable.

The following is the result of the `GET` method:

```
<atom:feed ...>
  ...
  <atom:title>RoleCollection</atom:title>
  <atom:id>ac:role:oid:myPage</atom:id>
  <atom:link href="/wps/mypoc/!ut/p/digest!TSVlGy5DI0S5vyp5i-yTw/ac/role:oid:myPage?mode=download"
        rel="self" type="application/atom+xml"/>
  <opensearch:startIndex>0</opensearch:startIndex>
  <opensearch:itemsPerPage>2147483647</opensearch:itemsPerPage>
  <opensearch:totalResults>1</opensearch:totalResults>
  <atom:updated>2009-10-01T12:46:16.509Z</atom:updated>
  <atom:entry>
  <atom:id>ac:role:User@oid:6_M8768B1A00FT20I480O8A53000</atom:id>
  <atom:title>RoleCollection</atom:title>
  <atom:updated>2009-10-01T12:46:16.509Z</atom:updated>
  <atom:link ac:rel="members" href="/wps/mycontenthandler/!ut/p/digest!TSVlGy5DI0S5vyp5i-yTw
            /ac/member:User@oid:6_M8768B1A00FT20I480O8A53000"
    rel="related" type="application/atom+xml"/>
  <atom:link href="/wps/mypoc/!ut/p/digest!TSVlGy5DI0S5vyp5i-yTw
                /ac/role:User@oid:6_M8768B1A00FT20I480O8A53000?mode=download"
    rel="self" type="application/atom+xml"/>
  <atom:content type="application/xml">
   <ac:role ac:type="User"/>
  </atom:content>
 </atom:entry>
</atom:feed>
```

Resource Config feed and the GET method

UID: `ac:resourceconfig:oid:<resourceID>`

Parameters: None.

The HTTP GET method:

- Provides configuration information on the following resources:
  - Owner
  - Role blocks
  - IsPrivate
- Equivalent Portal Access Control Java API:

  accessControlHome.getManagedProtectedResource(resourceID)
- Returns 200 if successful.
- Returns 404 if resourceID cannot be resolved.

The following is the result of the GET method:

```
<atom:entry ...>
 ...
 <atom:title>ResourceConfig</atom:title>
 <atom:id>ac:resourceconfig:oid:myPage</atom:id>
 <atom:link href="/wps/mycontenthandler/!ut/p/digest!TSVlGy5DI0S5vyp5i_yTw
                  /ac/resourceconfig:oid:myPage" rel="self" type="application/atom+xml"/>
 <atom:updated>2009-10-01T12:15:59.683Z</atom:updated>
 <atom:content type="application/xml">
  <ac:resource-config>
   <ac:ownerac:id="9eAePPC2JP8C2ROIJMG6M1D8JMG6PPD0JM0723P8JM06KPOC6IH66BOCMQC6N1"
      ac:display-name="wpsadmin"
      ac:DN="uid=wpsadmin,o=defaultWIMFileBasedRealm" ac:type="user"/>
   <ac:role-block ac:block-type="inheritance" ac:type="Privileged User"/>
   <ac:role-block ac:block-type="propagation" ac:type="Delegator"/>
  </ac:resource-config>
 </atom:content>
</atom:entry>
```

Resource Config feed and the PUT method

UID: ac:resourceconfig:oid:<resourceID>

Parameters:

mode Specify one of the following values:

merge to add role blocks to existing ones. The owner is changed if you specify this value.

update to replace existing role blocks and owners. The role blocks and owners are removed if you do not specify them. This is the default value.

The HTTP PUT method:

- Updates the following resource configurations:
  - Owner
  - Role blocks
- Equivalent Portal Access Control Java API:

  accessControlHome.getManagedProtectedResource(resourceID)
- Returns 200 if successful.
- Returns 404 if resourceID cannot be resolved.
- Returns 400 if the current user does not have sufficient access rights.

The PUT payload for the Resource Config feed is as follows:

URI: ac:resourceconfig:oid:<resourceID>

The payload must be as follows:

```
<atom:entry xmlns:atom="http://www.w3.org/2005/Atom"
      xmlns:ac="http://www.ibm.com/xmlns/prod/lotus/access-control/v1.0" >
  <atom:content type="application/xml">
    <ac:resource-config>
      <ac:owner ac:id="9eAe6BD86PGCHPD6JMCCG1D8MMG63JD0JM4C3BDAJMK6"/>
      <ac:role-block ac:block-type="inheritance" ac:type="MANAger"/>
      <ac:role-block ac:block-type="propagation" ac:type="Editor"/>
      <ac:role-block ac:block-type="propagation" ac:type="user"/>
    </ac:resource-config>
  </atom:content>
</atom:entry>
```

The owner and role-block elements are optional.

As an alternative to `ac:id`, which is used to identify the owner using the `ObjectID`, use the following alternative identifiers:

ac:DN="uid=wpsadmin,o=defaultWIMFileBasedRealm" ac:type="user"
: Specifies the principal using the distinguished name. The type can be user, group or virtual (for virtual principals), where user is the default.

ac:email="wpsadmin@de.ibm.com"
: Specifies users by their email addresses.

Allowed Access feed and the GET method

UID: `ac:access:oid:<resourceID>`

Parameters: None.

The HTTP `GET` method:

- Provides run time information on the resource for the current user as follows: All access levels that the current user has on the given resource.
- Equivalent Portal Access Control Java API: There is no 1:1 mapping to the Java API, however, the following API is called for all applicable `roleTypes`:

  ```
  accessControlHome.getAccessControlRuntimeModel(resourceID).hasPermission(roleType)
  ```
- Returns `200` if successful.
- Returns `404` if `resourceID` cannot be resolved.

The following is the result of the `GET` method:

```
<atom:entry...>
 ...
 <atom:title>allowed-access</atom:title>
 <atom:id>ac:access:oid:ibm.portal.Home</atom:id>
 <atom:link href="/wps/mycontenthandler/!ut/p/digest!TSVlGy5DI0S5vyp5i-yTw
                 /ac/access:oid:ibm.portal.Home" rel="self" type="application/atom+xml"/>
 <atom:updated>2009-10-01T12:55:16.620Z</atom:updated>
 <atom:content type="application/xml">
  <ac:allowed-access xmlns:ac="http://www.ibm.com/xmlns/prod/lotus/access-control/v1.0"
                     ac:user-owned="false">
   <ac:access-level ac:type="Administrator"/>
   <ac:access-level ac:type="Security Administrator"/>
   <ac:access-level ac:type="Delegator"/>
   <ac:access-level ac:type="Manager"/>
   <ac:access-level ac:type="Editor"/>
   <ac:access-level ac:type="Contributor"/>
   <ac:access-level ac:type="Privileged User"/>
   <ac:access-level ac:type="User"/>
  </ac:allowed-access>
 </atom:content>
</atom:entry>
```

Developing portlets

Get an overview of the process of creating portlets, learn about the concepts of the APIs used to develop portlets, and view the samples to get you started. Also, learn about integrating features such as single sign-on, cooperative sharing of information using the property broker, and migrating Struts applications to the portlet environment.

WebSphere Portal Express supports portlets written to the standard portlet API.

- If you are not an experienced Java developer...

  you can quickly integrate many of your existing resources into WebSphere Portal Express without developing your own portlets.
- If you have Java and servlet development experience but are new to portlet development...

  you should start by reading "Portlet concepts" on page 2932 and "Standard portlet API" on page 2934. When you are ready to start developing portlets, see Portlet creation basics for samples that demonstrate common features of the API.
- If you have experience with Struts applications...

  "Struts Portlet Framework" on page 2943 explains how to take an existing Struts application and convert it to a portlet.
- If you are already familiar with portlet development using either the standard portlet API...

  "Portlet communication" on page 3069 describes how you can enable portlets to share information with other portlets on the page, using the property broker service. "Collaborative Services API and the person tag" on page 3103 explains how you can integrate collaborative functionality into your portlets, such as online status, chat, and email.

Rational Application Developer includes tools designed to help you develop portlet applications for WebSphere Portal Express.

"Portlet concepts" on page 2932
Learn about portlets from a user's and an application developer's perspective. View a brief comparison between a portlet and a servlet and understand basic portlet concepts; know the effect of Java 2 security enablement on the operation of portlets that rely on certain privileges for processing.

"Standard portlet API" on page 2934
The Java Portlet Specification addresses the requirements of aggregation, personalization, presentation, and security for portlets running in a portal environment.

"Portlet services" on page 2937
Portlet services are used to provide common functionality to portlets. Each portlet service has its own service-specific interface for the functionality that it offers.

"Struts Portlet Framework" on page 2943
Learn how to write Struts applications that can be deployed in WebSphere Portal Express. Know special considerations when developing such applications and additional concepts, such as portlet modes, multiple device support, and portlet communication, that might need to be addressed by the Struts application.

"Web 2.0 user interface features" on page 2979
Learn about portal features that pertain to the Web 2.0 generation type of Web user interface.

Portlet concepts

Learn about portlets from a user's and an application developer's perspective. View a brief comparison between a portlet and a servlet and understand basic portlet concepts; know the effect of Java 2 security enablement on the operation of portlets that rely on certain privileges for processing.

Portlets are reusable Web modules that run on a portal server and provide access to Web-based content, applications, and other resources. Companies can create their own portlets or select portlets from a catalog of third-party portlets. Portlets are intended to be assembled into a larger portal page, with multiple instances of the same portlet displaying different data for each user.

From a user's perspective, a portlet is a window on a portal site that provides a specific service or information, for example, a calendar or news feed. From an application development perspective, portlets are pluggable Web modules that are designed to run inside a portlet container of a portal server.

The portlet container provides a run time environment in which portlets are instantiated, used, and finally destroyed. Portlets rely on the portal infrastructure to access user profile information, participate in window and action events, communicate with other portlets, access remote content, lookup credentials, and to store persistent data. The Portlet API provides standard interfaces for these functions. The portlet container is not a stand-alone container like the servlet container. Instead, it is implemented with the servlet container and reuses the functionality provided by the servlet container.

- The Java Portlet Specification API. This is based on javax.portlet interfaces. WebSphere Portal Express supports the Java Portlet Specifications 1.0 and 2.0, also known as JSR168 and JSR286.

You can place both types of portlets on portal pages. However, a portlet cannot mix classes and methods from both packages.

Each portlet on the page is responsible for providing its output in the form of markup fragments to be integrated into the portal page. The portal is responsible for providing the markup surrounding each portlet. In HTML, for example, the portal can provide markup that gives each portlet a title bar with minimize, maximize, help, and edit icons.

Portlets and the Servlet API

Portlets are special types of Web modules designed to run in the context of a portal. They are written to comply with a portlet API that is similar to the servlet API but addresses portal specific areas of concerns. In contrast to servlets, portlets may not send errors directly to browsers, forward requests, or write arbitrary markup to the output stream. Another difference compared to servlets is that portlets rely on specific features of the portal infrastructure, such as user profile information, storing and retrieving persistent settings, and getting client information.

Generally, portlets are administered more dynamically than servlets. Portlet applications consisting of several portlets can be installed and removed using the portal administration interface while the portal server is running. In a similar manner, the settings of a portlet can be changed by an administrator with appropriate access rights at any time without stopping/restarting the portal server Web application. Portlets can be created and deleted dynamically. For example, a clipping administration portlet can create new portlet instances whenever an administrator creates a new clipping.

Java 2 security

Enablement of Java 2 security on the portal server can affect the operation of portlets that rely on certain privileges for processing. If your portlet requires certain privileges, for example, access to the file system or to the network, you might need to package a `was.policy` file in the portlet WAR indicating which privileges are needed. Even more important, any privileges needed by the portlet should be documented for the administrator.

Related concepts:

"Java 2 security with WebSphere Portal Express" on page 1572
Java 2 (J2SE) security provides a policy-based, fine-grain access control mechanism that increases overall system integrity by checking for permissions before allowing access to certain protected system resources. J2SE security allows you to set up individual policy files that control the privileges assigned to individual code sources. If the code does not have the required permissions and still tries to execute a protected operation, the Java Access Controller will throw a corresponding security exception.

Related reference:

"Standard portlet API" on page 2934
The Java Portlet Specification addresses the requirements of aggregation, personalization, presentation, and security for portlets running in a portal environment.

Standard portlet API

The Java Portlet Specification addresses the requirements of aggregation, personalization, presentation, and security for portlets running in a portal environment.

Version 1.0 of the Java Portlet Specification was approved by the Java Community Process in October 2003 as JSR 168. The purpose of the specification is to solve the problem of portlet compatibility between portal servers offered by different vendors. Version 2.0 of the Java Portlet Specification extended the capabilities to include coordination between portlets, resource serving, and other advanced features. The final version 2.0 was approved by the Java Community Process in March 2008 as JSR 286.

WebSphere Portal Express includes a portlet run time environment that supports portlets developed according to the Java Portlet Specification, hereafter called *standard portlets*.

With WebSphere Portal Express there are some IBM specific parameters for portlets available. For more information about these parameters refer to the topic *Deployment descriptors* under the section *Reserved parameter names*.

For portlets conforming to JSR 286, WebSphere Portal Express also includes support for two-phase rendering.

The following topics provide specific information about the implementation of the standard portlet API within WebSphere Portal Express.

"Using two-phase rendering with JSR 286 portlets" on page 2935
For portlets conforming to JSR 286, IBM WebSphere Portal Express includes support for two-phase rendering, which allows portlets to set cookies and the HTTP headers and to change the portal page title dynamically.

Related concepts:

"Portlet concepts" on page 2932
Learn about portlets from a user's and an application developer's perspective. View a brief comparison between a portlet and a servlet and understand basic portlet concepts; know the effect of Java 2 security enablement on the operation of portlets that rely on certain privileges for processing.

"IBM Portlet API" on page 3110
The IBM Portlet API is no longer supported starting with WebSphere Portal Express Version 8.5.0. You must convert portlets based on the IBM Portlet API to the Standard Portlet API. Learn how to convert IBM Portlets to Standard API portlets.

"Deployment descriptors" on page 3129
The deployment descriptors define configuration information for the portlets that a portlet application contains.

Related reference:

"JSP tags for standard portlets" on page 3133
The standard portlet API defines several tags that can be used in portlet JSPs to access the portlet request and response and to generate URLs.

Related information:

"Converting IBM portlets (IBM i Linux Windows)" on page 3110
You can convert your basic IBM portlets and IBM portlets that use the Struts Portlet Framework to the standard portlet API.

Java Portlet Specification JSR 286 (http://jcp.org/en/jsr/detail?id=286)

Java Portlet Specification JSR 168(http://jcp.org/en/jsr/detail?id=168)

Best practices: Developing portlets using JSR 168 and WebSphere Portal

Introducing the Portlet Specification, Part 1

Introducing the Portlet Specification, Part 2

Using two-phase rendering with JSR 286 portlets

For portlets conforming to JSR 286, IBM WebSphere Portal Express includes support for two-phase rendering, which allows portlets to set cookies and the HTTP headers and to change the portal page title dynamically.

"Enabling two-phase rendering for a portlet"
By default, two-phase rendering is turned off. To enable two-phase rendering for a portlet, you must update the `portlet.xml` deployment descriptor for the portlet.

"Setting headers for a JSR 286 portlet" on page 2936
To set HTTP header information in your JSR 286 portlet, use the setProperty and addProperty methods of the PortletResponse.

"Setting cookies for a JSR 286 portlet" on page 2936
Although cookies can be set like any other HTTP header, the portlet API provides the addProperty convenience method on the PortletResponse for setting cookies.

"Modifying the HTML head section of a JSR 286 portlet" on page 2936
To write into the HTML head section of your JSR 286 portlet, for example, to change a page title, use the `addProperty` method on the `PortletResponse`.

"Setting portlet caching values for a JSR 286 portlet" on page 2937
You can dynamically modify portlet caching parameters for a JSR 286 portlet during the render phase.

Related reference:

"WSRP two-phase rendering" on page 1501
The WSRP Consumer and the WSRP Producer in the portal support two-phase rendering for JSR 286 portlets. Two-phase rendering allows a remote portlet to set headers and cookies and to modify the HTML head section.

Enabling two-phase rendering for a portlet:

By default, two-phase rendering is turned off. To enable two-phase rendering for a portlet, you must update the `portlet.xml` deployment descriptor for the portlet.

Procedure

Edit the `portlet.xml` file for the portlet. Add the following entry to the file:

```
<portlet>
    ...
    <container-runtime-option>
        <name>javax.portlet.renderHeaders</name>
        <value>true</value>
    </container-runtime-option>
</portlet>
```

Setting headers for a JSR 286 portlet:

To set HTTP header information in your JSR 286 portlet, use the setProperty and addProperty methods of the PortletResponse.

Procedure

Specify the appropriate method, depending on whether you want to overwrite or append key values.

- `setProperty(String key, String value)`: Overwrites all previous values for the given key.
- `addProperty(String key, String value)`: Attempts to append the key value.

When setting headers in the render lifecycle phase, portlets should set the header in the render headers part or simply override the GenericPortlet.doHeaders method to make sure the server's response headers have not already been committed. Note, however, that the delivery of HTTP headers to the client cannot be guaranteed, because other portlets on a page might override it or the setting of some header attributes might be against the portal's policy.

Setting cookies for a JSR 286 portlet:

Although cookies can be set like any other HTTP header, the portlet API provides the addProperty convenience method on the PortletResponse for setting cookies.

Procedure

Invoke the addProperty method to set cookies.

- PortletReponse.addProperty(javax.servlet.http.Cookie cookie)

Note: When setting cookies, you must invoke the addProperty method before the response headers are committed. This should occur no later than during the render headers sub phase of the render lifecycle phase.
Example:

```
protected void doHeaders(RenderRequest request, RenderResponse response)
{
    ...
    Cookie c = new Cookie("myCookieName", "myCookieValue");
    c.setPath(request.getContextPath());
    response.addProperty(c);
    ...
}
```

Modifying the HTML head section of a JSR 286 portlet:

To write into the HTML head section of your JSR 286 portlet, for example, to change a page title, use the `addProperty` method on the `PortletResponse`.

Procedure

Invoke the `addProperty` method to modify the HTML head section.

- `PortletReponse.addProperty(String key, org.w3c.dom.Element element)`

Note: When modifying the HTML head section, you must invoke the `addProperty` method before the response headers are committed. This should occur no later than during the render headers sub phase of the render lifecycle phase.

Example

Example:

```
protected void doHeaders(RenderRequest request, RenderResponse response)
{
 Element title = response.createElement("title");
        title.setTextContent("My Portal Page Title");
        response.addProperty(MimeResponse.MARKUP_HEAD_ELEMENT, title);
}
```

What to do next

If you add a script tag to the portlet head section, be sure to add text to the tag. If you do not add text, the script tag is not closed properly. Consider the following incorrect code sample from a `doHeader` :

```
String url = "/sample.js";
Element scriptElement = response.createElement(Tag.SCRIPT.toString());
scriptElement.setAttribute(Attribute.TYPE.toString(), "text/javascript");
scriptElement.setAttribute(Attribute.SRC.toString(), url);
response.addProperty(MimeResponse.MARKUP_HEAD_ELEMENT, scriptElement);
```

This generates a script tag in the header like as follows:

```
<script src="/sample.js" type="text/javascript" />
```

This causes rendering problems in Mozilla FireFox and other browsers.

The code needs to call `setTextContent` with a non-empty string as given in the fifth line of the following code sample:

```
String url = "/sample.js";
Element scriptElement = response.createElement(Tag.SCRIPT.toString());
scriptElement.setAttribute(Attribute.TYPE.toString(), "text/javascript");
scriptElement.setAttribute(Attribute.SRC.toString(), url);
scriptElement.setTextContent(" ");
response.addProperty(MimeResponse.MARKUP_HEAD_ELEMENT, scriptElement);
```

This generates a properly closed script tag in the header as follows:

```
<script src="/sample.js" type="text/javascript"> </script>
```

Setting portlet caching values for a JSR 286 portlet:

You can dynamically modify portlet caching parameters for a JSR 286 portlet during the render phase.

Procedure

Update the doHeaders call in the javax.portlet.CacheControl object to set the portlet caching parameters.
The following sample code sets the portlet's cache scope to private and the cache expiration time to a value of 30 seconds.

```
protected void doHeaders(RenderRequest request, RenderResponse response)
{
  response.getCacheControl().setExpirationTime(30);
  response.getCacheControl().setPublicScope(false);
}
```

Portlet services

Portlet services are used to provide common functionality to portlets. Each portlet service has its own service-specific interface for the functionality that it offers.

WebSphere Portal Express supports portlet services for both IBM portlets and standard portlets:

- Standard portlets use a JNDI lookup to retrieve a `PortletServiceHome` object, which is used to retrieve a portlet service implementation.
- IBM portlets retrieve portlet services using the PortletContext.getService() method.

A portlet service can be invoked only from within a portlet.

Portlet service interfaces used by standard portlets are different from those used by IBM portlets. You can write your own portlet service and register it in the portal, so that all portlets can use it. Various services may be implemented by different vendors, for example, a SearchService, LocationService, or a MailService. The following services are available with WebSphere Portal Express:

- ContentAccessService
- CredentialVaultService
- Model SPI services
- PumaHome
- DynamicUIManagementFactoryService

"Accessing portlet services"
Using an example, learn how a standard portlet can retrieve and use a sample portlet service. Accessing a portlet service requires a JNDI lookup for a PortletServiceHome. To use the portlet service, you retrieve a service object from the home, cast it to the service-specific interface and invoke service methods.

"Creating your own portlet service" on page 2939
Write a portlet service by defining the interface, writing the service implementation, making the service accessible, and registering the service.

Accessing portlet services

Using an example, learn how a standard portlet can retrieve and use a sample portlet service. Accessing a portlet service requires a JNDI lookup for a PortletServiceHome. To use the portlet service, you retrieve a service object from the home, cast it to the service-specific interface and invoke service methods.

This section describes how a portlet can invoke a portlet service. See "Portlet services" on page 2937 for a general overview of portlet services. The following example shows how a standard portlet can retrieve and use a sample portlet service. The service implementation and deployment is explained in "Creating your own portlet service" on page 2939.

Accessing a portlet service requires a JNDI lookup for a PortletServiceHome. As this is may be a rather expensive operation, you should do it in the init() method of the portlet and store the returned object in an instance variable:

```
import javax.portlet.*;
import com.ibm.portal.portlet.service.*;

private PortletServiceHome helloServiceHome = null;
...

public void init(PortletConfig config)
{
   javax.naming.Context ctx = new javax.naming.InitialContext();
   try {
       Object home =
             ctx.lookup("portletservice/sample.portletservice.HelloService");
```

```
        if (home != null)
                helloServiceHome = (PortletServiceHome) home;
    } catch(javax.naming.NameNotFoundException ex) {
    // we can do without the service, if it is not available
    config.getPortletContext().log("No hello service available");
    }
}
```

The example is written to gracefully handle a situation where the service is not available because the service is optional for the functionality of the portlet. If possible, you should write your portlets so that they are portable across portal installations, whether they support a given portlet service or not. Such portlets can still run on portals that have not support for portlet services at all and do not even provide the IBM-specifc PortletServiceHome class, because this class is only loaded when the service is actually registered in JNDI.

To use the portlet service, you retrieve a service object from the home, cast it to the service-specific interface and invoke service methods:

```
public void doView(RenderRequest request, RenderResponse response)
{
   if (helloServiceHome != null)
   {
      HelloService service =
            (HelloService) helloServiceHome.getPortletService(HelloService.class);
      service.sayHello(request, response);
   }

   //... do other stuff
}
```

Note that, while it is good practice to store the PortletServiceHome object in an instance variable, you must not store the actual service object, because references to service objects may not be held for longer than a single request.

Related information

- "Portlet services" on page 2937
- "Creating your own portlet service"

Creating your own portlet service

Write a portlet service by defining the interface, writing the service implementation, making the service accessible, and registering the service.

Writing a portlet service consists of four steps:

The service provider interfaces can be used to write portlet services for Standard portlets. The IBM Portlet API is no longer supported starting with WebSphere Portal Express Version 8.5.0. You must convert portlets based on the IBM Portlet API to the Standard Portlet API. Learn how to convert IBM Portlets to Standard API portlets.

Defining the interface

This step is not required if you want to implement your service against an existing interface. Defining a portlet service interface requires the same careful considerations as defining any public API interface. A portlet service interface must extend the `PortletService` interface defined in the `com.ibm.portal.portlet.service` package. The following is an example interface for the `HelloWorldService` .

```
package sample.portletservice;

import java.io.IOException;
import javax.portlet.RenderRequest;
import javax.portlet.RenderResponse;
import com.ibm.portal.portlet.service.PortletService;

public interface HelloService extends PortletService
{
    /** print a nice greeting */
    public void sayHello(RenderRequest request, RenderResponse response)
           throws IOException;
}
```

Figure 38. Extending the PortletService interface

Writing the service implementation

The service implementation must implement the `PortletServiceProvider` interface of the `com.ibm.portal.portlet.service.spi` package to be able to make use of the portlet service life cycle methods in addition to your service interface. The `PortletServiceConfig` parameter of the init() method allows you, for example, to access the configuration of the service (see "Registering the service" on page 2942 for more information).

```
package sample.portletservice;

import java.io.IOException;
import java.io.PrintWriter;
import java.util.prefs.Preferences;

import javax.portlet.RenderRequest;
import javax.portlet.RenderResponse;

import com.ibm.portal.portlet.service.spi.PortletServiceProvider;

public class HelloServiceImpl
    implements HelloService, PortletServiceProvider {

    private String message;

    // called by the portal when the service is initialized
    public void init(Preferences servicePreferences) {

        // read the message from the configuration, default is "Hello"
        message = servicePreferences.get("message", "Hello");
    }

    public void sayHello(RenderRequest request, RenderResponse response)
                throws IOException {
        String user = request.getRemoteUser();
        if (user == null)
            // no user logged in
            user = "Stranger";

        PrintWriter out = response.getWriter();
        out.print(message);
        out.print(", ");
        out.print(user);
    }

}
```

Figure 39. Implementing the PortletServiceProvider interface

Making the service accessible for IBM portlets

This step is optional. If you want your portlet service to be available for IBM portlets, you need to create an additional service interface that extends `org.apache.jetspeed.portlet.service.PortletService` and provides the same functionality.

```
package sample.portletservice;

import java.io.IOException;
import org.apache.jetspeed.portlet.PortletRequest;
import org.apache.jetspeed.portlet.PortletResponse;
import org.apache.jetspeed.portlet.service.PortletService;

public interface HelloServiceIBM extends PortletService {

    /** print a nice greeting */
    public void sayHello(PortletRequest request, PortletResponse response) throws IOException;

}
```

Figure 40. Extending the PortletService interface for IBM portlets

You can have a single implementation that is registered for both interfaces and implements both. If the service methods take arguments that are classes or interfaces from the portlet API, the method signatures are different for the two service interfaces. You can still use a common implementation for both interfaces by using the `APIConverterFactory` class of the `com.ibm.portal.portlet.apiconvert` package. This class includes methods that wrap objects from the IBM portlet API, such as `PortletRequest` and `PortletSession`, and implement the corresponding standard portlet API objects on the service side, so that you can re-use your service implementation for standard portlets.

```
package sample.portletservice;

import java.io.IOException;
import java.io.PrintWriter;
import java.util.prefs.Preferences;

import javax.portlet.RenderRequest;
import javax.portlet.RenderResponse;

import org.apache.jetspeed.portlet.PortletRequest;
import org.apache.jetspeed.portlet.PortletResponse;

import com.ibm.portal.portlet.apiconvert.APIConverterFactory;
import com.ibm.portal.portlet.service.spi.PortletServiceProvider;

public class HelloServiceImpl2
    implements HelloService, HelloServiceIBM, PortletServiceProvider {

    private String message;

    // called by the portal when the service is initialized
    public void init(Preferences servicePreferences) {

        // read the message from the configuration, default is "Hello"
        message = servicePreferences.get("message", "Hello");
    }

    public void sayHello(RenderRequest request, RenderResponse response)
                throws IOException {
        String user = request.getRemoteUser();
        if (user == null)
            // no user logged in
            user = "Stranger";

        PrintWriter out = response.getWriter();
        out.print(message);
        out.print(", ");
        out.print(user);
    }

    public void sayHello(PortletRequest request, PortletResponse response)
                throws IOException {
        sayHello (APIConverterFactory.getInstance().getRenderRequest(request),
            APIConverterFactory.getInstance().getRenderResponse(response));
    }

}
```

Figure 41. Using the APIConverterFactory class

Registering the service

1. Put all service interface and implementation classes into a JAR file.
2. Place the JAR file in the *wp_profile_root*/PortalServer/config directory.
3. Register the new portlet service with the WP PortletServiceRegistryService resource environment provider in the WebSphere Integrated Solutions Console.
   - Create an entry to register the implementation in the JNDI directory. The name for this entry is jndi.*service_interface* and the value is *service_implementation*. The fully qualified service interface name can then be used to lookup the service.
   - Optional: Provide configuration parameters for the implementation. The name for this entry is *service_implementation*.parameter and the value is the required parameter value.
4. Restart WebSphere Portal Express to activate the new settings.

In the following example, `HelloService` is the name of the portlet service, `HelloServiceIBM` is the name of the interface for IBM portlets, and the `message` configuration parameter is set with the value `Greetings`.

Note: The colon (:) used in previous versions of WebSphere Portal Express to designate JNDI entries with `jndi:` is not supported for resource environment providers. Use `jndi.` instead.
To register this portlet service, add the following property names and values to the `PortletServiceRegistryService`:

Property name	Value
`jndi.sample.portletservice.HelloService`	`sample.portletservice.HelloServiceImpl2`
`sample.portletservice.HelloServiceIBM`	`sample.portletservice.HelloServiceImpl2`
`sample.portletservice.HelloServiceImpl2.message`	`Greetings`

Tip: To check whether your service has been registered successfully, use the application server **`dumpNamespace`** tool. The following command, run from the *AppServer_root*`/bin` directory, lists all portlet service entries in JNDI:

```
dumpNamespace -port bootstrap_port -root server -startAt portletservice
```

Using the WebSphere Integrated Solutions Console, you can find the bootstrap port of your portal server in the "End Points" section of the settings for the server "WebSphere_Portal".

Struts Portlet Framework

Learn how to write Struts applications that can be deployed in WebSphere Portal Express. Know special considerations when developing such applications and additional concepts, such as portlet modes, multiple device support, and portlet communication, that might need to be addressed by the Struts application.

Struts is a popular open source project for implementing Web applications using a Model-View-Controller design pattern. Struts has an active development community and continues to evolve. The user community is equally active and many Web applications have been implemented using Struts. The Struts framework can be used to effectively design the Web application. The framework supports both large or small development teams as well as organizations that need to split the development work between specialists (for example, between user interface designers and programmers). The reasons that Struts has became popular for the servlet environment apply for portlets as well. This document introduces the Struts Portlet Framework, which adds support for writing Struts applications that can be deployed in WebSphere Portal Express.

Developers that have worked with Struts in the servlet environment should adapt easily to the Struts Portlet Framework. The packaging of a Struts portlet application is very similar to a Struts application in the servlet environment. However, WebSphere Portal Express also introduces additional concepts, such as portlet modes, multiple device support, and portlet communication, that might need to be addressed by the Struts application. This document provides information about special considerations when developing Struts applications for the portlet environment.

For more detailed information, you can find numerous sources available about Struts. The Struts website offers resources for developers, including a User Guide, Javadoc information, and the source code. There are also several excellent books on

Struts that are referenced from the Struts website. These references describe in detail the aspects of designing and implementing a Web application using Struts. For more information about Struts, see the link to the Struts Application Framework.

The Struts Portlet Framework ships several example WAR files, including source code, that demonstrate how to implement a Struts application in WebSphere Portal Express and also how to exploit portal features, such as portlet modes and device support. Throughout this documentation, source code from `SPFLegacyMailReader.war` and `SPFStandardMailer.war` is used to demonstrate the changes necessary to enable a Struts Web application to be deployed and run as a portlet. `SPFLegacyMailReader.war` is available on the WebSphere Portal Express Catalog.

The following subtopics describe the development of Struts applications in the portal environment.

"Changes to Struts application code" on page 2945
Learn about the main differences between a servlet-based Struts application and a Struts application created for the portal environment.

"Changes to Struts JSPs" on page 2955
The JSPs for Struts applications in the portal environment have to be modified to adapt to the way the portal server expects portlet URIs to be created. There are some changes to the tag library for HTML markup and additional tag libraries have been added to support cHTML and WML markup.

"Changes to configuration files" on page 2958
Struts Portlet Framework-specific init parameters were added so you can customize the Struts application for the portal environment. The portlet and web deployment descriptors require configurations specific to the Struts Portlet Framework. In addition, changes to the Struts configuration file must be made to specify a portal specific request processor as the controller.

"Supporting multiple Struts applications" on page 2972
Learn how web applications that allow packaging more than one Struts-based portlet in a single application can be created either using Struts modules or a namescoped servlet context.

"Create link tags in Struts" on page 2975
Get an overview of how to write tags to create links in a JSP in both a servlet and a portlet in Struts framework.

"Formatting XML documents with XSLT" on page 2976
Learn how a Struts application supports applying a style sheet to XML data.

"Static content in Struts" on page 2976
Portlet JSPs can link to the static content within the portlet's WAR directory structure by using the services of the portlet container to create portlet URIs from which the content can be referenced.

"Migrating existing Struts applications" on page 2977
Get an overview of how the existing Struts applications can be migrated using the Struts Portlet Framework so that these applications can be deployed in IBM WebSphere Portal Express.

Related concepts:

"Portlet communication" on page 3069
IBM WebSphere Portal Express supports multiple ways for portlets to exchange or share information.

Portlet creation basics
Get introduced to the concepts of portlet creation, starting with a simple portlet

that is modified throughout.

Related information:

The Struts Application Framework

Changes to Struts application code

Learn about the main differences between a servlet-based Struts application and a Struts application created for the portal environment.

Comparison of servlets and portlets

There are two main differences between the portlet and servlet environment that affects a Struts application in WebSphere Portal Express.

1. Action processing and rendering

   All servlet processing occurs during the service() method. The Struts rendering of the page is usually immediately preceded by action processing; they are essentially part of one step. The request and response object are passed to the service() method, which writes the resulting output to the response object. The servlet-based Struts RequestProcessor is designed to complete the Struts action processing and eventually forward, based on the URI, to a JSP or another Struts action.

   Portlet processing, however, is implemented in two phases, an event phase and a render phase. Action processing is performed prior to rendering the display view. Only the request object is passed to the portlet during the event phase. When a Struts application is migrated to the portlet environment, some of the information that was available during the event phase, namely the request parameters, is no longer available during the render phase. Additionally, since rendering methods, such as doView(), can be called when the portlet page is refreshed without a new event occurring for that portlet, all information required to render the page must be available every time that method is called.

   Note: The struts action is not invoked when moving between modes (view, edit, and configure). Only the service method is called (doView, doEdit) when switching modes. To cause the struts action to fire, place a call to that action in these methods.

   The Struts Portlet Framework provides a RequestProcessor that creates an IViewCommand. IViewCommand encapsulates the information so that the command object can be rendered at a later time, during the rendering method. IViewCommand and IViewCommandFactory are discussed in Saving information for rendering the view.

2. URI construction

URIs are constructed differently for portlets than for servlets. The portlet creates the URI programmatically using the PortletResponse object. The Struts Portlet Framework has modified the tags in Struts so that they create portal links. The Struts link tags behave the same as they do in the servlet environment, but the URL is a portlet URL and the Struts URL is passed as a parameter on the URL.

Saving information for rendering the view

A command pattern can be used to encapsulate the rendering of the view, and the information required during this rendering (See *Design Patterns: Elements of Reusable Object-Oriented Software* by Gamma, Helm, Johnson, and Vlissides for more information about the command design pattern). The pattern is implemented using the `IViewCommand` interface.

During the action processing, information needed to render the view needs to be saved, including the path to the page to be displayed. Also, for a JSP in a Struts application, the associated `ActionForm` is normally needed in order to complete the page. So, any `ActionForm` attributes need to be saved along with the path to the page to render. The attributes that are saved with a command should be limited because, since the action processing and view rendering steps involve separate requests, the command is saved in session. Since the session information may need to be serialized in a high availability environment, it is important to minimize the amount of data stored in the session wherever possible. Thus, the `ActionForm` is typically needed, this is saved by default.

In doView() processing, nothing needs to be known about the IViewCommand except that it should be executed. The execute method of the IViewCommand receives the request and response objects as parameters. As part of the processing, the previously saved attributes are populated into the request object that is passed in on the execute method.

Also, during the render phase, an execution context object (`ViewCommandExecutionContext`) is passed in on the execute call, which provides additional objects to help with the actual execution. You can save additional information with the view command and provide additional objects and information in the command execution context.

Note: For more information about `IViewCommand`, refer to the Javadoc documentation for the com.ibm.portal.struts.command package. The product Javadoc documentation is located at *`PortalServer_root`*`/doc/Javadoc/api_docs/` directory. Also see Using the command factory.

Updating ActionForm in the render phase

Some Struts applications may need to refresh the beans for the render phase every time the portlet renders. If the user is interacting with the portlet and clicking action links then the portlet will have an action phase and then followed by a render phase. The typical scenario is the case where an action phase does not precede the render phase. For example, if a user is interacting with another portlet on the page, then only that portlet has an action phase. The other portlets on the page may need to refresh the data to be rendered. The Struts action that implements the IStrutsPrepareRender action can read data from the model and update the ActionForm bean associated with the JSP through the information available through the ActionMapping. The page controller action is executed in the render phase of the portal server and therefore cannot change portlet state

or write to the model. It also cannot interact with other portlets using Property Broker or portlet messaging. The collaboration between portlets must occur during the action phase of the portal server.

The IStrutsPrepareRender interface is way to indicate that a Struts action should be called in the render phase of portal, for the specific task of refreshing the data to be displayed. When the Struts Portlet Framework Request Processor detects a Struts Action that implements the IStrutsPrepareRender in the action phase, the request processing is stopped and the ViewCommandFactory is called to create a WpsStrutsViewActionCommand object. The WpsStrutsViewActionCommand object is then executed in the render phase of the portal, at which time the request processor will be called again to execute the Struts action. A Struts action implementing IStrutsPrepareRender may forward to another Struts action that implements the Prepare-Render interface or to the JSP that renders the view. To see an example of how to use the IStrutsPrepareRender interface, see the Stock Quote portlet or Clock portlet. Both of these portlets are available for download from the WebSphere Portal Express Business Solutions Catalog.

Stock Portlet Sample:

The Stock Quote portlet is a Struts application that allows configuring companies in the edit portlet mode, and then the Stock Quotes for the portlet are displayed in the view mode. The quotes are supplied by a StockService, which returns random numbers, but the StockQuoteService could be replaced by a real service. The Stock Quote sample also uses a Struts Action that implements the IStrutsPrerpareRender so the stock quotes are refreshed even if a request phase does not precede the render phase.

```
public class RefreshAction extends Action implements IStrutsPrepareRender
{
  // ----------------------------------------------- Instance Variables
   /**
    * The Log instance for this application.
    */
    private Log log =
       LogFactory.getLog(this.getClass());

  // ------------------------------------------------ Public Methods
   /**
    * This action will refresh the stock quotes and then forwards to an
    * ActionForward for displaying the quotes.
    *
    * @param mapping The ActionMapping used to select this instance
    * @param form The optional ActionForm bean for this request (if any)
    * @param request The HTTP request we are processing
    * @param response The HTTP response we are creating
    *
    * @exception Exception if the application business logic throws
    *  an exception
    */
    public ActionForward execute(ActionMapping mapping,
                         ActionForm form,
                         HttpServletRequest request,
                         HttpServletResponse response)
           throws Exception
   {
      HttpSession httpSession = request.getSession();
        //-----------------------------------------------------------
        // Get the SubscribedNewsgroupsBean. This should have been
        // put in the session earlier
        //-----------------------------------------------------------
        SubscribedCompaniesBean subscribedCompaniesBean =
          ( SubscribedCompaniesBean )
            httpSession.getAttribute( Constants.SUBSCRIBED_COMPANIES );

        ActionForward actionForward = mapping.findForward("welcome");
```

```
        if ( subscribedCompaniesBean != null )
        {
          log.debug("Refresh quotes");

          Collection companies = subscribedCompaniesBean.getCcompanies();
          if ( companies != null )
          {
            StockQuoteService stockQuoteService =
                              new StockQuoteService();
            Iterator iterator = companies.iterator();
            while ( iterator.hasNext() )
            {
              CompanyBean companyBean = ( CompanyBean ) iterator.next();
              try
              {
                log.debug("Get quote for company " +
                                          companyBean.getSymbol() );
                float quote =
                      stockQuoteService.getQuote(companyBean.getSymbol());
                companyBean.setQuote( Float.toString( quote) );
              }
              catch ( Exception ex )
              {
                companyBean.setQuote( "---");
              }
              subscribedCompaniesBean.addCompany(companyBean.getName(),
                                                    companyBean );
            }
          }
        }
        return actionForward;
    }
}
```

Response object

During portlet action processing the response object is not available. The methods called during Struts processing expect both a request and response. To address the need for a response object, the Struts Portlet framework creates a temporary one. Other than for calling sendError(), you should not write to the response object. Instead your Action should return an ActionForward object and let the Struts RequestProcessor complete the doForward set.

sendError() processing

The typical Struts application has no need to access the response object during the Action processing. However, when an application checks and finds some state information invalid during action processing, it is not unusual for a Struts application to report the error using the response.sendError() method. The Struts Portlet Framework intercepts the calls to sendError() and saves the error information in the session. During the later view render phase, this error information is found, and the error information is displayed in lieu of displaying other content for the portlet.

Overriding Error Response Formatter

The Struts Portlet Framework provides default formatting for error responses. For standard portlets, the default error response formatter is `com.ibm.portal.struts.plugins.DefaultErrorResponseFormatter`. For IBM portlets, it is `com.ibm.wps.portlets.struts.plugins.DefaultErrorResponseFormatter`. This formatter can be overridden from within a Struts Plug-in. You can provide your own Plug-in which sets your own subclass of `DefaultErrorResponseFormatter`. The SPFLegacyPlugins.war and SPFStandardPlugins.war are examples of how to implement a custom error response formatter.

Saving information in a bean

A typical Struts application does not write directly to the response object and instead forwards to a JSP. The JSPs should not need access to the original event information, request parameters, which initially led to the current rendering. Instead, action processing should store information in a bean from which the JSP extracts the information. As long as the bean is available during the rendering, the rendering can be repeated. Since beans, such as `ActionForm`, are usually stored in the original request attributes, the request attributes from the original request processing must be saved in the iViewCommand and made available when doView() is invoked.

Forwards and redirects

Although you can write portlets using Struts, there are some aspects of the servlet environment that you cannot do in a Struts portlet. For example, Struts provides support for redirects and forwards to other servlets. These are provided because they are functions generally available to servlets. However, these capabilities are not available in portlets because WebSphere Portal Express does not support forwards or redirects.

The Struts `Action` object returns an ActionForward object, which contains a path to be either forwarded or redirected to. WebSphere Portal Express does not support forward operations because the response object has already been committed. If the path is not an action, include the page using the PortletContext.include() method. If the path is an action, then a forward to the Action is simulated. Forwards to other actions can be handled by recursively sending the new Action URI through the Struts base processing of an `Action`. Redirects are treated in the same way as a forward. This can be implemented using PortletApiUtils class. The following example demonstrates how to implement a forward to a Struts Action from an Action or a custom tag.

```
PortletApiUtils portletUtils = PortletApiUtils.getInstance();
if (portletUtils != null)
{
   portletUtils.forward( page, request );
}
else
{
   pageContext.forward( page, request );
}
```

The PortletApiUtils.getInstance() method is called to see if a PortletApiUtils instance is available. If a non null value is returned, then the execution is inside of WebSphere Portal. In this case, the portletApiUtils.foward() method is used instead of the PageContext.forward().

The issue with forwards affects tag handlers as well. In tag handlers, it is possible to access the PageContext object and invoke the forward method. An alternative method to the PageContext.forward() is available via the PortletApiUtils class that will provide a mechanism to emulate the forward functionality.

The tags in the PortalStrutsExample were modified to use the PortletApiUtil class instead of the `PageContext.forward`. The tags shipped with the Struts example are `CheckLogonTag`, `LinkSubscriptionTag`, and `LinkUserTag`. Here is a code sample of the change.

```
PortletApiUtils portletUtils = PortletApiUtils.getInstance();
if (portletUtils != null)
{
   // when run in a portlet use this execution path.
   Object pResponse = portletUtils.getPortletResponse( request );
   Object portletURI =
   portletUtils.createPortletURIWithStrutsURL(pResponse, url.toString() );
   // don't need to call response.encodeURL,
   // the portletURI.toString takes care of that.
   results.append(portletURI.toString() );
}
else
{
   // when run as a servlet, execute this path
   results.append(response.encodeURL(url.toString()));
}
```

If a forward is required in a JSP, then the logic forward tag is the suggested solution. The forward tag has been modified to use the PortletApiUtils forward implementation, and is the preferred method for a forward from a JSP. The PortletApiUtils can be obtained in a JSP through java code, but obtaining the Struts ModuleConfig to prefix the path is problematic. The logic forward tag handles these issues.

Customizing Commands

The section Comparison of servlets and portlets describes two phase processing of the portal server and the need for the `IViewCommand` objects that are created in the actionPerformed() phase. The Struts Portlet Framework ships several different commands that handle dynamic content, static content, XML data and error information from response.sendError(). This section describes some of the customizations that can be applied to a command, and also how to implement a command factory for creating new commands.

Struts View command

The Struts View command is the base class for commands in the Struts Portlet Framework.

- Standard portlets: com.ibm.portal.struts.command.StrutsViewCommand
- IBM portlets: com.ibm.wps.portlets.struts.WpsStrutsViewCommand

This class provides the foundation for saving the required information, including attributes, so the command can be rendered at a later time and multiple times. The Struts View command limits the number of attributes saved in the command object because these objects are saved in session. The command object needs to be available any time the portlet is refreshed so the command is saved in session. There is one command per portlet mode supported.

When the command object is created, Struts View command saves the request attributes that are configured to be saved. When the command is executed, it restores these attributes in the request object. An anticipated change to a command is the need for additional request attributes to be stored with the command. The Struts View command has two static methods that allow adding additional request attributes to save so this change can be made easily. The new attributes can be request attributes with a specific name, or attributes of a specific type.

If the rendering step requires additional attributes, then The Struts View command can be requested to save those attributes.

Struts View Jsp command

The Struts View Jsp command is the most commonly used command in the Struts Portlet Framework.

- Standard portlets: com.ibm.portal.struts.command.StrutsViewJspCommand
- IBM portlets: com.ibm.wps.portlets.struts.WpsStrutsViewJspCommand

The dynamic content rendered in the JSP most likely needs request attributes in the render step. The request attributes that are saved and made available in the render step are:

Attributes with the name

`Globals.MODULE_KEY`

`Globals.MESSAGES_KEY`

`Globals.ERROR_KEY`

Attributes of type

`ActionForm`

Adding request attributes stored with a command

The Struts application should call the Struts View command to add static methods before any `IViewCommand` objects are created. There are two ways to implement this customization of the Struts View command.

- Subclass `WpsStrutsPortlet` and implement the init() method. The `WpsStrutsPortlet` subclass then uses the following `WpsStrutsViewCommand` static methods.

  ```
  addAttributeNameToSave(String attributeName)
  addAttributeTypeToSave(Class attributeType)
  ```

- Use a Struts plug-in. The Struts plug-in is started when the configuration for the Struts module is read. The plug-in can be used to add the attributes that are needed by a command. Here is an example plug-in definition from the Struts configuration file.

  ```
  <plug-in
       className="com.ibm.struts.sample.plugins.ExampleAddAttributePlugin">
  </plug-in>
  ```

  Here is an example of the plug-in's init() method:

```
/**
 * Add attributes to save.
 *
 * @param config The ModuleConfig for our owning modules
 * @param servlet The ActionServlet
 *
 * @exception ServletException if we cannot configure
 * ourselves correctly
 */
public void init(ActionServlet actionServlet, ModuleConfig config)
throws ServletException
{
    // Call StrutsViewCommand.addAttributeNameToSave for each
    // request attribute we want to make available to the code
    // that renders our page. We can either specify the name
    // of the attribute or the class
    // type of attributes to save.
    StrutsViewCommand.addAttributeNameToSave("testAttribute");
}
```

Accessing request attributes

All request attributes that are set by the Struts action will be available when the WpsStrutsViewCommand is rendered the first time without the need to take these additional steps to save them. When a user interacts

with a portlet, this defines a single request and the same request object is available to the actionPerformed() method, where the Struts action is executed, and the service() method, where the rendering occurs. If the portlet is requested to refresh the view at a later time, the attributes might not be available. The refresh may be because the user is interacting with another portlet and the request object will not have the expected attributes. If the additional attributes are required during the refresh at a later time, then WpsStrutsViewCommand should be modified to save these attributes as described previously. Also, the rendering step should not rely directly on request parameters. Request parameters should be processed in the Action. However, request parameters can be saved automatically by Struts in the ActionForm. The ActionForm objects are available to the render phase.

Adding commands through the command factory

The Struts Portlet Framework uses the ViewCommandFactory object to create the IViewCommand objects for representing the view to render based on the given input information. The view command object typically contains the URL of the object (usually a JSP) creating the output for the portlet and other information needed for the rendering.

- For standard portlets, the default view command factory class is `com.ibm.portal.struts.plugins.ViewCommandFactory`.
- For IBM portlets, the default view command factory class is `com.ibm.wps.portlets.struts.plugins.ViewCommandFactory`.

The command object needs to contain the information to render the output properly. For a JSP within a Struts application, this is typically a form bean. It is possible that a particular portlet needs additional information in order to do the rendering. In such a case, you would need to add code to cause the additional information to be added to the command. The likely approach would be to override the saveAttributes() method in your subclass of the IViewCommand class. If the given URI (or perhaps type of URI) was one that would require additional information for rendering, then a new type of WpsStrutsViewCommand would need to be created which contained the needed information. The saveAttributes() method of this subclass of IViewCommand could then save the additional objects to be placed in the request object as attributes when the command was executed. To cause an existing request attribute to be saved, the saveAttribute method can be used to provide the request object and name of the attribute.

Note: The `SPFLegacyPlugins.war` and `SPFStandardPlugins.war` are examples of how to implement a custom ViewCommandFactory.

Controlling where commands are stored using the Command Manager Factory

The CommandManagerFactory allows a Struts application to specify where the IViewCommand objects are stored. The Command Manager is pluggable through the Command Manager Factory interface.

- Standard portlet container: The Command Factory plug-in is typically configured in the portlet deployment descriptor.
- IBM portlet container: The Command Factory plug-in is configured in the web deployment descriptor.

The default Command Manager Factory is the SessionCommandManagerFactory which stores the IViewCommands in the portlet session. The SessionCommandManagerFactory is the default command manager factory and recommended for portlets.

Note: The SPFLegacyCommandManager.war are examples of how to implement a custom CommandManagerFactory.

API enhancements

Ideally, portal-specific requirements should be minimized when writing Struts Applications. There are certain areas that need to be surfaced to the developer, like tag modifications and forwards. For those developing Struts applications specifically for the portlet environment, a class is provided to hide some of the servlet details that do not map well to execution in the portal server environment.

StrutsAction

The `StrutsAction` class provides an execute method that is passed portlet objects instead of servlet objects so a cast is not required to access portal-specific objects. The signatures for functions in the `StrutsAction` class are as follows:

Standard portlets

```
public ActionForward execute(ActionMapping mapping,
       ActionForm form, PortletRequest request) throws Exception;

public ActionForward execute(ActionMapping mapping,
                     ActionForm form, ActionRequest request,
                     ActionResponse response) throws Exception;

public ActionForward execute(ActionMapping mapping,
                     ActionForm form, RenderRequest request,
                     RenderResponse response) throws Exception;

public void sendError(ActionRequest request,
            int sc, String msg) throws IOException;

public void sendError(RenderRequest request,
            int sc, String msg) throws IOException;

public void sendError(ActionRequest request, int sc )
                                          throws IOException;

public void sendError(RenderRequest request,
                                  int sc ) throws IOException;
```

IBM portlets

```
public ActionForward execute(ActionMapping mapping,
                     ActionForm form, PortletRequest request)
                                       throws Exception;

public ActionForward execute(ActionMapping mapping,
                     ActionForm form, PortletRequest request,
                     PortletResponse response) throws Exception;

public void sendError(PortletRequest portletRequest,
                        int sc, String msg) throws IOException;

public void sendError(PortletRequest portletRequest,
                                  int sc) throws IOException;
```

LookupDispatchAction and DispatchAction

The `WpsLookupDispatchAction` and `WpsDispatchAction` classes offer the same functionality as their Jakarta `LookupDispatchAction` and `DispatchAction` counterparts.

- Standard portlets

  ```
  com.ibm.portal.struts.action.LookupDispatchAction
  com.ibm.portal.struts.action.DispatchAction
  ```

- IBM portlets

```
com.ibm.wps.struts.action.WpsLookupDispatchAction
com.ibm.wps.struts.action.WpsDispatchAction
```

`DispatchAction` extends `StrutsAction`, so the `StrutsAction` methods can be used for the sendError() implementation.

In the following getKeyMethodMap() example, the add and delete buttons each have their own methods for processing.

```
protected Map getKeyMethodMap() {
      Map map = new HashMap();
      map.put("button.add", "add");
      map.put("button.delete", "delete");
      return map;
  }
```

The `LookupDispatchAction` subclass implements the following two methods.

```
public ActionForward add(ActionMapping mapping,
                     ActionForm form, PortletRequest request )
                     throws IOException, ServletException {
      // do add
      return mapping.findForward("success");
  }
  public ActionForward delete(ActionMapping mapping,
                       ActionForm form, PortletRequest request )
                       throws IOException, ServletException {
 *      // do delete
 *      return mapping.findForward("success");
 *  }
 *
```

The Struts Portlet Framework implementation of `LookupDispatchAction` is not passed a response object and the request object is a `PortletRequest`. These two classes are a convenience for applications that are intended for the portal environment only.

Accessing portlet objects

There will be situations when a Struts Action needs to access objects in the Portlet API. For example, the Struts Action might need to access the PortletRequest object to get to PortletSettings or PortletPreferences, depending on the container. The PortletApiUtils should be used to obtain the PortletRequest object from the HttpServletRequest object.

```
PortletApiUtils portletUtils = PortletApiUtils.getUtilsInstance();

if (portletUtils != null)
{

   PortletRequest portletRequest = (PortletRequest)
   portletUtils.getPortletRequest( request );
}
```

The Struts Action class also makes the ActionServlet class available. When the Struts application is authored with the Struts Portlet Framework, the ActionServlet class is actually the `WPActionServlet`. For more information see the Javadoc documentation for:

- Standard portlet container: com.ibm.portal.struts.portlet.WpActionServlet class information.
- IBM portlet container: com.ibm.wps.portlet.struts.WpsActionServlet class information.

Sending a message

The Struts examples use the Property Broker for sending messages from one portlet to another.

The Struts Portlet Framework ships the SPFLegacyMultipleModules, and SPFStandardMultiplModules as examples of using the property broker and Struts Portlet Framework

Changes to Struts JSPs

The JSPs for Struts applications in the portal environment have to be modified to adapt to the way the portal server expects portlet URIs to be created. There are some changes to the tag library for HTML markup and additional tag libraries have been added to support cHTML and WML markup.

Creating portlet URIs

The Struts application paths, both to actions and to pages, must be sent and retrieved using portlet URIs. Portlet URIs have a specific format. A special API is used to generate the URI and add the required information to be passed to the portlet. If portlet URIs were not used, control would not get passed to the correct portlet. Thus, the portlet URIs must be used to get control passed to the correct portlet, with the additional path information needed by the Struts application made available. The Struts tags have been modified to automatically provide this needed functionality.

Struts Action mappings are defined in terms of paths. The name and location of page objects (for example, JSPs) are also defined using paths. Thus, although portlets have their own form of URI, it is still necessary to associate the Struts path with an action sent to a portlet and to retrieve that Struts path when the portlet action is handled. The Struts URI is passed as a parameter so the Struts request processor can process the action. If the link is not an action then the portlet processes the information and create the appropriate IViewCommand.

Typically a Struts application passes parameters on such a path using the query string on the HTTP URL. Often the actions containing these paths are generated from tags provided by Struts. The most obvious examples of these are the tags for the HTML elements `<link>` and `<form>`. For IBM Struts portlets, the **`urlType`** attribute must be included in the <html:form>, <html:link> and <html:rewrite> tags to support the different types of URIs. This attribute can take the following values:

- return - Creates a portlet return URL indicating the caller of the portlet. For example, this URL can be useful for creating a form action that can take a user back to view mode from edit mode.
- standard - Creates a standard portlet URL.

Examples:

<html:rewrite>

```
<link rel="stylesheet" type="text/css"
      href="<html:rewrite page='/assets/styles/base.css'/>">
```

<html:form> (Standard portlets)

```
<html:form action="/saveConfiguration.do" portletMode="view">
```

Style sheets

Many existing Struts application use the rewrite tag to create a link element for a cascading style sheet. This is not the documented intention of the rewrite tag, which is supposed to create the same path as the link tag without the <a> element. Since the Struts Portlet Framework had to modify how links are created, the rewrite tag required some customizations to be used to create link elements for style sheets. The rewrite tag will create the same path as the link tag, except when the page or forward reference is to a CSS file. In the case where a CSS file is referenced, the rewrite tag will use the Jakarta Struts implementation, which results in a path to the CSS file. Here are examples of how to create link elements for style sheets using the Struts Portlet Framework.

Using a forward

```
<link rel="stylesheet" type="text/css"
                       href="<html:rewrite forward='baseStyle'/>">
```

Using a page

```
<link rel="stylesheet" type="text/css"
                       href="<html:rewrite page='/basestyle.css'/>">
```

Using the portlet tags

```
<%@ taglib uri="/WEB-INF/tld/portlet.tld" prefix="api" %>
<api:init />
<link rel="stylesheet" type="text/css"
      href="<%= portletResponse.encodeURL( basestyle.css ) %>">
```

Markup support

The Struts tag library has been modified to support the additional markup languages supported by WebSphere Portal Express. For HTML, the tags that create links have been modified to support portlet URIs. See Creating portlet URIs for details. Also, be aware of the restrictions for HTML output described in “Markup guidelines” on page 3123.

There might be a cases where the JSPs for a Struts application need to run in both the servlet and portlet environment. For this reason, page level tags are implemented in tag libraries. The Struts application can use them in its JSPs, but the tags will not generate markup when executed within WebSphere Portal.

You should also refrain from setting color, and fonts. The portal server supports skins and themes that give the page a consistent look and feel. The JSP should be authored so it adheres to the conventions of the theme by using the appropriate style sheet.

Using the cHTML tags

The use of the cHTML tags is similar to the use of the HTML tags. The name of the cHTML tag library file is `struts-chtml.tld`. The following is an example of the cHTML taglib definition.

```
<%@ taglib uri="/WEB-INF/struts-chtml.tld" prefix="chtml" %>
```

Using the WML tags

The WML tags are a new addition for creating a user experience in WAP

devices. The use of the WML tags is similar to the use of the Struts HTML tags. Use the following directive to make these tags available to a JSP.

```
<%@ taglib uri="/WEB-INF/struts-wml.tld" prefix="wml" %>
```

The use of the WML tags provided with this distribution is similar to the use of Struts HTML tags. The name of the WML tag library file is `struts-wml.tld`. The following is an example of the WML version of `index.jsp` .

```
<%@ taglib uri="/WEB-INF/struts-bean.tld" prefix="bean" %>
<%@ taglib uri="/WEB-INF/struts-wml.tld" prefix="wml" %>
<%@ taglib uri="/WEB-INF/struts-logic.tld" prefix="logic" %>
<p>
<wml:link page="/editRegistration.do?action=Create">
   <bean:message key="index.registration"/>
</wml:link>
<br/>
<wml:link page="/logon.jsp">
   <bean:message key="index.logon"/>
</wml:link>
<br/>
<wml:link page="/tour.do">
   <bean:message key="index.tour"/>
</wml:link>
</p>
```

See General tips for portlet output and the WML markup guide for general guidelines about providing WML markup.

WML does not have a <form> element like HTML. Struts, however, uses the <form> tag for the scoping of parameters and supporting form beans. For that reason, the WML implementation also includes a <form> tag to support some of the Struts features in WML. The <form> tag in the WML taglib takes an action as an attribute. The following is an example of a form in WML.

```
<wml:form action="/logon">

<do type="options" label="send">

  <wml:go method="post">
    <postfield name="username" value="$username"/>
    <postfield name="password" value="$password"/>
  </wml:go>
</do>

<bean:message key="prompt.username"/><wml:text property="username"/>
<bean:message key="prompt.password"/>
<wml:password property="password" size="16" maxlength="16"/>

</p>

</wml:form>
```

Figure 42. Example form for WML

The JSPs for the HTML Struts example have been modified for the WML markup. This example also demonstrates the use of the WML taglib. These JSPs can be found in the `/wml/view` directory of the expanded `PortalStrutsExample.war` file.

The tags in the Struts portlet WML tag library can throw a `JspException` at run time. An error page can be declared using the `<%@ page %>` directive to

detect the exception and supply information about the error condition. The actual exception is passed as an attribute in the request object with the key `org.apache.struts.action.EXCEPTION`.

The following is a brief description of each tag. See Detailed descriptions of the Struts WML tags for more information.

Table 455. Summary of WML tags for Struts portlets

Tag	Description
<wml:cancel/>	Renders a WML `<postfield>` element with a value of `cancel`.
<wml:card/>	Renders a WML card element.
<wml:errors/>	Retrieves the set of error messages from the request object with the default key of `Action.ERROR_KEY` or the value specified by attribute name.
<wml:form/>	Does not render any markup, but it is used to scope beans and transactions.
<wml:go/>	Renders a WML <go> element.
<wml:head/>	Renders a WML <head> element with language attributes extracted from the user's current Locale object, if there is one.
<wml:link/>	Renders a WML <a> element as a hyperlink to the specified URL.
<wml:option/>	Renders a WML <option> element, representing one of the choices for an enclosing <select> element.
<wml:options/>	Renders a set of WML <option> elements, representing possible choices for a <select> element.
<wml:password/>	Renders a WML <input> element of type password, populated from the specified value or the specified property of the bean associated with our current form.
<wml:postfield/>	Renders a WML <postfield> element.
<wml:rewrite/>	Renders a request URI based on exactly the same rules as the link tag does, but without creating the <a> hyperlink.
<wml:select/>	Renders a WML <select> element, associated with a bean property specified by our attributes.
<wml:text/>	Renders a WML <input> element of type text, populated from the specified value or the specified property of the bean associated with our current form.
<wml:wml/>	Renders a <wml> element.

Changes to configuration files

Struts Portlet Framework-specific init parameters were added so you can customize the Struts application for the portal environment. The portlet and web deployment descriptors require configurations specific to the Struts Portlet Framework. In addition, changes to the Struts configuration file must be made to specify a portal specific request processor as the controller.

Configuring the Struts application for the IBM container

Most of the configuration for a Struts application is in the web deployment descriptor file (`web.xml`). The servlet class that is specified in the web deployment descriptor must be the portlet servlet. The class name is `WpsStrutsPortlet`. Here is an example definition.

```
<servlet id="Servlet_1">
    <servlet-name>MyStrutsApp</servlet-name>
    <servlet-class>com.ibm.wps.portlets.struts.WpsStrutsPortlet
    </servlet-class>
    ...
</servlet>
```

When subclassing WpsStrutsPortlet, you must designate the subclass name for <servlet-name/>. The servlet **`id`** must be suffixed with a unique ID to prevent conflicts with other portlets. See "Deployment descriptors" on page 3129 for more information.

Initialization parameters

The following is a list of names of the initialization parameters that can be configured in the web deployment descriptor for Struts portlets with the `<init-param>` element.

CommandManagerPlugin

This configuration parameter allows setting a pluggable CommandManager.

struts-servlet-mapping

Identifies a Struts action mapping so that paths that must be treated as Struts actions can be recognized. See Servlet mapping for further details.

ModuleSearchPath

Indicates the search path for Struts modules, default value is the markupName and mode. See Using modes for more information.

IncludesSearchPath

Indicates a more fine-tuned search path for including a file. If this value is not specified, then PortletContext.include() is used instead.

NamescopeServletContext

This allows the WpsStrutsPortlet to be defined in the web deployment descriptor more than once by namescoping the parameters in the servlet context.

NamescopeFormName
The default behavior is for the Struts form tag to namescope the name of the portlet. Setting this parameter to false will allow the behavior to be overridden.

UseGroupsForAccess
Configures the Struts portlet to use portal server group names as if they were role names. Struts uses roles to determine access to an `Action`. Set this parameter to **`true`** to configure the Struts application to use the portal's group names as if they were role names to emulate role-based access control.

UsePortalsLocale
This parameter allows a Struts application to override default behavior of using the portal's locale. If this parameter is set to false the Struts application can use the Request processor's processLocale method for controlling locale or set the locale in a Struts action; otherwise, it uses the portal's locale and also supports dynamic changes. The default value is true.

WelcomeFileSearchPath
This configuration parameter allows customizing the initial view page search.

Servlet mapping
In a portlet, you must use a servlet URL mapping for your portlet. So, one would typically use a path prefix servlet mapping similar to the following.

```
<servlet-mapping id="ServletMapping_1">
     <servlet-name>MyStrutsApp</servlet-name>
     <url-pattern>/Struts/*</url-pattern>
</servlet-mapping>
```

The typical Struts application uses an extension mapping. The Struts example uses extension mapping and it is expected that most Struts applications use the extension mapping "`*.do`". In the normal Struts case, this means that URIs ending in "`*.do`" are directed to the Struts servlet. Struts provides a single servlet that you can use (or subclass) for all of your Struts applications. When you want to cause control to be given to a Struts Action when a link is selected or form submitted, use the extension from the extension mapping, for example "`*.do`" on the URI so that the processing is routed to the Struts ActionServlet. Other URIs, such as those for JSPs, do not have this extension and, thus, would not be routed through the Action Servlet for processing.

A Struts application, which is itself a servlet application, also requires a servlet mapping. Although the Struts portlet already has a servlet mapping as shown previously, you still must identify what to use as a Struts action mapping so that paths that must be treated as Struts actions can be recognized. To specify the pseudo servlet mapping that is used by Struts, specify it as a servlet initialization parameter in the `web.xml` file, as follows:

```
 <init-param>
     <param-name>struts-servlet-mapping</param-name>
     <param-value>*.do</param-value>
 </init-param>
```

The Struts Portlet Framework supports the same servlet mappings modes as the servlet-based Struts application. However, the current Struts implementation supports extension mapping for the module implementation. Since the portal server implementation uses modules for the mode and markup support, use the extension mapping.

Configuring the Struts application for the standard portlet container

In the standard container the Struts application is configured in the portlet deployment descriptor. The portlet class and init parameters are specified in the portlet deployment description. In the standard container, the only configuration in the web deployment descriptor is the welcome file list and the tag library (taglib) elements. The portlet class that is specified in the portlet deployment descriptor must be the `com.ibm.portal.struts.portlet.StrutsPortlet` class or subclass.

```
<portlet>
  <portlet-class>com.ibm.portal.struts.portlet.StrutsPortlet</portlet-class>
</portlet>
```

If subclassing the `StrutsPortlet`, the subclass class name must be designated as the <portlet-class/>.

Initialization parameters

The following is a list of names of the initialization parameters that can be configured in the web deployment descriptor for Struts portlets, with the `<init-param>` element):

IncludesSearchPath
Indicates a more fine-tuned search path for including a file. If this value is not specified, then PortletContext.include() is used instead.

ModuleSearchPath
Indicates the search path for Struts modules, default value is the markupName and mode. See Support for multiple Struts applications for more information

NamescopeFormName
The default behavior is for the Struts form tag to namescope the name of the portlet. Setting this parameter to false allows the behavior to be overridden.

NamescopeServletContext
This allows the WpsStrutsPortlet to be defined in the web deployment descriptor more than once by namescoping the parameters in the servlet context.

struts-servlet-mapping
Identifies a Struts action mapping so that paths that must be treated as Struts actions can be recognized. See Servlet mapping for further details.

UseGroupsForAccess
Configures the Struts portlet to use portal server group names as if they were role names. Struts uses roles to determine access to an Action. However, WebSphere Portal V 4.2 does not support roles. Set this parameter to true to configure the Struts application to use the portal's group names as if they were role names to emulate role-based access control.

UsePortalsLocale
This parameter allows a Struts application to override default behavior of using the portal's locale. If this parameter is set to false, the Struts application can use the Request processor's processLocale method for controlling locale or set the locale in a Struts action. Otherwise, it uses the portal's locale and also supports dynamic changes. The default value is true.

WelcomeFileSearchPath
This configuration parameter allows customizing the initial view page search.

Setting an initial view

The Struts application must specify what the initial view is for the application. The typical way that the initial screen is specified is through the welcome file list. The Struts Portlet Framework supports the welcome file list configuration, and has some additional features. The servlet implementation of the welcome file list only supports JSP and HTML pages. The Struts Portlet Framework supports a JSP or HTML welcome file, and also allows a Struts action to be specified as the welcome file. See Example 1 for implementation details.

The alternative method allows Welcome files to be configured on a per portlet basis as a configuration parameter in the portlet deployment descriptor `portlet.xml`. The default search looks for the initial view file per portlet mode. The other feature in the Struts Portlet Framework is the support for modules. A unique welcome file must be configured for each module by specifying the application prefix. For example, to support the view mode for the markup WML, a welcome file `wml/view` can be specified. See Example 2 for implementation details.

```
<welcome-file-list>
  <welcome-file>index.jsp</welcome-file>
  <welcome-file>help/index.jsp</welcome-file>
  <welcome-file>wml/view/index.jsp</welcome-file>
</welcome-file-list>
```

Figure 43. Example 1: Welcome file list that is specified in the file `web.xml`

```
<portlet-preferences>
   <preference>
      <name>viewMode.page</name>
      <value>portlet1/welcome.jsp</value>
   </preference>
   <preference>
      <name>editMode.page</name>
      <value>html/edit/index.jsp</value>
   </preference>
</portlet-preferences>
```

Figure 44. Example 2: Welcome file list that is specified on a per portlet basis (standard portlets)

```
<config-param>
    <param-name>viewMode.page</param-name>
    <param-value>portlet1/welcome.jsp</param-value>
 </config-param>
 <config-param>
    <param-name>editMode.page</param-name>
    <param-value>portlet1/edit.jsp</param-value>
 </config-param>
```

Figure 45. Example 3: Welcome file list that is specified on a per portlet basis (IBM)

Note:

- A welcome file entry must be configured for each module. The welcome file entry is the initial screen that is shown when first switching to that module.
- The default Struts configuration, which has a "" prefix, is used any time a module prefix is not found.

Using modes

The Struts Portlet Framework is set up to use the mode and markup to find the Struts configuration. However, if you do not want to use a Struts application in one of the modes, the Struts portlet `(Wps)StrutsPortlet` must be subclassed. The appropriate service() method must be overridden to implement the wanted behavior. For example, if the developer wants a help mode that does not use a Struts application, then the doHelp() method must be implemented in a subclass.

Portlet modes can be supported by the Struts Portlet framework. The Struts Portlet Framework stores the IViewCommand objects on a per mode basis. That means that an application that supports view and edit mode has an IViewCommand object for view and edit mode. The welcome file list is typically used for the initial page when entering a mode.

The `SPFLegacyMultipleModules.war` example for the IBM container and the `SPFStandardMultipleModules.war` example for the standard portlet container demonstrate support for edit mode. The Module 1 portlet implements an edit mode wizard that steps through three screens. The example also demonstrates how to use a StrutsActionForward so that the IViewCommand object for the last page of the wizard can be marked for removal on a mode change. This allows the IViewCommand object for edit mode to be deleted when the user switches from edit mode back to view. The anticipated use is that the user is entering data in edit mode and the configuration data is submitted to be stored. The last page displays that the data has been successfully stored. The user then leaves edit mode with the skin. If the user goes back into edit mode, they do not want to see the message that the data was successfully stored. Since the command was deleted, the welcome file list entry can be used to start the wizard again. This feature also reduces the number of attributes that are stored in session.

```
<action path="/nextEditPage2"
        type="com.ibm.wps.struts.example.multipleapps.actions.ForwardAction">
        <forward className="com.ibm.wps.struts.action.StrutsActionForward"
                 name="success"
                 path="/edit3.jsp">
                <set-property property="removeOnModeChange"
                              value="true"/>
         </forward>
</action>
```

The Struts example application demonstrates support for help mode. A Struts configuration was added to the web deployment descriptor, `web.xml`, to demonstrate help mode. This mode is entered when the user clicks the help button for the Struts example portlet. The following code fragment from the web deployment descriptor demonstrates how to configure the struts-configuration to support help mode.

```
<init-param>
 <param-name>config/html/help</param-name>
 <param-value>/WEB-INF/html/help/struts-config.xml</param-value>
</init-param>
```

The search path that is set in the web deployment descriptor includes mode, which enables a separate Struts configuration for a mode.

```
<init-param>
 <param-name>ModuleSearchPath</param-name>
 <param-value>markupName, porletMode</param-value>
</init-param>
```

The welcome file list defines the welcome file for the help mode, as follows:

```
<welcome-file-list>
 <welcome-file>index.jsp</welcome-file>
 <welcome-file>html/help/index.jsp</welcome-file>
 <welcome-file>wml/view/index.jsp</welcome-file>
</welcome-file-list>
```

Support for devices

The Struts Portlet Framework's support of devices was flexible. The way the configuration and JSP files are placed in the directory structure must be tolerant of the devices that must be supported. Some applications can support HTML browsers and need a single Struts configuration file. Other applications can support multiple markups and locales but do not need to know specifics about the device itself, while other implementations need unique implementations for each of the supported devices.

Note: This support requires extension mapping. The prefix mapping supports only a single configuration file.

WebSphere Portal Express is designed to allow portlets to access information about the client device, such as locale and markup language supported with the Client interface. The Struts support for modes and markups was implemented by allowing a configuration parameter to specify the fields in the client object that are interesting. The Client object is treated like a bean, where the field value corresponds to a get method in the Client object. This allows developers to specify the fields in the Client object that are necessary to their device support.

Note: The standard portlet container does not have a Client object but the following fields are supported:

- markupName
- locale
- portletMode

There are two paths that can be specified for supporting modes and device types (clients). The first path is the search path, and that path is used to specify the module to use. The search path locates the appropriate Struts configuration to use and is the prefix that is used as the base directory for the necessary JSPs. The default search path is *markupName, portletMode*. This searches for a configuration that is based on the current markup and the portlet mode.

```
<init-param>
  <param-name>ModuleSearchPath</param-name>
  <param-value>markupName, portletMode</param-value>
</init-param>
```

For example, if a desktop browser is used and the user is in view mode, then the search path would be:

- html/view
- html
- "" (default configuration)

And if the Client object specified "wml" as the markupName then the search path would be:

- wml/view
- wml
- "" (default configuration)

The search path is checked against all of the prefixes for the ModuleConfig objects for a match. The modules are configured in the web deployment descriptor, `web.xml`, by specifying a prefix for a Struts configuration.

For example, a module for the WML markup in view mode would be configured by adding a configuration to the web deployment descriptor like:

```
<init-param>
   <param-name>config/wml/view</param-name>
   <param-value>/WEB-INF/wml/view/struts-config.xml</param-value>
</init-param>
```

If the `wml/view` module is selected, the corresponding JSPs are searched for under the `wml/view` directory of the Struts application WAR file.

This is a good start for separating the files for different markups and modes, but this granularity is probably not enough to deal with all available devices, or even locales. You can modify this search path to have a finer granularity. For example, you can create a unique module for every phone that they support.

The search path of markupName, portletMode, locale, manufacturer, model, and version can be used. This offers a fine-grained deployment of the JSPs. The problem with this approach is that most of the JSPs would probably be the same for each of the devices. There are probably a few that must be modified for each device, but this approach does not allow sharing.

This is why two paths are offered, search and include. The search path is used to find the module. This would be for applications that can use a common configuration, share actions, and some of the JSPs. The include path can then be used to look for specific implementations of the JSP and if not found, fall back to a common file.

Take the following include path for including JSPs that are dependent on the device.

```
<init-param>
   <param-name>IncludesSearchPath</param-name>
   <param-value>manufacturer, model, version</param-value>
</init-param>
```

This would search for the file relative to the module and then take the requested fields into account. If the search path found the `wml/view` module and the file `index.jsp` is requested, then the previous search path for an ACME T400 would be as follows:

- wml/view/Acme/T400/index.jsp
- wml/view/Acme/index.jsp
- wml/view/index.jsp

That version did not show up in the search. The default client configuration for the T400 did not have one defined. If one was configured later, then it would be used.

Portal aggregation is flexible in how the differences between devices are supported. The common features are supported through the search path; all WML devices in view mode in a locale. The subtitle differences between devices would be supported through the file search of the include path. The common versions of the JSPs would be at a root level, and specific files would be in subordinate directories.

The Struts example includes several tags, two of which required a minor change to support modules. The LinkSubscriptionTag and LinkUserTag tags had to be modified so that the current ModuleConfig's prefix had to be added to the computed URL.

Globalization support

The Struts Portlet Framework supports i18n by using resource bundles. Each Struts application can specify a message resource. The Java resource bundle implementation determines the resource bundle to use based on the locale and variant. The Struts application typically uses the message tag to include translated text from the application-supplied resource bundle.

```
<init-param>
  <param-name>IncludesSearchPath</param-name>
  <param-value>locale</param-value>
</init-param>
```

The Struts framework was designed to give the application control over locale by using a locale object that is set in session with a known key. The Struts RequestProcessor sets the locale object for each user during the processLocale() method. The processLocale() method sets the locale object using the key Globals.LOCALE_KEY if the feature is enabled through the Struts configuration file and the attribute was not already set in session.

A setting on the controller element that is available in the Struts configuration file controls the locale support in Struts. The support can be turned off, which stops the locale processing in the Request processor processLocale() method.

WebSphere Portal supports dynamic changes to the locale. Portlets must use the locale object that is obtained through the PortletRequest object for the locale setting. The Struts Portlet Framework sets the locale object in the session with the key Globals.LOCALE_KEY during the actionPerformed() and service() methods. The support is implemented by calling the (Wps)StrutsPortlet processLocale() method, which sets the locale object if one is not found or the locale is different from the one already in session. This assures that a portlet written using the Struts Portlet Framework uses the correct locale.

Locale support can also be customized. The (Wps)StrutsPortlet processLocale() method can be overridden for a customized implementation. The invocation of (Wps)StrutsPortlet processLocale() can also be stopped by adding the following init-parameter to the web deployment descriptor for the application.

```
<init-param>
  <param-name>UsePortalsLocale</param-name>
  <param-value>false</param-value>
</init-param>
```

However, in most cases the portlet must use the default implementation.

Using the IBM container

WebSphere Portal also provides an implementation to find globalized JSPs using the include() method on PortletContext. PortletContext inserts the locale into the path to locate the locale-specific JSP. For more information, see Generating markup for multiple devices, markups, and languages.

The Struts Application can be deployed to use the portal server's aggregation method to find globalized JSPs, instead of a resource bundle for each locale. The

Struts Portlet Framework uses the PortletContext.include() on the existing container to include JSP pages, so both methods of globalization are supported.

PortletContext.include() searches for locale-specific files by inserting the markup and locale name in the path. This behavior can be configured through an initialization parameter in the web deployment descriptor. An include search path can be specified that controls how the includes are located. For example, the default implementation would be markup, locale but can be changed to just locale if required. The mode and markup support may have been already configured to include markup, so this configuration allows the markup to not be needed a second time.

WML support

Because WebSphere Portal Express supports pervasive devices, such as WAP phones, the example was configured to support a generic WML device. Adding a Struts configuration for WML devices supports the view mode for a WML device. The following code fragment demonstrates how to set up the struts-configuration for the view mode for WML markup.

```
<init-param>
 <param-name>config/wml/view</param-name>
 <param-value>/WEB-INF/wml/view/struts-config.xml</param-value>
</init-param>
```

The search path that is set up in the web deployment descriptor, takes markup, mode, and locale into account. When a WAP device connects to a portal server, the markup is WML, and the mode is view. This example does not use the locale that is requested.

The welcome file list also defines the welcome file for the `wml/view` mode.

```
<welcome-file-list>
 <welcome-file>index.jsp</welcome-file>
 <welcome-file>html/help/index.jsp</welcome-file>
 <welcome-file>wml/view/index.jsp</welcome-file>
</welcome-file-list>
```

For more information about WML support in the Struts Portlet Framework, see "Changes to Struts JSPs" on page 2955.

Logging

The Struts Portlet Framework uses the Commons-Logging interface for a logging facility. The Struts Portlet Framework was supplied an implementation of the Commons-Logging Log interface that can be used to map the trace messages to the logging facility used by the portal server. This file is normally found in the *PortalServer_root*/log directory. Setting properties in the *wp_profile_root*/PortalServer/config/config/log.properties file enables trace logging. The trace string for tracing a Struts portlet application can be configured in `log.properties`. The trace string can be modified to add or remove classes. The entire trace string must be on one line in `log.properties`.

The logging in WebSphere Portal Express typically creates a logging object per class and the trace messages open the log class that the message is from. It is convenient to determine where a trace message originated from and also allows enabling the tracing on certain classes.

The typical trace string that is configured in the log.properties file for tracing a Struts application would be as the following examples (all on one line). This can be modified to add or remove classes.

IBM container:

```
traceString=
org.apache.struts.*=all:
com.ibm.wps.portlets.struts.*=all:
com.ibm.wps.struts.common.*=all:
com.ibm.wps.struts.base.*=all
```

Some of the Apache source uses a string to specify the class name and, in some cases, this string is not be a valid class. In those cases the class name in the trace message is com.ibm.wps.portlets.struts.logging.WpsStrutsTraceLogger. This package or class must be specified in the traceString to account for these occasions.

The log messages from Struts use the Portal Server's logging facility when logging is enabled. When this is the case, the log messages from Struts do not open the correct method name as part of the log message. There is a configuration parameter to enable correcting this problem, but this is an expensive operation. This feature is not required as much with the logging changes implemented in this release.

JSR-compliant container:

```
traceString=
org.apache.struts.*=all:
com.ibm.portal.struts.*=all:
com.ibm.wps.struts.common.*=all:
com.ibm.wps.struts.base.*=all
```

Customizing logging

The Struts Portlet Framework specifies the commons logging Log Factory in the META-INF/services directory. In the samples, the org.apache.commons.logging.LogFactory file specifies the default logger that the Struts Portlet Framework uses. The log factory class name is dependent on the WebSphere Portal Express container. The org.apache.commons.logging.LogFactory file can be edited to specify other commons log factories.

Using the default logging implementation class: Struts Portlet Framework 5.0.3 or later Struts Portlet Framework 5.0.3 and later no longer use the System property org.apache.commons.logging.LogFactory to specify the log factory class. The commons log factory can be specified either in the META-INF/services directory or the commons-logging.properties file that must be found through the class path. The suggested way to specify the commons log factory is to through the META-INF/services directory of the WAR file.

IBM container with a Struts Portlet Framework version prior to 5.0.3: Versions of the Struts Portlet Framework before SPF 5.0.3 set the logging implementation class with the System property org.apache.commons.logging.LogFactory. The portlet initialization method, init, in com.ibm.wps.portlets.struts.WpsStrutsPortlet calls the setCommonsLogFactory method.

```
public void setCommonsLogFactory()
{
   String methodName = "setCommonsLogFactory";
```

```
    s_traceLogger.text( WpsStrutsTraceLogger.TRACE, methodName,
                "Set common logging to use StrutsLogFactory" );
    System.setProperty( "org.apache.commons.logging.LogFactory",
                "com.ibm.wps.portlets.struts.logging.StrutsLogFactory");
}
```

Note: A portlet must not use this mechanism, as the system property affects other applications in the WebSphere Application Server. If an older version of the Struts Portlet Framework is used, first the WpsStrutsPortlet must be extended and the setCommonsLogFactory must be implemented to do nothing. Then modify the application to specify the commons log factory through the org.apache.commons.logging.LogFactory file in the `META-INF/services` directory of the application.

Struts Portlet Framework version 5.0.3 or later Log Factory's class name

Standard portlet container: org.apache.commons.logging.LogFactory

IBM portlet container: com.ibm.wps.portlets.struts.logging.StrutsLogFactory

Overriding the default logging implementation class

Struts portlet developers can override the default logging implementation class. Depending on the version of Struts, the implementation varies.

Struts Portlet Framework 5.0.3 or later There are two ways to customize the logger. The first and recommended option is to specify the log factory in the org.apache.commons.logging.LogFactory file in the `META-INF/services` directory. The Commons Logging implementation checks the `META-INF/services` directory before looking for the `commons-logging.properties` file in the class path to determine the class name of the LogFactory. The second option is to specify the `commons-logging.properties` file in the application's class path. The org.apache.commons.logging.LogFactory file can be updated to specify the required LogFactory.

Note: Version 5.0.3 of the Struts Portlet Framework used the `commons-logging.properties` file to specify the logger, and this file was packaged in the `wp.struts-commons-logging.jar`. Specifying the log factory in the `META-INF/services` directory takes precedent over the log factory specified in the `wp.struts-commons-logging.jar` file.

IBM container with a Struts Portlet Framework version prior to 5.0.3: To override the default logging implementation class, developers can extend com.ibm.wps.portlets.struts.WpsStrutsPortlet and override the setCommonsLogFactory method. The developer could remove the call to set the org.apache.commons.logging.LogFactory System property in setCommonsLogFactory, and create the file org.apache.commons.logging.LogFactory in the `META-INF/services` directory (or the file commons-logging.properties in the `WEB-INF/classes` directory of the application to determine the logging implementation class. The `META-INF/services` directory is the preferred method.

Note: The portlet web deployment descriptor will then need to be updated to specify the WpsStrutsPortlet subclass as the servlet class.

Logging conflict between new and old versions of Struts portlets

The JCL determines the logging implementation by first looking for the System property org.apache.commons.logging.LogFactory. If the System property has not been set, it then looks for the

org.apache.commons.logging.LogFactory file in the `META-INF/services` directory and then for commons-logging.properties in the classpath.

The value of the System property becomes available to all portlets once it is set, and this would determine the logging implementation class for the portlets on the page. There would be no effect on the logging if the default logging implementation is used by all portlets. However, if the logging implementation class for a portlet is set in a custom common-logging.properties file, it would be ignored. To overcome this, the developer would need to override the method, which sets the System property. See the section Overriding the default logging implementation class for more details.

Commons Logging JAR file

The WebSphere Application Server includes a version of the `commons-logging.jar` file and specifies the default commons logger to be the com.ibm.ws.commons.logging.TrLogFactory class. If the Struts portlet does not specify a log factory, then the default log factory of TrLogFactory is used. However, if the application includes a version of commons-logging, then a ClassCastException results. This release of the Struts Portlet Framework no longer includes `commons-logging.jar`, but instead uses the version that is included in the Application Server. Older applications can continue to use the `commons-logging.jar` file that is packaged with the Struts Portlet Framework if the application specifies a logger, which is the typical case.

Changes to the Struts configuration file

The Struts configuration that is used by Struts portlets references the 1.1 version of the DTD.

```
<!DOCTYPE struts-config PUBLIC
          "-//Apache Software Foundation//DTD Struts Configuration 1.1//EN"
          "http://jakarta.apache.org/struts/dtds/struts-config_1_1.dtd">
```

For the Struts Portlet Framework operate properly, the `RequestProcessor` must be configured. The `RequestProcessor` is responsible for the main function of processing a request. The default `RequestProcessor` can be overridden in the `struts-config.xml` with the controller element. In the `struts-config.xml`, the following portal server `RequestProcessor` must be specified:

Standard portlet container:

```
<controller
   processorClass="com.ibm.portal.struts.portlet.WpRequestProcessor">
</controller>
```

IBM portlet container:

```
<controller
   processorClass="com.ibm.wps.portlets.struts.WpsRequestProcessor">
</controller>
```

You can also extend the Struts RequestProcessor in order to extend the function of Struts. If you place the path to your RequestProcessor in the controller element in the struts-config file, your application does not work properly in the portlet environment. Instead, what you must do is have your RequestProcessor class extend the WpsRequestProcessor rather than the normal Struts RequestProcessor.

The following are the methods overridden in WpsRequestProcessor:

- processLocale
- processActionPerform()

- processActionForm()
- processActionForward()
- processPath()
- processRoles()
- processMapping()
- processValidate()
- doForward()
- doInclude()
- getMapping()

If you override any of the methods listed previously, you must make sure to start the super class version of the method from within your method.

There are existing Struts Extensions, which require you to use their RequestProcessor. Clearly, you cannot use their RequestProcessor and the WpsRequestProcessor at the same time. Instead, you must get the source to that RequestProcessor and following the approach described.

Tiles support

Struts integrates the Tiles package for creating views. A Tiles sample is included with the Struts Portlet Framework that demonstrates how to configure Tiles in the Struts Portlet Framework. Tiles requires a specific Request processor that is configured through a plug-in. The following is from the Struts configuration to show how to configure Tiles for a Struts application that is deployed in WebSphere Portal.

Standard portlet container:

```
<!-- add the Tiles plugin. -->
<plug-in className="com.ibm.portal.struts.plugins.WpTilesPlugin">
   <set-property property="definitions-config"
                 value="/WEB-INF/conf/tiles-def.xml"/>
   <set-property property="definitions-debug" value="2"/>
   <set-property property="definitions-parser-details" value="2"/>
   <set-property property="definitions-parser-validate" value="true"/>
</plug-in>
```

The Tiles plug-in requires that compatible request processor is configured in the controller element. The WpTilesPlugin was written to accept the default Struts RequestProcessor, WpStrutsRequestProcessor, or the WpTilesRequestProcessor.

IBM portlet container:

```
<!-- add the Tiles plugin. -->
<plug-in
   className="com.ibm.wps.portlets.struts.plugins.WpsStrutsTilesPlugin">
   <set-property property="definitions-config"
                 value="/WEB-INF/conf/tiles-def.xml"/>
   <set-property property="definitions-parser-details" value="2"/>
   <set-property property="definitions-parser-validate" value="true"/>
</plug-in>
```

The Tiles plug-in requires that compatible request processor is configured in the controller element. The WpsStrutsTilesPlugin was written to accept the default Struts RequestProcessor, WpsStrutsRequestProcessor, or the WpsStrutsTilesRequestProcessor.

Supporting multiple Struts applications

Learn how web applications that allow packaging more than one Struts-based portlet in a single application can be created either using Struts modules or a namescoped servlet context.

The Jakarta Struts Framework does not support configuring more than one Struts ActionServlet in the web deployment descriptor. This limitation occurs because there is only one servlet context per web application and the commons digester imports the Struts configuration into the servlet context. Struts stores the required information as attributes in the servlet context, and the information is retrieved by using globally defined key names. Struts will suffix these attribute names with the prefix of the Struts modules, so several Struts modules are supported in a single web application, but multiple Struts applications are not.

The Struts Portlet Framework is used to create Struts applications that are deployed in WebSphere Portal Express. This section discusses two techniques for creating web applications that allow packaging more than one Struts-based portlet in a single application. As stated, Jakarta Struts does not support declaring the Struts ActionServlet in the web deployment descriptor more than once. The Struts Portlet Framework has the same limitation because the servlet context is used when the Struts configuration is digested in the portal as well. However, the Struts Portlet Framework supports two ways that an application can be developed in order to implement multiple Struts-based portlets. The first technique uses a single Struts controller while defining a Struts module for each portlet in the application. The second technique uses a servlet context wrapper that namescopes the attributes stored in the servlet context, so each portlet can have its own namespace in the servlet context.

Note: Legacy portlets, such as `SPFLegacyMultipleServletContext.war` referenced here, are available for download from the WebSphere Portal Business Solutions Catalog.

Using Struts modules (IBM portlet container only)

One method of supporting multiple Struts portlets in a single war file is to define a Struts module for each portlet. The welcome file can be specified as a configuration parameter for each concrete portlet. This technique takes advantage of the fact that the Struts Portlet Framework uses the welcome file to set the Struts module. The prefix corresponding to the Struts module for a portlet is stored in the IViewCommand object. The IViewCommand object is retrieved for each portlet and the module prefix is used to select the module, so each portlet in the application will use the Struts module selected during the welcome file processing. Obviously, the portlet can then forward to other modules as well, but initially each portlet can have a unique configuration using the Struts module.

The Struts Portlet Framework controller, WpsStrutsPortlet, is defined as the servlet-class in the web deployment descriptor. The corresponding abstract portlet is then defined in the portlet deployment descriptor. The application can then define each concrete portlet, and specify a welcome file to use. The configuration for one of the concrete portlets is demonstrated as follows:

```
<concrete-portlet-app uid="wp.struts.legacy.examples.MultipleStrutsApp.1">
  <portlet-app-name>Struts Legacy Multiple Modules Application 1</portlet-app-name>
  <concrete-portlet href="#Portlet_1">
    <portlet-name>Struts Legacy Multiple Modules 1</portlet-name>
      <default-locale>en</default-locale>
      <language locale="en">
        <title>Struts Legacy Multiple Modules 1</title>
```

```
        <title-short>Struts Legacy Multiple Modules 1</title-short>
        <description>Struts Legacy Multiple Modules 1</description>
        <keywords>WPS, Struts, Legacy</keywords>
      </language>
      <config-param>
        <param-name>viewMode.page</param-name>
        <param-value>portlet1/welcome.jsp</param-value>
      </config-param>
      <config-param>
        <param-name>editMode.page</param-name>
        <param-value>portlet1/edit.jsp</param-value>
      </config-param>
  </concrete-portlet>
</concrete-portlet-app>
```

The application can configure a separate Struts module that maps to each portlet. This scheme is customizable and will support as many Struts modules as required. However, only one abstract portlet can be defined because there can only be one Struts-based servlet-class. All of the concrete portlets created from the one abstract portlet must share the abstract's portlet configuration. See the SPFLegacyMultipleModules example for more information.

Using a namescoped servlet context

Multiple Struts portlets in a single web application can also be supported by configuring the portlet to use a servlet context wrapper. The (Wps)StrutsPortlet class instantiates the WpsActionServlet object to digest the Struts configuration. The Wp(s)ActionServlet extends the Struts ActionServlet and uses the ActionServlet's `init` implementation to digest the configuration. The (Wps)StrutsPortlet object will invoke the init method of the Wp(s)ActionServlet to digest the configuration. Since the (Wps)StrutsPortlet instantiates the Wp(s)ActionServlet and calls the `init` method, the Struts Portlet Framework has the opportunity to provide a servlet context wrapper. If the application is configured to use the servlet context wrapper, then the Wp(s)ActionServlet will return the servlet context wrapper when the getServletContext method is invoked. The servlet context wrapper defines a namespace unique to the portlet and then stores the attributes in the real servlet context. The concept is similar to how Struts prefixes the attributes in the servlet context for the configuration of each Struts module.

Note: The Wp(s)ActionServlet should be used to obtain the servlet context wrapper when this technique is used. The servlet context obtained from the pageContext, for example, will not be the wrappered servlet context. The Wp(s)ActionServlet can be obtained from the request object using the Globals.ACTION_SERVLET_KEY.

For an IBM portlet container:

The WpsStrutsPortlet class, which is specified as the servlet class in the web deployment descriptor, can be defined multiple times when the NamescopeServletContext initialization parameter, `init-param`, is set to true.

```
<servlet id="wp.struts.legacy.examples.ServletContext.1">
  <servlet-name>StrutsLegacyMultipleServletContexts1</servlet-name>
  <servlet-class>
     com.ibm.wps.portlets.struts.WpsStrutsPortlet
  </servlet-class>
  <init-param>
    <param-name>config</param-name>
    <param-value>
```

```
      /WEB-INF/portlet1/struts-config.xml
    </param-value>
  </init-param>
  <init-param>
    <param-name>struts-servlet-mapping</param-name>
    <param-value>*.do</param-value>
  </init-param>
  <init-param>
    <param-name>NamescopeServletContext</param-name>
    <param-value>true</param-value>
  </init-param>
</servlet>
```

An abstract portlet for each servlet-class can then be defined in the portlet deployment descriptor.

```
<portlet id="Portlet_1"
         href="WEB-INF/web.xml#wp.struts.legacy.examples.ServletContext.1"
         major-version="1" minor-version="0">
  <portlet-name>
     Struts Legacy Multiple Servlet Contexts 1
  </portlet-name>
    <cache>
      <expires>0</expires>
      <shared>NO</shared>
    </cache>
    <allows>
      <maximized />
      <minimized />
    </allows>
    <supports>
      <markup name="html">
        <view />
        <edit />
      </markup>
    </supports>
  </portlet>
```

For a standard portlet container:

The StrutsPortlet class, which is specified as the portlet class in the portlet deployment descriptor, can be defined multiple times when the NamescopeServletContext initialization parameter, `init-param`, is set to true.

```
<portlet>
   <description xml:lang="en">
      Struts Standard Multiple Servlet Contexts 2
   </description>
   <portlet-name>
      Struts Standard Multiple Servlet Contexts 2
   </portlet-name>
   <display-name xml:lang="en">
      Struts Standard Multiple Servlet Contexts 2
   </display-name>
   <portlet-class>
      com.ibm.portal.struts.portlet.StrutsPortlet
   </portlet-class>
   <init-param>
      <name>config</name>
      <value>/WEB-INF/portlet2/struts-config.xml</value>
   </init-param>
   <init-param>
      <name>struts-servlet-mapping</name>
      <value>*.do</value>
   </init-param>
   <init-param>
      <name>NamescopeServletContext</name>
```

```
      <value>true</value>
   </init-param>
   ...
</portlet>
```

Each portlet is independent of the other Struts-based portlets. See the SPFLegacyMultipleServletContexts.war example for more information.

Create link tags in Struts

Get an overview of how to write tags to create links in a JSP in both a servlet and a portlet in Struts framework.

Tags that create links need to create links that are serviced by the portlet. The URL that is created because of the Struts processing needs to be passed as a parameter back to the portlet. There must be a common tag for both a servlet and a portlet. The following code sample demonstrates how to write a tag that can be used by a JSP in both a servlet and a portlet.

```
PortletApiUtils portletUtils = PortletApiUtils.getInstance();
if (portletUtils != null)
{
   Object pResponse = portletUtils.getPortletResponse((HttpServletRequest)
                                                    pageContext.getRequest());
   Object portletURI =
      portletUtils.createPortletURIWithStrutsURL(pResponse,
                   calculateURL());

   results.append(portletURI.toString() );
}
else
{
   // servlet environment
   results.append(calculateURL());
}
```

The else clause was the original statement for the servlet-only environment. An instance of PortletAPIUtils is initially obtained. In a non-portlet environment, this call would return null. The initialization of the Struts Portlet Framework support would set the instance of the PortletAPIUtils implementation. A PortletResponse object is obtained as an Object. The PortletURI is then created with a DefaultPortletAction and a parameter that contains the Struts path through the call to `createPortletURIWithStrutsURL`.

Creating links in a JSP

The preferred method of creating a link in a JSP is to use a tag. This method is the most convenient method to assure that the link is portal-aware and takes Struts Modules into account. The Struts module implementation requires that the link is prefixed with the ModuleConfig's prefix. A typical Struts tag implementation would include logic similar to the following sample.

```
ModuleConfig config =
  (ModuleConfig) pageContext.getRequest().getAttribute(Globals.MODULE_KEY);
if (config != null) {
  value.append(config.getPrefix());
}
```

There are times when a link needs to be created in a JSP by using Java code. The links that are created in a JSP must also prefix the `ModuleConfig` prefix so the links support Struts Modules. PortletApiUtils has a convenience method that can be used to prefix the URL that uses the `ModuleConfig`. The method is called `addModulePrefix`. The following snippet demonstrates creating a link in a JSP.

```
<%@ page language="java" import="com.ibm.wps.struts.common.PortletApiUtils"
%>

<%
  PortletApiUtils portletUtils = PortletApiUtils.getInstance();
%>

<%
    if (portletUtils != null)
   {
      String url = "/bean-write.jsp";
      // add the module prefix to the url
      url = portletUtils.addModulePrefix(url, request);
      Object portletURI = portletUtils.createPortletURIWithStrutsURL(request, url);
%>
  <a href="<%=portletURI.toString()%>"><bean:write></a>
<%
   }
%>
```

Related information

- The Struts Application Framework
- Portlet API
- Portlet development basics

Formatting XML documents with XSLT

Learn how a Struts application supports applying a style sheet to XML data.

The typical way that a Struts application supports applying a style sheet to XML data is to implement a Struts Action that writes directly to the response object. The IStrutsPrepareRender interface allows implementing a Struts action that will be executed in the render phase of portal, and therefore will have a response object that can be written to. The SPFLegacyTransformation.war is an example of how to implement a Struts action in the Struts Portlet Framework that can apply a style sheet to an XML document.

Note: You can locate legacy portlets, such as SPFLegacyTransformation.war, from the IBM WebSphere Portal Express Catalog.

Related information

- The Struts Application Framework
- Portlet API
- Portlet development basics

Static content in Struts

Portlet JSPs can link to the static content within the portlet's WAR directory structure by using the services of the portlet container to create portlet URIs from which the content can be referenced.

Portlet JSPs cannot link directly to static content within the portlet's WAR directory structure. Instead, they have to use the services of the portlet container to create portlet URIs from which the content can be referenced. Links within static pages (for example, HTML pages) also need to be converted into portlet URIs. The Struts Portlet Framework processes links to static content and converts them to the form of portlet URIs so that linked content is brought back into the portlet and displayed.

Creating a Home button for static content

When the Struts Portlet Framework is used to display static content, it can be difficult to navigate back through a number of screens to a location earlier in the browser history. To overcome this, you can add a `homeFromStatic` forward to your action mapping which references the static content. This causes a home button to be added to your static content. Pressing the home button causes the link specified by the `homeFromStatic` forward to be used.

As with any action mapping, it is possible also to use a global forward. Thus, if for all sequences of static pages it was required to use the same link for every home button, one could use a global forward:

```
<global-forwards>
  <forward name="homeFromStatic"     path="/index.jsp"/>
   ...
</global-forwards>
```

A `homeForStatic` forward can also be created for a specific ActionMapping:

```
<action path="/staticpage"
        type="org.apache.struts.actions.ForwardAction"
   parameter="/html/view/static.html">
   <forward   name="homeFromStatic"       path="/index.jsp"/>
</action>
```

The home button is not only placed on the first static content page, but on all pages generated by links resulting from that page. If no `homeFromStatic` forward is found with the action mapping which initiated the sequence of static content pages, and there is no global `homeFromStatic` forward either, the home button is not generated.

Migrating existing Struts applications

Get an overview of how the existing Struts applications can be migrated using the Struts Portlet Framework so that these applications can be deployed in IBM WebSphere Portal Express.

The Jakarta Struts distribution ships an example Struts application that demonstrates the features of Struts, including implementing Struts Actions, creating custom tags, using the Struts tag libraries, authoring JSPs, and form validation. The Struts Portlet Framework demo builds on this example by incorporating WebSphere Portal Express features, including support for help mode and WML markup. The Struts example required very few changes to enable it for the Struts Portlet Framework.

The file structure of the WAR file for a Struts Portlet Framework is the same as that used for a Struts servlet. The Struts application in the portal server is a WAR file just like it is in the servlet environment. The portlet WAR file has some additional JARS and other requirements, but the basis of the implementation is similar to the servlet application. See Creating a simple portlet for more information.

Migrating servlet-based Struts applications

Existing Struts applications can be migrated using the Struts Portlet Framework so the application can be deployed in WebSphere Portal. Since Struts is a framework there are many variations to how the application can be built with Struts. There are also numerous other frameworks that may also be incorporated into the Struts

application. The steps in this section can be used as a starting point for the migration effort, but may not cover all of the issues that can be encountered.

1. Make sure the existing Struts application has been built as a Struts 1.1 application. (See http://jakarta.apache.org/struts/.)
2. Check Struts Actions to see if the action writes directly to the response object. If it does then, the action must be modified to either return an ActionForward instead, or check to see if the IStrutsPrepareRender interface should be used.
3. The Web deployment descriptor must be updated to use the WpsStrutsPortlet as the servlet class instead of the ActionServlet. (See "Changes to configuration files" on page 2958.)
4. The servlet mapping for Struts actions must be specified as the struts-servlet-mapping init parameter. (See "Changes to configuration files" on page 2958.)
5. Create a `portlet.xml`. The `SPFLegacyMailReader.war` can be used as an example. (See "Deployment descriptors" on page 3129.)
6. Modify the `struts-config.xml` to specify the WpsRequestProcessor as the controller. (See "Changes to configuration files" on page 2958.)
7. Modify tags that use pageContext.forward() to use the PortletApiUtils forward. (See "Changes to Struts application code" on page 2945.)
8. Modify JSPs that use a forward to use the logic forward tag. (See "Changes to Struts JSPs" on page 2955.)
9. Modify JSPs as necessary to use the Struts tags for creating URLs, like the Struts Link and Form tags. (See "Changes to Struts JSPs" on page 2955).
10. The JAR files from the `WEB_INF/lib` directory of the `SPFLegacyBlank.war` must be used. These files are the Struts JARs and the required Struts Portlet Framework JARs.
11. The TLD files must be updated from the `WEB_INF/lib` directory of `SPFLegacyBlank.war`. Verify the taglib attributes and that the JSP correctly reference the TLD files. This has been a common source of problems when migrating existing applications.
12. The JSPs should be modified so they do not use html, head and body elements. All HTML output to the portal is written in the context of an HTML table cell.

Migrating servlet-based Struts applications to the Standard version

Existing Struts applications can be migrated using the Struts Portlet Framework so the application can be deployed in WebSphere Portal. Since Struts is a framework there are many variations to how the application can be built with Struts. There are also numerous other frameworks that may also be incorporated into the Struts application. The steps in this section can be used as a starting point for the migration effort, but may not cover all of the issues that can be encountered.

1. Make sure the existing Struts application has been built as a Struts 1.1 application. (See http://jakarta.apache.org/struts/.)
2. Check Struts Actions to see if the action writes directly to the response object. If it does then, the action must be modified to either return an ActionForward instead, or check to see if the IStrutsPrepareRender interface should be used.
3. The Web deployment descriptor must be updated to use the StrutsPortlet as the portlet class. Remove the servlet-class from the Web deployment descriptor, and specify the StrutsPortlet as the portlet class in the portlet deployment descriptor. (See Changes to the Web deployment descriptor.)

4. The servlet mapping for Struts actions must be specified as the struts-servlet-mapping init parameter. The portlets init parameters are specified in the portlet deployment descriptor.
5. Create a `portlet.xml`. The SPFStandardMailReader.war can be used as an example. (See Portlet deployment descriptor.)
6. Modify the `struts-config.xml` to specify the WpRequestProcessor as the controller. (See "Changes to the Struts configuration file" on page 2970.)
7. Modify tags that use pageContext.forward() to use the PortletApiUtils forward. (See "Forwards and redirects" on page 2949.) The following example illustrates the change in which the PortletApiUtils object is obtained:
   - Old: PortletApiUtils portletUtils = PortletApiUtils.getInstance();
   - New: PortletApiUtils portletUtils = PortletApiUtils.getUtilsInstance();
8. Modify JSPs that use a forward to use the logic forward tag.
9. Modify JSPs as necessary to use the Struts tags for creating URLs, like the Struts Link and Form tags. (See "Creating portlet URIs" on page 2955).
10. The JAR files from the `WEB_INF/lib` directory of the SPFStandardBlank.war must be used. These files are the Struts JARs and the required Struts Portlet Framework JARs.
11. The TLD files must be updated from the `WEB_INF/lib` directory of SPFStandardBlank.war. Verify the taglib attributes and that the JSP correctly reference the TLD files. This has been a common source of problems when migrating existing applications.
12. The JSPs should be modified so they do not use html, head and body elements. All HTML output to the portal is written in the context of an HTML table cell.

Web 2.0 user interface features

Learn about portal features that pertain to the Web 2.0 generation type of Web user interface.

Web technology has evolved towards a different direction. In the public this evolution has been named Web 2.0. This term does not describe a new type of technology, but has been used in a broad manner to describe a change to a more user centered focus. Among the benefits are improved customer and service orientation, increased user activities, easier communication and collaboration, better usability, faster performance, etc.

Within portal, the Web 2.0 features are implemented as follows:

- Client side portlet programming model: You can use the client side programming model for your portlets. You can do everything with the client side programming model that you can do with the server side portlet programming model. Additionally, the client side programming model has the following advantages:
  - Improved user experience by faster responses and performance, as many interactions are processed on the client side rather than on the server.
  - User customization of user profile, preferences, and changes to the portlet state are done locally, and therefore with a faster response time. A fragment that contains the customization is later sent to the server and saved.
  - The user experience is consistent between both client side aggregation and server side aggregation.
- Live text: You can use live text. Live text provides elements embedded in portal pages that become active in the browser and are enhanced with additional

functionality by JavaScript libraries. For example, if you include portal user IDs in your portlet output and mark them as live text, users can click these IDs in the browser and see a person info card or a menu that allows them to send a mail to the person. Live text has the following advantages:
  - Live text allows easier click to action.
  - You can adopt new portal content within your company more easily, as it is now easier to handle portal tags. For example, you can write tags and make them available centrally, and UI developers can reuse these tags for in their portlets for various purposes.
  - Content editors can add meaningful live text elements to portlets without requiring portlet development knowledge.
  - You can embed content from other sources, for example, from a HTTP or .NET server.
- REST services:
  - Portal offers many REST services, such as Layout model, Portlet model, Content model, Navigation model, Wire model, and User profile.
  - By the use of REST services you can write your own advanced Web application and build it with REST (Representational State Transfer) services that provide the XML request information.
  - REST services allow you to access portal models remotely for both read and write access. The Navigation model allows read access only; it is updated by changes made to the content model.
- Controller SPI: The Controller SPI is a public portal interface. It is not directly related to the Web 2.0 type of user experience, but it allows you to perform certain administrative tasks more easily.

"The client side portlet programming model"
You can use the client side programming model to make use of AJAX techniques in your standard API portlets.

"Outbound HTTP connection" on page 2984
Applications in your IBM WebSphere Portal Express and the related user activities can require outbound HTTP connections to remote computer systems. The outbound HTTP connection service provides an administration infrastructure with a central point of administration for all outbound HTTP connections that are defined in the portal environment.

"Live text for click-to-action" on page 3062
IBM WebSphere Portal Express supports a live text API for user controlled data transfer between components. With live text, a component on a page can declare sources and targets for data transfer. For example, this can be a portlet or a navigation element. When the user clicks on a source element, the portal displays a menu listing targets that match the selected source. When a menu item is selected, the portal invokes the corresponding target passing it the source data. This process is called Click-to-Action (C2A).

The client side portlet programming model

You can use the client side programming model to make use of AJAX techniques in your standard API portlets.

You can use the client side programming model for writing your portlets. You can do everything with the client side programming model that you can do with the server side portlet programming model. Additionally, the client side programming model has the following advantages:

- The client side portlet programming model works in both client side aggregation (CSA) and server side aggregation (SSA).

- Improved user experience by faster response and performance. Portlets that use the clients side programming model render faster, as the portal does not re-render the whole page, but only the aspects of the portlet that change.
- The client side programming model allows you to handle changes of the portlet mode and window state, preferences, and user customization of user profile locally. This provides a faster response time for the user. A fragment that contains the customization is later sent to the server and saved.
- Writing portlets to the client side programming model does not require deep Java knowledge. You can write such portlets as HTML code with css style sheets and JavaScript. They have few or no JSPs. Example scenarios for writing client side portlets are:
  - A user can add some markup to the portlet view by selecting options in a form.
  - Preference changes that a user makes to the portlet are applied immediately in the browser view without reloading the whole portlet. The preference change is later sent to the server and saved.
  - Access to the user profile on the client side.
  - Portlet mode changes are performed entirely on the client. Performance improves as no server-client communication is required.
  - The portlet can retrieve markup fragments from the server. The user does not notice this, as the ATOM feed format is hidden by the XMLPortletRequest implementation, similar to a XMLHttpRequest (XHR).
  - XSLT and XPath helper functions make it easier to handle XML feeds.

For the full description of the JavaScript interfaces refer to the Web 2.0 Javadoc documentation available on developerWorks.

"Getting started with the client-side programming model for portlets"
Getting started with the client-side programming Model requires a few updates to the portlet JSPs.

"Handling portlet preferences on the client" on page 2982
One of the most useful aspects of the client side programming model is the ability to read, modify, and save portlet preferences on the client.

"Changing portlet mode and window state on the client side" on page 2983
With the client-side programming model, you can handle portlet mode and window state changes entirely on the client, rather than requiring a full server-client roundtrip.

"JavaScript namespacing - observing good practice" on page 2984
The client side context of Web programming requires good namespacing.

Getting started with the client-side programming model for portlets:

Getting started with the client-side programming Model requires a few updates to the portlet JSPs.

Procedure

1. Add a reference for the client-side programming model tag library:
```
<%@ taglib uri="http://www.ibm.com/xmlns/prod/websphere/portal/v6.1/portlet-client-model"
    prefix="portlet-client-model" %>
```
2. Declare the modules of the client-side programming model that you want to use:
```
<portlet-client-model:init>
     <portlet-client-model:require module="ibm.portal.xml.*"/>
     <portlet-client-model:require module="ibm.portal.portlet.*"/>
</portlet-client-model:init>
```

For a complete list of modules for use with the client-side programming model refer to the public API documentation.

3. To be able to work with the client-side programming model in your portlets, use the object `ibm.portal.portlet.PortletWindow` as an entry point:

```
<script>
   dojo.addOnLoad(function() {
      <%=namespace%>_portletWindow = new ibm.portal.portlet.PortletWindow("<%=portletWindowID%>");
   });
</script>
```

Note: Use the tag `portlet-client-model:init` to create the scripting variable `portletWindowID` for use in the JSP. The following example shows how you can start an `XMLPortletRequest`:

```
<script>

function sendXPR( /*string*/url, /*Function*/callback, /*String*/errMsg ) {
   var xpr = <%=namespace%>_portletWindow.newXMLPortletRequest();
   var me = this;
   xpr.onreadystatechange = function () {
      if ( xpr.readyState == 4 ) {
         if ( xpr.status == 200 ) {
            callback( xpr.responseText );
         }
      else {
         <%=namespace%>_portletWindow.logError("The request failed!", errMsg );

         }
      }
   };
   xpr.open( "GET", url );
   xpr.send( null );
}
</script>
```

Handling portlet preferences on the client:

One of the most useful aspects of the client side programming model is the ability to read, modify, and save portlet preferences on the client.

About this task

The interface `ibm.portal.portlet.PortletPreferences` JavaScript mirrors its server side counterpart `javax.portlet.PortletPreferences` as closely as possible. For the full description of the JavaScript interfaces refer to the public API documentation.

The following code sample shows how you can retrieve and read the portlet preferences on the client:

```
<script>
    function <%=namespace%>_handleLoadPortletPreferences(portletWindow, status, portletPrefs) {
        if (status==ibm.portal.portlet.PortletWindow.STATUS_OK) {
            portletWindow.setAttribute("preferences", portletPrefs);
            alert("Preferences.getValue()\n"+portletPrefs.getValue("test"));
            var prefs = portletPrefs.getMap();
            var mapStr = "Preferences.getMap()\number of preferences: "+prefs.length + "\n";
            for (var i=0; i<prefs.length; i++) {
                mapStr += i+" - "+prefs[i].name+" - "+prefs[i].values+" - "+prefs[i].readonly + "\n";
            }
            alert(mapStr);
        }
        else { alert("error loading feed"); }
    }
    <%=namespace%>_portletWindow = new ibm.portal.portlet.PortletWindow("<%=portletWindowID%>");
    <%=namespace%>_portletWindow.getPortletPreferences(<%=namespace%>_handleLoadPortletPreferences);
</script>
```

For the full description of the JavaScript interfaces refer to the public API documentation.

Related information:

http://www-10.lotus.com/ldd/portalwiki.nsf/xpViewCategories.xsp?lookupName=IBM%20WebSphere%20Portal%207%20API%20and%20SPI%20Reference

Changing portlet mode and window state on the client side:

With the client-side programming model, you can handle portlet mode and window state changes entirely on the client, rather than requiring a full server-client roundtrip.

About this task

In client-side aggregation, you can provide an event handler for changes of portlet mode and portlet window state. This handler gets called when a mode change or a window state change is triggered.

Note: This function is only supported in the portal CSA theme and the CSA skin. You can adapt the CSA theme and skin to write your own custom themes and skins to support this feature.
The return value of your handler determines whether the default action is run:

- A return value of `true` allows execution of the default action, in this case the portlet mode or window state change.
- A return value of `false` cancels the default action.

This action allows the portlet to handle these changes entirely on the client, with no server interaction.

You define the change handlers as follows:

- To define a portlet mode change handler, define a JavaScript function with the name `<portlet:namespace/>doPortletMode` in your portlet.
- To define a portlet window state change handler, define a function with the name `<portlet:namespace/>doWindowState`.

The CSA skin checks whether such a function is defined before it processes any portlet mode or window state changes. The handler function provides two arguments:

- The first argument is the required portlet mode or window state in string form.
- The second argument is the markup element that must be used to insert a mode-specific markup. This argument is necessary to support the inline modes that you see in the CSA version of the IBM portal skin.

Refer to the following example:

```
<portlet:namespace/>doPortletMode( portletMode, div ){
   var retVal = true;
   if ( portletMode == ibm.portal.portlet.PortletMode.VIEW ) {
      //do view mode here
      div.innerHTML = "...some view mode markup...";
      retVal = false;
   }
   return retVal;
   }
```

JavaScript namespacing - observing good practice:

The client side context of Web programming requires good namespacing.

About this task

When using the client side programming model for your portlets you need to apply good namespacing practices in JavaScript. This helps to avoid collisions in the page which can be timely and expensive to track down. The namespace need must be balanced with the performance advantage of moving all JavaScript into external JavaScript files that can be cached. An additional benefit is that this makes portlet JSPs much smaller. Refer to the following code sample of writing JavaScript for a portlet by the client side programming model:

`htmlEditor.js:`

```
if ( typeof( HTMLEditorController ) == "undefined" ) {
     var HTMLEditorController = function( namespace ) {
          this._namespace = namespace;
          this.getSaveForm = function () {
               return document.forms[ namespace + "saveForm" ]
          }
     }
}
```

`htmlEditor.jsp:`

```
var <%=namespace%>htmlController = new HTMLEditorController( "<%=namespace%>" );
```

Outbound HTTP connection

Applications in your IBM WebSphere Portal Express and the related user activities can require outbound HTTP connections to remote computer systems. The outbound HTTP connection service provides an administration infrastructure with a central point of administration for all outbound HTTP connections that are defined in the portal environment.

Advantages:

- The outbound HTTP connection service gives portal system administrators central control over outbound HTTP connections that are used.
- The outbound HTTP connection infrastructure provides functions for authentication and cookie handling.
- Application developers do not have to implement common HTTP connections support. For example, they do not need to write the handler code for various types of HTTP-based authentication, such as the following types:
  - Basic authentication
  - Form based authentication
  - Authentication through SSL tokens
  - SAML authentication.
- In WebSphere Portal Express Version 8.0 and earlier versions, outbound HTTP connections were accessible through the Ajax Proxy service. The Ajax Proxy service was configured by a configuration document named `proxy-config.xml`. You find this document in the `/WEB-INF` directory of the web module that uses the Ajax Proxy service. Starting with WebSphere Portal Express Version 8.5 and the new outbound connection service, the configuration of outbound HTTP connections is now part of the standard datastore-based portal configuration.

Outbound HTTP connections are created by application code to establish an HTTP connection to a remote system. For example, such application code can be part of portlets, portlet services, themes, servlet filters, or other items. A remote outbound HTTP connections can be called from different components. Examples:

- Code that runs in the context of a portlet
- Code that runs in the context of a servlet, for example, a servlet filter
- Code that runs on the portal server outside the context of a servlet request
- Code that runs on the client.

You can therefore establish outbound HTTP connections in the following different ways:

- Through the portlet-request based outbound HTTP connection service
- Through the servlet-request based outbound HTTP connection service
- Through the Ajax proxy.

The WebSphere Portal Express outbound HTTP connection includes the following components:

- A common outbound HTTP connection configuration infrastructure. This infrastructure provides base functions for managing the configuration of outbound HTTP connections.
- The outbound HTTP connection service. This service makes it possible to establish outbound HTTP connections from code that runs in the portal context.
- The HTTP proxy for Ajax application. This application provides an interface for accessing outbound HTTP connections by using an HTTP proxy channel.

The Ajax proxy and the outbound HTTP connection services use the same common configuration infrastructure.

You can use an outbound HTTP connection in the following two ways:

- Through the outbound HTTP connection service
- Through the Ajax Proxy, which is also known as the HTTP proxy for Ajax application.

The following image illustrates the two different ways to connect to a remote system by using outbound HTTP connections:

"The programming model for the outbound HTTP connection service" on page 2986
The outbound HTTP connection service can be used from the context of a servlet request service or from the context of a portlet request service. Here are some code examples.

"HTTP proxy for Ajax applications, also known as Ajax Proxy" on page 2989
One of the basic technologies that emerged in the context of the next generation web user interface is *Ajax* (Asynchronous JavaScript and XML). Using Ajax can increase the responsiveness and usability of your web applications.

"Configuring outbound HTTP connections" on page 2992
In WebSphere Portal Express Version 8.0 and earlier versions, outbound HTTP connections were accessible through the Ajax Proxy service. The Ajax Proxy service was configured by a configuration document named `proxy-config.xml`. You find this document in the `/WEB-INF` directory of the web module that uses the Ajax Proxy service. Starting with WebSphere Portal Express Version 8.5 and the new outbound connection service, the configuration of outbound HTTP connections is now part of the standard datastore-based portal configuration.

"Authenticating outbound HTTP connections" on page 3031
You can protect the access to the remote host by an authentication mechanism.

"Using dynamic elements in outbound HTTP connection settings" on page 3053
In some cases, it is useful to control configuration settings at run time. For example, an administrator might want to have the program decide at run time which policy rule is applied. In another scenario, parts of a policy rule configuration that are known only at run time must be included in the configuration.

"Using programmatic extensions for outbound HTTP connections" on page 3054
To extend the functions of an outbound connection, portal administrators can implement a custom outbound service filter.

"Migration" on page 3061
WebSphere Portal Express Version 8.5 provides a migration process for the change from the Ajax proxy of previous portal versions to the new outbound HTTP connection.

The programming model for the outbound HTTP connection service:

The outbound HTTP connection service can be used from the context of a servlet request service or from the context of a portlet request service. Here are some code examples.

The following example shows how to obtain the outbound HTTP connection service from the context of a portlet:

```
import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.portlet.PortletRequest;
import javax.portlet.PortletResponse;
import com.ibm.portal.outbound.service.OutboundConnectionService;
import com.ibm.portal.outbound.service.OutboundConnectionServiceHome;
import com.ibm.portal.outbound.service.OutboundConnectionServiceException;

// obtain an Outbound HTTP connection service object (portlet context)

private OutboundConnectionService getService (PortletRequest p_request,
           PortletResponse p_response)
 throws  OutboundConnectionServiceException, NamingException
{

 Context ctx = new InitialContext();
 final OutboundConnectionServiceHome home = (OutboundConnectionServiceHome)
           ctx.lookup(OutboundConnectionServiceHome.JNDI_NAME);
      final OutboundConnectionService service =
          home.getOutboundConnectionService(p_request, p_response);
 return service;
}
```

In a different scenario, the code that calls the outbound HTTP connection service is part of a servlet. In this case, the method `getOutboundConnectionService ()` receives the servlet request and servlet response variable. See the following example:

```
import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import com.ibm.portal.outbound.service.OutboundConnectionService;
import com.ibm.portal.outbound.service.OutboundConnectionServiceHome;
import com.ibm.portal.outbound.service.OutboundConnectionServiceException;
...
```

```
// obtain an Outbound HTTP connection service object  (servlet context)
private OutboundConnectionService getService ( HttpServletRequest s_request,
       HttpServletResponse s_response)
 throws OutboundConnectionServiceException, NamingException
{

 Context ctx = new InitialContext();
 final OutboundConnectionServiceHome home = (OutboundConnectionServiceHome)
           ctx.lookup(OutboundConnectionServiceHome.JNDI_NAME);
      final OutboundConnectionService service =
          home.getOutboundConnectionService(p_request, p_response);
 return service;
}
```

Applications use the `createConnection()` method of the outbound HTTP connection service to open an outbound HTTP connection. The following code snippet connects to a URL resource through outbound HTTP connections and reads the content of the web page and writes it to a byte array output stream. See the following example:

```
import java.io.IOException;
import java.io.InputStream;
import java.io.ByteArrayOutputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import com.ibm.portal.outbound.service.OutboundConnectionService;

...
/**
 * usage sample for GET requests using Outbound HTTP Connections
 * @param service the outbound HTTP connection service
 * @param theURL the remote URL
 * @param bos An object that receives the content of the GET request.
 * @return the HTTP status
 **/
private int doGet (OutboundConnectionService service,
                   URL theURL,
                   ByteArrayOutputStream bos)
    throws OutboundConnectionServiceException, IOException
{
    // obtain a connection object
    HttpURLConnection connection = createConnection(theURL);
    try {
       // Submit the URL connection to the remote host.
       connection.connect();

       // read the returned content:
       InputStream is = (InputStream)connection.getContent();
       int byt = is.read();
       while (byt >= 0) {
          bos.write(byt);
          byt = is.read();
       }
  int status = connection.getStatus();
       return status;
    } finally {
       connection.disconnect();
       bos.close();
    }
}
```

The following code snippet connects to a URL resource through outbound HTTP connections and submits a POST request:

```
import java.io.IOException;
import java.io.InputStream;
import java.io.ByteArrayOutputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import com.ibm.portal.outbound.service.OutboundConnectionService;

...
/**
 * usage sample for POST requests using Outbound HTTP Connections
 * @param service the outbound HTTP connection service
 * @param theURL the remote URL
 * @param postData the POST data.
 * @param bos An object that receives the content of the GET request.
 * @return the HTTP status
 **/
private int doPost (OutboundConnectionService service,
                    URL theURL,
                    byte[] postData,
                    ByteArrayOutputStream bos)
    throws OutboundConnectionServiceException, IOException
{
    // obtain a connection object
    HttpURLConnection connection = createConnection(theURL);
    try {
       // write the POST data
       OutputStream os = connection.getOutputStream();
       os.write(postData, 0, postData.length);
       os.close();

       // Submit the URL connection to the remote host.
       connection.connect();

       // read the returned content:
       InputStream is = (InputStream)connection.getContent();
       int byt = is.read();
       while (byt >= 0) {
          bos.write(byt);
          byt = is.read();
       }
  int status = connection.getStatus();
       return status;
    } finally {
       connection.disconnect();
       bos.close();
    }
}
```

The following code snippet obtains an outbound connection service and requests the content of the URL www.ibm.com:

```
ByteArrayOutputStrream bos = new ByteArrayOutputStream();

OutboundConnectionService ocs = getService (request, response);
int status = doGet(ocs, new URL("http.//www.ibm.com"), bos);
if (status >= 400) {
    System.err.println("Remote connection failed with HTTP status "+status);
} else {
    byte[] da = bos.toByteArray();
    String first = da.length > 10 ? new String(da,0,da.length) : new String(da);
    System.out.println("The content starts with "+first);
}
```

HTTP proxy for Ajax applications, also known as Ajax Proxy:

One of the basic technologies that emerged in the context of the next generation web user interface is *Ajax* (Asynchronous JavaScript and XML). Using Ajax can increase the responsiveness and usability of your web applications.

Ajax enables web pages to load data or markup fragments from a server by using asynchronous requests. These requests are processed in the background. Therefore, they do not interfere with the web page that is displayed in the browser. When you use Ajax, your web application exchanges only small amounts of data between the server and the client. Therefore, it refreshes only small parts of the markup. Therefore, Ajax is also useful for developing portlets and for building mashups, that is, aggregating content from various different sources into a uniform user experience. For example, such content can consist of RSS or Atom feeds or other data that is retrieved from external REST services.

To prevent cross-site scripting attacks in such web applications, browsers introduced the so called same-origin policy. This policy prevents client side scripts, in particular JavaScript, from loading content from an origin that has a different protocol, domain name, or port. To overcome this restriction, some browser vendors offer solutions that are based on signed scripts. However, using a signed script does not mean that a script can be trusted. Another disadvantage of these browser-specific solutions is that they rely on the user to configure the browser accordingly.

The solution that IBM WebSphere Portal Express offers is based on a server-side HTTP proxy. This proxy is named the HTTP Proxy for Ajax Applications, or also known as the Ajax Proxy. The underlying security model allows administrators to restrict access to trusted origins in a flexible way. The Ajax Proxy can be used for developing themes, skins, static pages, or portlets. The following sections explain how to use and configure the Ajax proxy.

"The programming model for using the AJAX proxy"
View information on using the proxy when programming portlets.

The programming model for using the AJAX proxy:

View information on using the proxy when programming portlets.

The programming model specifies the following:

- How you make requests, that is how you format URLs that address the proxy
- How the proxy reacts in specific cases, for example in case of an error.

Note: The proxy does not rewrite URLs that are part of the received content, because the proxy does not know the semantics of the content that it fetches. Consequently, AJAX applications that use the proxy need to generate proxy URLs by themselves.

"URL format and examples" on page 2990
The proxy can be addressed via URLs that comply with the REST-based URL format given here.

"AJAX proxy status codes" on page 2990
The AJAX proxy responds to requests by defined status codes. Possible errors that can occur in the proxy are mapped to the corresponding HTTP status codes.

"Using the AJAX proxy in portlets" on page 2991
Here is how you use the AJAX proxy in portlets.

URL format and examples:

The proxy can be addressed via URLs that comply with the REST-based URL format given here.

This format is as follows:

```
<protocol>://<portal-domain>/<proxyurl>/<protocol>/<host>%3a<port>/<url>?<query>
```

The proxy URL carries the target URL in its path information part. Depending on the context path mapping in the proxy configuration, the target URL can be an absolute or relative URL.

Examples: Refer to the following sample proxy URLs. They show how you can load a URL via the proxy depending on the context path mapping in the proxy configuration.

```
URL: http://www.ibm.com/developerworks/news/dw_dwtp.rss
Context path mapping (from proxy-config.xml): <mapping contextpath="/proxy" url="*"/>
Proxy URL: http://myportal.com:10040/wps/proxy/http/www.ibm.com/developerworks/news/dw_dwtp.rss

URL: http://myotherportal.com:1234/sitemap
Context path mapping (from proxy-config.xml): <mapping contextpath="/proxy" url="*"/>
Proxy URL: http://myportal.com:10040/wps/proxy/http/myotherportal.com%3a1234/sitemap

URL: http://www-01.ibm.com/software/swnews/swnews.nsf/swnewsrss?openview&RestrictToCategory=lotus
Context path mapping (from proxy-config.xml): <mapping contextpath="/ibmsoftwarenews"
        url="http://www-01.ibm.com/software/swnews/"/>
Proxy URL: http://myportal.com:10040/wps/ibmsoftwarenews/swnews.nsf/swnewsrss?openview&RestrictToCategory=lotus
```

AJAX proxy status codes:

The AJAX proxy responds to requests by defined status codes. Possible errors that can occur in the proxy are mapped to the corresponding HTTP status codes.

By default, the AJAX proxy does not forward an HTTP status code that it receives from the target host. Only 2xx and 3xx status codes are directly forwarded to the client. If the proxy receives an 4xx error code, it will always return a 404 'Not Found' error instead. You can change this behavior by using the `forward-http-errors` configuration parameter. For more details about this parameter refer to the topic about General configuration parameters.

The proxy returns its response with one of the following status codes:

200 OK
: This code indicates that the request was accepted by the proxy and that the target server returned a response with a status code 200. This means that the request complied with one of the access policies that you specified in the proxy configuration.

400 Bad Request
: The proxy returns this code if the syntax of the request was incorrect.

403 Forbidden
: The proxy returns this code in either of the following two cases:
  - The request was not accepted by the proxy, that is the proxy found no matching access policy that grants access to the target server.
  - Basic authentication failed.

404 Not found
: This code indicates that the proxy accepted the request, but the target server either returned status code 404 itself or a different 4xx error code.

302 Found
This code indicates that the authentication was successful.

Using the AJAX proxy in portlets:

Here is how you use the AJAX proxy in portlets.

The easiest way to create proxy URLs in a portlet is to register the AJAX proxy servlet in the `web.xml` file of the portlet. This allows you to create the base proxy URL by using the portlet API, whereas you append the target URL according to the URL format specified in the section about the URL format. The class name of the proxy servlet is `com.ibm.wps.proxy.servlet.ProxyServlet`. For more details refer to the sample `web.xml` file.

If you want the proxy to be able to access resources that require authentication, specify a second servlet mapping that is associated with a security constraint. In the sample, only authenticated users can access proxy URLs that match the URL pattern `/myproxy/*`.

Note: You must associate the user roles that you specify in the `web.xml` file with the user roles of the portal server. You can do this by creating the corresponding role mappings for the respective application in the WebSphere Integrated Solutions Console.

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://java.sun.com/xml/ns/j2ee" ...>
   ...
   <servlet>
      <servlet-name>ProxyServlet</servlet-name>
      <servlet-class>com.ibm.wps.proxy.servlet.ProxyServlet</servlet-class>
   </servlet>
   ...
   <servlet-mapping>
      <servlet-name>ProxyServlet</servlet-name>
      <url-pattern>/proxy/*</url-pattern>
   </servlet-mapping>
   <servlet-mapping>
      <servlet-name>ProxyServlet</servlet-name>
      <url-pattern>/myproxy/*</url-pattern>
   </servlet-mapping>
   ...
   <security-constraint id="SecurityConstraint_1">
      <web-resource-collection id="WebResourceCollection_1">
         <web-resource-name/>
         <url-pattern>/myproxy/*</url-pattern>
         <http-method>GET</http-method>
         <http-method>POST</http-method>
         <http-method>PUT</http-method>
         <http-method>HEAD</http-method>
      </web-resource-collection>
      <auth-constraint id="AuthConstraint_1">
         <description>only for authenticated</description>
         <role-name>All Role</role-name>
      </auth-constraint>
   </security-constraint>
   ...
   <security-role id="SecurityRole_1">
      <description>Everyone</description>
      <role-name>Everyone Role</role-name>
   </security-role>
   <security-role id="SecurityRole_2">
```

```
      <description>All authenticated users</description>
      <role-name>All Role</role-name>
   </security-role>
</web-app>
```

Registering the proxy servlet in the Web deployment descriptor of a portlet does not imply that the portlet is based on an application specific configuration. If no `proxy-config.xml` file is provided with the portlet, the proxy servlet uses the global proxy configuration instead. The only constraint that you need to consider is that for each servlet mapping, a corresponding context path mapping must exist in the proxy configuration. This can be either in the global or in the application specific configuration. For details on how to configure the AJAX proxy refer to the section about AJAX proxy configuration.

Configuring outbound HTTP connections:

In WebSphere Portal Express Version 8.0 and earlier versions, outbound HTTP connections were accessible through the Ajax Proxy service. The Ajax Proxy service was configured by a configuration document named `proxy-config.xml`. You find this document in the `/WEB-INF` directory of the web module that uses the Ajax Proxy service. Starting with WebSphere Portal Express Version 8.5 and the new outbound connection service, the configuration of outbound HTTP connections is now part of the standard datastore-based portal configuration.

About this task

For more detailed information, read the following topics.

"Configuration structure"
The Outbound Connection Service Configuration model has the following structure.

"XML format for outbound HTTP connection configuration settings" on page 3000
To export or import the outbound HTTP Connections configuration, you use an XML file with a specific schema.

"Administration tools for configuring outbound HTTP connections" on page 3012
Portal system administrators can administer the outbound HTTP connection configuration in one of two ways: either by using configuration tasks, or by using the portal Model and Controller SPIs.

"Sample administration tasks" on page 3023
Here you find some procedures for running administration tasks on the outbound HTTP connection settings. These tasks use the portal configuration engine interface.

Configuration structure:

The Outbound Connection Service Configuration model has the following structure.

For more detailed information about the configuration structure of the outbound connection service, read the following topics. Examples follow in later topics.

"Outbound connection profile" on page 2993
The outbound connection profile contains the elements of the configuration model. The following profiles can exist.

"Policy mappings" on page 2994
When an outbound HTTP connection is established through the outbound connection service, the caller can specify the mapping context in an optional parameter.

"Policy rules" on page 2994
Policy rules define handling rules for outbound HTTP connections. Policy rules are identified by a URL pattern. When an outbound HTTP connection is opened, the portal compares the URL of that outbound HTTP connection against the URL pattern of the policy rule and handles the URL accordingly.

"Cookie rules" on page 2995
Cookie rules define handling rules for cookies that you use in the context of an outbound HTTP connection.

"Policy variables" on page 2997
You can use policy variables to resolve tokens of the form `{$Variable_name}` that can appear in the URL pattern setting of a proxy rule or in metadata.

Outbound connection profile:

The outbound connection profile contains the elements of the configuration model. The following profiles can exist.

System profile
This profile is reserved for configuration settings that are used by the system internally. A portal configuration contains exactly one system profile.

Global profile
This profile contains all global configuration settings. These settings are maintained by the portal administrator. In contrast to settings in application-scoped profiles, the policy mappings, proxy rules, and cookie rules that are defined in the global profile are available to all applications that use the outbound connection service. A portal configuration contains zero or one global profile.

Application-scoped profiles
Application scoped profile settings are valid within the scope of a specific web module. The application for which this profile is scoped is determined by the context root of the web module deployment descriptor. For example, if the deployment descriptor of the web module `bannerad.war` contains the context root setting `/wps/PA_Banner_Ad`, the application-scope profile must contain only `/wps/PA_Banner_Ad` as a scope reference.

An outbound connection profile item contains the application scope reference setting:

application scope reference
This setting is `null` for the system profile and for the global profile. For application-scoped profiles, the scope reference identifies the application to which this profile belongs. The scope reference contains the context root setting of the application. You can locate this information by either of the following ways:

- If the referenced application is a portlet application, proceed as follows:
  1. Log in to the portal by using the portal administrator user ID.
  2. Click the **Administration menu** icon. Then, click **Portlet Management** > **Web Modules**.

3. In the **Manage Web Modules** portlet, click **Select Web Module Properties** for the requested web module. The web module properties view lists the context root. For example, this context root can be `/wps/PA_Banner_Ad`.

- By alternative, you can determine the context root of an application by using the WebSphere Integrated Solutions Console. Proceed as follows:
  1. Select the enterprise application that you want to refer to.
  2. Click **View Deployment Descriptor**.
  3. Locate the context-root tag.
  4. Take the value of this tag as scope reference value.

Policy mappings:

When an outbound HTTP connection is established through the outbound connection service, the caller can specify the mapping context in an optional parameter.

Policy mappings have the following two purposes:

- The policy mapping is the container of the policy rules that define the characteristics of the outbound connections.
- If the outbound connection is accessed through the Ajax proxy, the policy mapping defines the connection settings for the Ajax proxy.

The following code sample shows how you create an outbound connection service by using the policy mapping context path `/mycontext`:

```
String theMappingContextPath = "/mycontext";

OutboundConnectionServiceHome home = (OutboundConnectionServiceHome)
          ctx.lookup("portal:service/model/OutboundConnectionService");
OutboundConnectionService connectionService = home.getOutboundConnectionService
          (request, Response, theMappingContextPath);
```

If the connection is established through the Ajax proxy, the mapping context root is used as context root for the proxy:

```
http://localhost:10039/wps/mycontext/(...encoded remote url ...);
```

An outbound connection profile contains at least one policy mapping item. This item is created when the outbound connection profile element is created. This default policy mapping does not have a context path. Therefore, policy rules that are defined in the default policy mapping are always accessible to outbound HTTP connection filters, independent of the context for which the connection was created. In the previous example, the following policy rules are available for the connection:

- The policy rules that are defined at the Policy Mapping with the context path `/mycontext`.
- The policy rules that are defined at the default policy mapping.

Policy rules:

Policy rules define handling rules for outbound HTTP connections. Policy rules are identified by a URL pattern. When an outbound HTTP connection is opened, the portal compares the URL of that outbound HTTP connection against the URL pattern of the policy rule and handles the URL accordingly.

A policy rule contains the following settings:

URL pattern:
Policy rules are identified by their URL patterns. If an outbound HTTP connection to a certain URL is connected, the outbound HTTP connection service checks the URL patterns of all policy rule items in the accessible policy mappings. The policy rule with the URL pattern that matches best is selected.

basicAuth
This flag indicates whether the connection that is defined by this policy rule is protected by basic authentication. If you set this key to `true`, the connection is protected by basic authentication. If you set it to `false`, the connection is not protected by basic authentication.

active This flag indicates whether this rule is active or not.

A set of allowed methods:
A policy rule can contain a set of allowed HTTP methods. This list restricts the HTTP methods for the URL connection. For example, if you specify `Get, Post` as the set of allowed methods, then only GET and POST requests are permitted for the URLs that this policy rule controls. If a policy rule does not contain a list of allowed methods, then all HTTP methods are permitted.

A set of mime types:
A policy rule can contain a set of mime types. They restrict the allowed mime type of the content that is received. For example, if you specify the set `text/html, text/plain`, it restricts outbound HTTP connections to content that is either plain text or HTML output. If you specify no list of supported mime types, then outbound HTTP connections are not restricted to certain mime types.

A set of Headers:
A policy rule can restrict the outbound connection to a set of request headers that are allowed for the remote connection.

A custom chain of URL connection filters:
To add an outbound HTTP connection that provides custom function, you can add a chain of custom URL connection filters. The outbound HTTP connection service provides a filter API. Programmers can use this API to write filters that are called whenever an outbound HTTP connection is opened. For more information, read *Program extensions for outbound HTTP connections*.

Related tasks:

"Using programmatic extensions for outbound HTTP connections" on page 3054
To extend the functions of an outbound connection, portal administrators can implement a custom outbound service filter.

Cookie rules:

Cookie rules define handling rules for cookies that you use in the context of an outbound HTTP connection.

Cookies are set by the remote server by using the `Set-Cookie:` response header setting. The cookie rules determine the handling of a created cookie. The cookie rule defines how this cookie is treated. Cookie rule definitions are owned by policy rules. Each policy rule defines individually how to handle cookies that are set in outbound HTTP connections.

A Cookie Rule contains the following settings:

cookie names
Use this setting to specify a set of wildcard expressions that contain names or name patterns of cookies. For example, the cookie rule with the name `LtpaToken*` applies to both LtpaToken and LtpaToken2. A cookie rule is owned by a proxy rule. Therefore, a cookie rule is applied if both of the following conditions apply:

- The owning proxy rule is applied.
- The cookie name matches with the wildcard expression.

cookie handling
Use this setting to define how the outbound HTTP connection service handles remote cookies. The outbound HTTP connection service can handle cookies in the following ways:

block This value is the default value. Cookies that you define as blocked cookies are filtered out: They are not returned in the response header of the outbound HTTP connection.

store in session
Cookies are stored in a cookie store that is placed in the local HTTP session.

store in request
Cookies are stored in a cookie store that is placed in the local HTTP request.

passthru
Cookies of the handling type `passthru` are copied into the response header of the connection of the Ajax proxy. The domain and cookie path of the cookie that is passed through are converted to the domain and path of the Ajax proxy servlet. The handling type `passthru` takes effect only if the outbound connection is established through the Ajax proxy.

wrap If you use cookies of the handling type `passthru`, they can conflict with local cookies, for example LtpaToken, LtpaToken2, or JSESSIONID cookies. In this case, you can use the handling type `wrap`. Cookies of handling type `wrap` are handled like cookies in `passthru` mode, but additionally, the cookie name is transformed.

cookie scope
This setting defines the owner of this cookie. A cookie can be associated with the following scopes:

user The cookie is scoped to the current user.

application
The cookie is scoped to the application that calls the outbound HTTP connection service.

system
The cookie is not scoped at all.

cookie transformations
This setting defines a programming interface. Application developers can use it to implement a custom cookie transformation handler. The custom extension code is called at the following two occasions:

1. Before the remote HTTP connection writes the request header to the remote connection

2. When the response header of the remote HTTP connection is evaluated.

   The custom transformation handler can modify the name, value, domain, and path of the cookie that is assigned with the custom transformation handler. For more information, read *Using custom cookie transformation handlers*.

Related tasks:

"Using custom cookie transformation handlers" on page 3059
You can use custom cookie transformation handlers to allow custom program extensions to get programmatic control over the outbound HTTP connection.

Policy variables:

You can use policy variables to resolve tokens of the form `{$Variable_name}` that can appear in the URL pattern setting of a proxy rule or in metadata.

Policy variables are used in the `urlPattern` setting of a proxy rule item, or in a metadata section. There are two types of policy variables:

Endpoint variables

Endpoint variables define a single-value variable. They are typically used to define a specific backend host. When application developers use endpoint variables, they can write a policy rule so that it extracts those parts of the URL pattern in a variable that are specific to the current portal environment. Example:

1. The portlet developer defines a proxy rule with the following URL pattern setting: `{$my_server}/mymail*`. This means that the resulting URL pattern is the value of the policy variable, appended with the path `/mymail`.
2. The portal system administrator defines the value `http://www.the-remote-system.com` for the variable `my_server`. Therefore, the URL pattern of the policy rule is `http://www.the-remote-system.com/mymail`.

The benefit of this technique is that you can divide a URL pattern into parts that are owned by the portlet developer and parts that are owned by the portal system administrator. Therefore, the portlet developer can define a policy rule without knowing the concrete host name of the backend system to which the outbound connection is established.

Dynamic policy variables

Dynamic policy variables can have more than one value. The purpose of this variable type is to define dynamic parts of the URL pattern without having to specify a policy rule more than once. Example:

- A policy rule URL pattern has the following URL pattern setting: `http://localhost/wps/{$my_dynamic_policy}/Main/*`.
- The variable `my_dynamic_policy` is associated with the values `portal` and `myportal`.

As a result, the policy rule applies to URLs that start with `http://localhost/wps/portal/Main/` or `http://localhost/wps/myportal/Main/`. The purpose of this technique is to make one policy rule available for multiple URL patterns that can contain dynamic parts.

For hints and tips for working with policy variables and setting them, read the following topic.

"Setting policy variables" on page 2998
You can set policy variables in different ways.

Setting policy variables:

You can set policy variables in different ways.

- You can set a policy variable in the portal WP Configuration Service Resource Environment Provider (REP) properties. This method is the preferred method to place global settings independent from the policies that refer to them.
- You can set a policy variable in the outbound HTTP connection configuration. This method is the preferred way for values or selections that are closely related with the policy rule.
- You can set a policy variable at run time. To do so, you use a URL query string parameter. Use this method if your policy contains dynamic elements that you want to define at run time.

For more details about these methods, read the following sections.

Setting policy variable values in the Resource Environment Provider properties

You can set policy variables in the portal WP Configuration Service Resource Environment Provider (REP) properties. You do so in the custom properties section of the WebSphere Integrated Solutions Console. For more information about setting properties in the portal configuration services, read Setting service configuration properties.

The portal matches policy variables based on the following custom property name scheme:

`wp.proxy.config.urlreplacement.`*`variable_name`*`.suffix`

The partial strings of the naming scheme have the following meanings:

`wp.proxy.config.urlreplacement`
This prefix identifies the property for defining a dynamic policy replacement value.

`variable_name`
This variable identifies the target dynamic policy variable that is used in the outbound HTTP connection configuration.

`suffix`
This suffix represents a string of your choice. If there is more than one key, this string makes the key unique among all replacement keys that refer to the same dynamic policy replacement variable.

Example: The following policy of an outbound HTTP connection configuration references the dynamic policy variable `my_remote_hosts`:

```
<policy url="http://{$my_remote_host}/*" >
     <actions>
     <method>GET</method>
     </actions>
</policy>
```

To attach this policy to outbound requests to URLs that match one of the URL patterns `http://www.some.server.com` or `http://some.other.server.com`, you can add the following two custom properties to the WP Configuration Service REP:

```
wp.proxy.config.urlreplacement.my_remote_host.1=http://www.some.server.com
wp.proxy.config.urlreplacement.my_remote_host.2=http://some.other.server.com
```

Notes: Dynamic policy values that are defined in the WP Configuration Service REP properties are not scoped. Therefore, such dynamic policy variables can be

used both by policies that are set in the global configuration profile and by policies that are set in an application-scoped profile.

Note: For configuration changes in the WP Configuration Service REP properties to take effect, restart IBM WebSphere Application Server.

Setting policy variables in the outbound HTTP connection configuration

As an administrator, you can define policy variables directly in the Outbound HTTP Connection configuration. Example:

```
<variables>
     <dynamic-policy name="my_path">
          <value>wps/portal</value>
          <value>wps/contenthandler</value>
     </dynamic-policy>
</variables>
...
<policy url="http://www.myremotesite.com/{$my_path}/*" >
     <actions>
     <method>GET</method>
     </actions>
</policy>
```

In this example, the policy variable `my_path` is defined in the outbound HTTP connection configuration. The variable is scoped to the configuration profile, in which the variable was defined:

- If the variable is defined in the global configuration, the variable can be referenced by policies or metadata settings of the global configuration.
- If the variable is defined in an application-scoped configuration, the variable can be referenced by policies or metadata settings of the same application-scoped profile.

To create, modify, or delete configuration-based policy variables, use the administration tools of the outbound HTTP connection services. For more detailed information, read Administration tools for configuring outbound HTTP connections.

Setting policy variables at run time

There are also cases where an application developer wants to set the policy variable directly by the application. Example: An application wants to choose the credential slot ID of an authenticated HTTP connection. For these scenarios, the outbound HTTP connection infrastructure provides a means to specify policy variables at run time. There are two ways to define policy variables for runtime resolution:

- The policy variable is defined as a dynamic-policy variable.
- The policy variable is defined as an endpoint variable. Its value is empty. At run time, the client sets the policy variable value. The value is not validated.

Security note: Define runtime policy variables carefully. Run time policy variables can have a security impact.
Example: The portal administrator wants to define two different user credentials for an outbound HTTP connection to the URL `http://www.myremotehost.com/test`. The user wants to select the credentials when the connection is created. A configuration defines the following policy variable:

```
name=SLOTID
type=Dynamic Policy
Values= MySlot2, MySlot33
```

The variable is referenced by the Credential Vault slot description that is defined in the metadata section of a policy mapping `/myproxy` as illustrated later:

```
metadata name:  paa.slotid
metadata value: {$SLOTID}
```

At run time, the client submits the following request at the Ajax proxy:

```
HTTP://localhost/wps/myproxy/www.myremotehost.com/test?SLOTID=MySlot2
```

At run time, the authentication filter of the outbound HTTP connection server reads the metadata `paa.slotid`. By doing so, it obtains the slot ID for the credentials that you want to be used for the outbound HTTP connection. The value of `paa.slotid` is mapped to the policy variable `SLOTID`. The client that called the connection defined a slot ID value `MySlot2`, which is in the valid set of policy variables.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Related reference:

"Administration tools for configuring outbound HTTP connections" on page 3012
Portal system administrators can administer the outbound HTTP connection configuration in one of two ways: either by using configuration tasks, or by using the portal Model and Controller SPIs.

XML format for outbound HTTP connection configuration settings:

To export or import the outbound HTTP Connections configuration, you use an XML file with a specific schema.

Example of a `proxy-config.xml` file

The following example is a sample outbound HTTP connection configuration script. It shows the schema for XML scripts that you can use to export or import the outbound HTTP Connections configuration.

```
<?xml version="1.0" encoding="UTF-8"?>
<proxy-rules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="http://www.ibm.com/xmlns/prod/sw/http/outbound/proxy-config/2.0">
    <variables>
        <dynamic-policy name="default_policy">
           <value>http://www.ibm.com/*</value>
           <value>http://www-3.ibm.com/*</value>
        </dynamic-policy>
    </variables>
    <mapping contextpath="/proxy" url="*" name="proxy"/>
    <mapping contextpath="/myproxy" url="*" name="myproxy">
       <policy url="http://www.test-server.com/*" name="ibm1">
           <actions>
              <method>GET</method>
           </actions>
     <cookie-rule name="my.application.cookie">
        <cookie>TestCookie</cookie>
              <scope>user</scope>
           <handling>store-session</handling>
     </cookie-rule>
        </policy>
```

```
    </mapping>
    <policy url="{$default_policy}" name="default">
        <actions>
            <method>GET</method>
            <method>HEAD</method>
        </actions>
    </policy>

    <meta-data>
        <name>socket-timeout</name>
        <value>10000</value>
    </meta-data>
</proxy-rules>
```

"XML schema of outbound HTTP connection configuration script"
An outbound connection configuration script file needs to conform to the following XML schema.

"Description of the outbound HTTP connection configuration script" on page 3003
The configuration settings of an outbound HTTP connection configuration script are described here.

XML schema of outbound HTTP connection configuration script:

An outbound connection configuration script file needs to conform to the following XML schema.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- ************************************************************** -->
<!--                                                                -->
<!-- Licensed Materials - Property of IBM                           -->
<!--                                                                -->
<!-- 5724-U69                                                       -->
<!--                                                                -->
<!-- Copyright IBM Corp. 2013  All Rights Reserved.                 -->
<!--                                                                -->
<!-- US Government Users Restricted Rights - Use, duplication or    -->
<!-- disclosure restricted by GSA ADP Schedule Contract with        -->
<!-- IBM Corp.                                                      -->
<!--                                                                -->
<!-- ************************************************************** -->

<xs:schema xmlns:proxy="http://www.ibm.com/xmlns/prod/sw/http/outbound/proxy-config/2.0"
           xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
    <xs:element name="proxy-rules" type="proxy-rulesType">
        <xs:unique name="mappingID">
            <xs:selector xpath=".//mapping" />
            <xs:field xpath="@name" />
        </xs:unique>
    </xs:element>

    <xs:complexType name="mappingType">
        <xs:choice minOccurs="0" maxOccurs="unbounded">
            <xs:element name="policy" type="policyType" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="ipfilter" type="ipfilterType" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="meta-data" type="meta-dataType" minOccurs="0" maxOccurs="unbounded"/>
        </xs:choice>
        <xs:attribute name="contextpath" type="xs:string" use="required"/>
        <xs:attribute name="url" use="optional">
            <xs:simpleType>
                <xs:restriction base="xs:anyURI"/>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="name" type="xs:string" />
    </xs:complexType>

    <xs:complexType name="policyType">
        <xs:sequence>
            <xs:element name="actions" type="actionsType" />
            <xs:element name="headers" type="headersType"
                minOccurs="0" />
            <xs:element name="mime-types" type="mime-typesType"
                minOccurs="0" />
            <xs:element name="cookie-rule" type="cookie-ruleType" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="filter-chain" type="filter-chainType"
                minOccurs="0" />
            <xs:element name="meta-data" type="meta-dataType"
                minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
        <xs:attribute name="url" type="xs:anyURI" use="required" />
        <xs:attribute name="basic-auth-support" type="xs:boolean"
            use="optional" default="false" />
        <xs:attribute name="active" type="xs:boolean" default="true"/>
```

```
    <xs:attribute name="name" type="xs:string"/>
</xs:complexType>

<xs:complexType name="ipfilterType">
    <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element name="allow" type="xs:string"/>
        <xs:element name="deny" type="xs:string"/>
    </xs:choice>
</xs:complexType>

<xs:complexType name="actionsType">
    <xs:sequence>
        <xs:element name="method" type="methodType" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>

<xs:simpleType name="methodType">
    <xs:restriction base="xs:string">
        <xs:enumeration value="GET"/>
        <xs:enumeration value="POST"/>
        <xs:enumeration value="PUT"/>
        <xs:enumeration value="HEAD"/>
        <xs:enumeration value="DELETE"/>
    </xs:restriction>
</xs:simpleType>

<xs:complexType name="headersType">
    <xs:sequence>
        <xs:element name="header" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>

<xs:complexType name="mime-typesType">
    <xs:sequence>
        <xs:element name="mime-type" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>

<xs:complexType name="meta-dataType">
    <xs:sequence>
        <xs:element name="name" type="xs:string"/>
        <xs:element name="value" type="xs:string"/>
    </xs:sequence>
</xs:complexType>

<xs:complexType name="proxy-rulesType">
    <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element name="variables" type="varsType" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="mapping" type="mappingType" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="policy" type="policyType" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="ipfilter" type="ipfilterType" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="meta-data" type="meta-dataType" minOccurs="0" maxOccurs="unbounded"/>
    </xs:choice>
</xs:complexType>

<xs:complexType name="filter-chainType">
    <xs:sequence>
        <xs:element name="filter-factory" type="filter-factoryType" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>

<xs:complexType name="filter-factoryType">
    <xs:sequence>
        <xs:element name="classname" type="javaClassName" />
        <xs:element name="meta-data" type="meta-dataType" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>

<xs:complexType name="cookie-ruleType">
    <xs:sequence>
        <xs:element name="cookie" type="xs:string" maxOccurs="unbounded"/>
        <xs:element name="scope" type="cookieScopeType" minOccurs="0" default="user"/>
        <xs:element name="handling" minOccurs="0" type="cookieHandlingType" default="passthru"/>
        <xs:element name="transformation" minOccurs="0" maxOccurs="unbounded" type="filter-factoryType"/>
    </xs:sequence>
    <xs:attribute name="name" type="xs:string"/>
</xs:complexType>

    <xs:complexType name="varsType">
        <xs:sequence>
            <xs:element name="endpoint" type="endpointType" minOccurs="0" maxOccurs="unbounded" />
            <xs:element name="dynamic-policy" type="dynamicPolicyType" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="endpointType">
        <xs:simpleContent>
            <xs:extension base="xs:string">
                <xs:attribute name="name" type="nameType" use="required"/>

            </xs:extension>
```

```
            </xs:simpleContent>
        </xs:complexType>

        <xs:complexType name="dynamicPolicyType">
            <xs:sequence>
                <xs:element name="value" type="policyValueType" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:attribute name="name" type="nameType" use="required"/>
        </xs:complexType>

    <xs:simpleType name="javaClassName">
      <xs:restriction base="xs:Name">
        <xs:pattern value="[a-zA-Z0-9\.]+"/>
      </xs:restriction>
    </xs:simpleType>

        <xs:simpleType name="policyValueType">
                <xs:restriction base="xs:string"/>
        </xs:simpleType>

        <xs:simpleType name="nameType">
                <xs:restriction base="xs:string"/>
        </xs:simpleType>

    <xs:simpleType name="cookieScopeType">
        <xs:restriction base="xs:string">
            <!-- default -->
            <xs:enumeration value="user"/>
            <xs:enumeration value="system"/>
            <xs:enumeration value="application"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="cookieHandlingType">
        <xs:restriction base="xs:string">
            <!-- default -->
            <xs:enumeration value="passthru"/>
            <xs:enumeration value="store-in-session"/>
            <xs:enumeration value="block"/>
            <xs:enumeration value="store-persistent"/>
            <xs:enumeration value="store-in-request"/>
            <xs:enumeration value="wrap"/>
        </xs:restriction>
    </xs:simpleType>

</xs:schema>
```

Description of the outbound HTTP connection configuration script:

The configuration settings of an outbound HTTP connection configuration script are described here.

proxy-rules

The `proxy-rules` setting is the root of a configuration profile. The configuration profile for which the script is applied is specified in the configuration task that is run against this script. The `proxy-rules` setting can contain the following subsettings:

variables

Use the `variables` setting to specify endpoint variables or dynamic policy variables. Variables are used in the URL attribute of the policy setting, or they are used in a variable value. For more details about how to use variables, read *Variables*.

mapping

Use the `mapping` setting to map incoming requests to a target URL, based on their context path. Therefore, each mapping must specify a `contextpath` attribute and optionally a `url` attribute. Optionally, mappings can declare policy settings that represent mapping specific access policies. For more details about how to use mappings, read *Context path mappings*. Mappings contain a name attribute that gives an administrative name for this mapping. This attribute is used by administrative tasks to identify the mapping for update tasks or delete tasks.

policy

Use the `policy` setting to define an access policy for a specific URL pattern. A proxy configuration can contain multiple policy definitions. If no policy is specified at all, the portal denies all incoming requests. For more details about policies, read *Access policies*.

ipfilter

Use `ipfilter` setting to define one or more IP patterns. You can use these IP filter patterns to either grant or deny a particular IP address or a set of addresses access to the Ajax proxy. For more details about IP filtering rules, read *IP filtering*.

meta-data

Use the `meta-data` setting to specify general configuration properties of the proxy, for example HTTP-related parameters. Each `meta-data` setting must have a `name` and a `value`. To get a list of configuration parameters that are available in the portal, read *General configuration parameters by metadata*.

"Variables"
Variables settings define variables that are used in other sections of the configuration.

"Context path mapping" on page 3005
You use a context path mapping to map a specific context path to a specific target URL. The proxy resolves context path mappings before it applies the matching access policy.

"Policies" on page 3006
Each `policy` setting defines an access policy for a URL pattern. You specify the pattern by using the **`url`** attribute. A **`url`** attribute can be either a URL, or the wildcard character, or URL part that ends with the wildcard character "*". The following are examples for url attribute values: `http://localhost/index.html` `*http://www.ibm.com/developerWorks/*`.

"IP filtering" on page 3007
You can use `ipfilter` settings to declare IP filtering rules and to either grant or deny a client access to the Ajax proxy.

"Configuration metadata for outbound HTTP connections" on page 3008
You can add general proxy configuration parameters to the file `proxy-config.xml` by using `meta-data` settings.

"Cookie rule" on page 3011
You can use cookie rules to determine how you want the cookie to be handled.

Related concepts:

"The programming model for the outbound HTTP connection service" on page 2986
The outbound HTTP connection service can be used from the context of a servlet request service or from the context of a portlet request service. Here are some code examples.

Variables:

Variables settings define variables that are used in other sections of the configuration.

There are two types of variable settings, endpoint variables and dynamic-policy variables:

endpoint variables

Endpoint variables define a single endpoint. For example, this endpoint can be the host name of a remote server. You specify the value of an endpoint in the content of an endpoint variable.

dynamic-policy variables

Dynamic-policy variables can have multiple values. Dynamic-policy variables contain one or more value elements. All of these value elements apply to the variable.

The following example illustrates the usage of an endpoint variable:

```
<proxy-rules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="http://www.ibm.com/xmlns/prod/sw/http/outbound/proxy-config/2.0">
     <variables>
          <endpoint name="the_remote_host">www.myremotehost.com</endpoint>
     </variables>
     <policy url="http://{$the_remote_host}/*" name="endpoint_sample">
          <actions><method>GET</method></actions>
     </policy>
</proxy-rules>
```

The following example shows the usage of a dynamic policy variable. The policy that is defined in this example applies to the following URL patterns:

- `http://www.myremotehost.com/*`
- `http://w3.myremotehost.com/*`
- `https://www.myremotehost.com/*`
- `https://w3.myremotehost.com/*`

```
<proxy-rules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="http://www.ibm.com/xmlns/prod/sw/http/outbound/proxy-config/2.0">
     <variables>
          <dynamic-policy name="my_prot_selector">
               <value>http</value>
               <value>https</value>
          </dynamic-policy>

          <dynamic-policy name="the_remote_host">
               <value>www.myremotehost.com</value>
               <value>w3.myremotehost.com</value>
          </dynamic-policy>
     </variables>
     <policy url="{my_prot_selector}://{$the_remote_host}/*" name="dyn_policy_example">
          <actions><method>GET</method></actions>
     </policy>
</proxy-rules>
```

Context path mapping:

You use a context path mapping to map a specific context path to a specific target URL. The proxy resolves context path mappings before it applies the matching access policy.

To work with context path mappings, you work with the `mapping` setting. An example `mapping` setting can look as follows:

```
<mapping contextpath="/ibmproducts" url="http://www.ibm.com/products"/>
```

It maps requests that contain the context path `/ibmproducts` to the URL `http://www.ibm.com/products`. As a result, the incoming proxy URL is `http://myportal.com/wps/ibmproducts/us/en`, the proxy forwards the request to `http://www.ibm.com/products/us/en`. For more details about the proxy URL format and the usage of context path mappings, read *The programming model for the outbound HTTP connection service*.

You can also define generic context path mappings that are not tied to a specific URL pattern. You do so by specifying the asterisk as a wildcard: `url="*"`. Example:

```
<mapping contextpath="/proxy" url="*"></mapping>
```

However, if your application must connect to only a few external systems, for example to one external REST service that provides the application data, it is better to use a specific context path mapping.

Note: The current servlet-based implementation of the proxy requires a corresponding servlet mapping for each defined `mapping` setting. You define the servlet mapping in the file `web.xml`. The servlet mapping maps all requests that address the specified context path to the proxy servlet. If you apply this rule to the previous examples, there must be a servlet mapping for either of the context paths `"/ibmproducts"` or `"/proxy"` that references the proxy servlet.

Policies:

Each `policy` setting defines an access policy for a URL pattern. You specify the pattern by using the **`url`** attribute. A **`url`** attribute can be either a URL, or the wildcard character, or URL part that ends with the wildcard character "*". The following are examples for url attribute values: `http://localhost/index.html` `*http://www.ibm.com/developerWorks/*`.

The following example shows a simple policy:

```
<proxy-rules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="http://www.ibm.com/xmlns/prod/sw/http/outbound/proxy-config/2.0">
     <policy name="SamplePolicy" url="http://www.myremotehost.com/*">
          <actions><method>GET</method></actions>
     </policy>
</policy-rules>
```

For each incoming request, the proxy applies the policy with the best URL match. If the proxy finds a policy, it applies this policy to the outbound connection. If the proxy finds no matching policy, the proxy rejects the request. Mappings can optionally declare `policy` setting that represents mapping-specific access policies.

A `policy` setting can have a `name` attribute that identifies this policy for administrative tasks. If this attribute is not set, the portal sets a unique administrative name for it. The administrative name must be unique for all policies that have the same parent `mapping` setting or parent `proxy-rules` setting.

To enable Basic Authentication for a policy, you can set the attribute `basic-auth-support` to `true`.

A `policy` setting can have the subsettings that are shown in the following list. Specify the subsettings in the same order in the configuration file `proxy-config.xml` as in the list:

actions

This setting is mandatory. Use it to define the list of HTTP methods that can be used to access resources in the target domain. These methods are `GET`, `HEAD`, `POST`, `PUT`, and `DELETE`. The proxy denies requests that use HTTP methods that are not on this list. Specify each HTTP method by using a separate `method` setting. The following example shows a policy that supports the methods DELETE, GET, HEAD, POST, and PUT:

```
<proxy-rules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="http://www.ibm.com/xmlns/prod/sw/http/outbound/proxy-config/2.0">
     <policy name="SamplePolicy_supporting_all_methods" url="http://www.myremotehost.com/*">
          <actions>
```

```
                <method>GET</method>
                <method>HEAD</method>
                <method>POST</method>
                <method>PUT</method>
                <method>DELETE</method>
            </actions>
        </policy>
</policy-rules>
```

headers

This setting is optional. Use it to define the list of header names that you want the proxy to forward to the target domain. The header names can include wildcard characters. If you specify no header names for the policy, the proxy by default forwards headers that match the following name expressions: `Cache-Control`, `Pragma`, `User-Agent`, `Accept*`, `Host`, and `Content*`. Specify each header name by using a separate `header` setting.

Note: The value `Cookies` is not allowed. Instead, use the `cookie-rules` setting to specify the cookie forwarding behavior for the policy.
The following example shows a policy that supports the `X-Method-Override` header:

```
<proxy-rules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="http://www.ibm.com/xmlns/prod/sw/http/outbound/proxy-config/2.0">
     <policy name="SamplePolicy_custom_header_added" url="http://www.myremotehost.com/*">
          <actions><method>GET</method></actions>

          <headers>
               <header>X-Method-Override</header>
          </headers>
     </policy>
</proxy-rules>
```

mime-types

This setting is optional. Use it to specify the list of accepted mime types. The mime types refer to the response that the proxy receives from the target server. If you specify at least one mime type, the proxy accepts only responses with a `Content-Type` response header that matches one of the specified mime types. If you specify no mime type, the proxy accepts all responses. Specify each mime type by using a separate `mime-type` setting. Servers might append the character encoding to the mime type. Therefore, it can be useful to use wildcard characters when you specify mime types. For example, if you specify `text/html*`, the proxy also accepts responses with the Content-Type `text/html; charset=utf-8`. The following example shows a policy that uses mime type filtering:

```
<proxy-rules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="http://www.ibm.com/xmlns/prod/sw/http/outbound/proxy-config/2.0">
     <policy name="SamplePolicy_filtered_mimetype" url="http://www.myremotehost.com/*">
          <actions><method>GET</method></actions>

          <mime-types>
               <mime-type>text/plain</mime-type>
          </mime-types>
     </policy>
</proxy-rules>
```

IP filtering:

You can use `ipfilter` settings to declare IP filtering rules and to either grant or deny a client access to the Ajax proxy.

You can use `allow` settings to grant access to a particular IP address or set of addresses. By alternative, you can use `deny` settings to deny access to a particular IP address or set of addresses.

The `allow` and `deny` settings support the following value formats:

- Net address and bit number. Example: `192.0.0.0/24`

- Net address and NetMask. Example: `192.0.0.0/255.255.255.0`
- Specific IP address. Example `192.168.0.1`
- Specific IP address with wildcards. Example `192.168.*.1`

Note: If you declare multiple `ipfilter` settings in the proxy configuration, the proxy processes them by the sequence in which you specify them. As a result, the last matching rule always takes effect, regardless of the previous rules.
The following example makes the policies in the `/myproxy` mapping accessible only to clients that have an IP address in the specified range:

```
<proxy-rules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="http://www.ibm.com/xmlns/prod/sw/http/outbound/proxy-config/2.0">
     <mapping contextPath="/myproxy" url="*">
          <policy name="SamplePolicy" url="http://www.myremotehost.com/*">
               <actions><method>GET</method></actions>
          </policy>
          <ipfilter>
               <allow>192.168.1.*</allow>
          </ipfilter>
     </mapping>
</policy-rules>
```

Configuration metadata for outbound HTTP connections:

You can add general proxy configuration parameters to the file `proxy-config.xml` by using `meta-data` settings.

You can specify all of these parameters within a `proxy-rules` or `mapping` setting.

HTTP connection handling parameters:

Use the following configuration parameters to determine how the proxy handles the HTTP connection:

`socket-timeout`

Use this parameter to define the default socket timeout in milliseconds. The socket timeout determines how long the proxy server waits for data after it successfully establishes a connection with the target server. The default value is 20 seconds. A timeout value of zero means that a timeout is not applied.

`retries`

Use this parameter to define the number of retries that you want the proxy to do if it cannot establish a connection with the target server. The default value is 2 retries.

`max-total-connections`

Use this parameter to define the maximum number of HTTP connections that the proxy can open to connect to arbitrary target hosts. The default value is 100 connections.

`max-connections-per-host`

Use this parameter to define the number of HTTP connections that the proxy can open to connect to a specific host. The default value is 50 connections per host.

User Agent identifier:

Use the following configuration parameter to redefine the user agent parameter:

`user-agent`

Use this parameter to specify the user agent identifier that you want to be used for the outbound connection.

Control Redirection:

Use the following configuration parameter to specify the reaction on an HTTP redirect status (HTTP status 302):

follow-redirects

Set this parameter to `true` if the caller wants an HTTP redirect status to be resolved automatically, that is, the connection is reestablished to the redirection URL. Set this parameter to `false` if no automated redirection is required. In that case, the proxy returns an HTTP status 302 to the caller. If you do not set this parameter, then the default reaction on a 302 HTTP status depends on the HTTP request method:

- For all requests of `HEAD` and `GET`, the portal follows the HTTP redirection.
- For all other request methods, the portal does not follow the HTTP redirection.

Deactivate policy rules, mappings, or custom connection filters:

A policy rule, a mapping, or a custom connection filter can be set to be deactivated. In this case, the disabled policy rule or mapping remains in the configuration, but does not take effect at run time. Ensure that you use the metadata setting to control the activation state of a policy rule or mapping:

active

Valid values are `false` and `true`:

true This is the default value. If you set this parameter to `true`, the policy rule is in effect.

false If you want the policy rule to be deactivated, set the `active` parameter to the value `false`.

Security parameters:

Use the following configuration parameters to specify security-related settings of the proxy. You can also define these security-related parameters within `policy` settings.

unsigned_ssl_certificate_support

Valid values are `false` and `true`:

true If you set this parameter to `true`, the proxy connects to any HTTPS URL that is allowed by the policy, regardless of whether it trusts the specified host. This is the default value.

false If you set this parameter to `false`, the proxy connects only to HTTPS URLs that it trusts.

forward-http-errors

This parameter defines whether the proxy forwards extra HTTP error codes to the client. Valid values are `false` and `true`:

false This value is the default value. It means that only 2*xx* and 3*xx* status codes are forwarded, whereas 4*xx* error codes are automatically mapped to a `404 'Not Found'` error.

true If you set the `forward-http-errors` parameter to `true`, the proxy forwards every status code, even if it represents an error code.

forward-credentials-from-vault
This parameter defines whether user credentials that are retrieved from the credential vault of the portal can be forwarded to the specified target host by using an HTTP authorization request header. Valid values are `false` and `true`:

false This is the default value. If you set this parameter to `false`, the proxy does not forward credentials.

true If you set this parameter to `true`, the proxy forwards credentials.

xhr-authentication-support
This parameter defines whether HTTP BASIC user credential challenges from the external service are rewritten. Valid values are `false` and `true`:

false This value is the default value. If you set this parameter to `false`, the proxy does not change the `WWW-Authenticate` HTTP header, and the user is prompted for credentials.

true If you set this parameter to `true`, the browser does not ask the user to provide credentials. Instead, the client-side code can handle that challenge. In detail, the proxy changes the value of the HTTP header `WWW-Authenticate` from `BASIC` to `XHRBASIC`.

Configuring a Boundary (pass-through) Proxy:
Use the following configuration parameters to make connections through a boundary (pass-through) proxy:

passthru_host
This parameter defines the host name of the pass-through proxy.

passthru_port
This parameter defines the port of the pass-through proxy.

passthru_realm
This parameter is optional. It defines the authentication realm of the pass-through proxy.

passthru_username
This parameter is optional. It defines the user name for the pass-through proxy.

passthru_password
This parameter is optional. It defines the password for the pass-through proxy.

The following example shows a typical metadata setting:

```
<proxy-rules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="http://www.ibm.com/xmlns/prod/sw/http/outbound/proxy-config/2.0">
     <policy url="http://www.myremotehost.com/*" name="SamplePolicy">
          <actions><method>GET</method></actions>
     </policy>

     <meta-data>
          <name>socket-timeout</name>
          <value>10000</value>
     </meta-data>
     <meta-data>
          <name>retries</name>
          <value>2</value>
     </meta-data>
     <meta-data>
          <name>max-connections-per-host</name>
```

```
        <value>50</value>
    </meta-data>
    <meta-data>
        <name>max-total-connections</name>
    <value>100</value>
    </meta-data>
</proxy-rules>
```

Cookie rule:

You can use cookie rules to determine how you want the cookie to be handled.

The following options are available:

- The cookie can be made visible for the client that calls the outbound HTTP connection.
- The cookie can be wrapped in another cookie.
- The cookie can be stored in either of the following types of cookie store:
  - A cookie store that is scoped for the session
  - A cookie store that is scoped for the request

To configure your cookie rules, use the following attributes:

cookie
This attribute is required. You can specify multiple `cookie` attributes. `cookie` attributes contain a wildcard expression of a cookie name for which this cookie rule applies.

scope This attribute is optional. The `scope` attribute denotes the scope for which the cookie is shared. The following scope values are valid:

user This value is the default value. It defines that the cookie is scoped to the user that receives the cookie from the remote server.

system
This value defines the cookie as shared. That means that all clients of an outbound connection that establish a connection use the same cookie.

application
This value determines that the cookie is scoped to the application from which the outbound connection is established.

handling
This attribute is optional. It determines how to proceed with cookies that are defined by the remote server in outbound HTTP connections: The following handling values are valid:

passthru
This value is the default value. It leaves the cookie unchanged. The cookie is returned to the user of the outbound HTTP connection. It is up to the caller of the outbound HTTP connection to handle this cookie.

wrap This value wraps this cookie in another cookie. If you specify this value, the transformation rules define the characteristics of this transformation.

block This value blocks this cookie. If the cookie is set by the caller of the outbound HTTP connection, the cookie is not sent to the remote host. In the same manner, cookies that are set by the remote system are not received at the outbound HTTP connection.

store-in-request
This value stores the cookie in a cookie store during the HTTP request. Cookies that are set by the remote host are filtered out by the cookie handling filter. When the next HTTP connections are made, the stored cookies are added to the request header of this request.

store-in-session
This value stores the cookie in a cookie store during the HTTP session. Cookies that are set by the remote host are filtered out by the cookie handling filter. When the next HTTP connections are made, the stored cookies are added to the request header of this request.

The following example shows a cookie rule that contains a cookie rule for LTPA tokens, an application cookie that is named `app_cookie`, and a further cookie that is named `another_cookie`. By these cookie rules, the LTPA token is saved in a cookie store, and the two cookies `app_cookie` and `another_cookie` are passed through the proxy to the client.

```
<proxy-rules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="http://www.ibm.com/xmlns/prod/sw/http/outbound/proxy-config/2.0">
     <policy name="SamplePolicy" url="http://www.myremotehost.com/*">
          <actions><method>GET</method></actions>
          <cookie-rule name="my_sso_tokens">
               <cookie>LtpaToken*</cookie>
               <handling>store-in-session</handling>
          </cookie-rule>
          <cookie-rule name="my_application_cookies">
               <cookie>app_cookie</cookie>
               <cookie>another_cookie</cookie>
               <handling>passthru</handling>
          </cookie-rule>
     </policy>
</proxy-rules>
```

Administration tools for configuring outbound HTTP connections:

Portal system administrators can administer the outbound HTTP connection configuration in one of two ways: either by using configuration tasks, or by using the portal Model and Controller SPIs.

"Configuring outbound HTTP connections by using configuration tasks" on page 3013
Programmers can create, read, update, or delete settings of the outbound HTTP connection by using the appropriate portal configuration engine tasks.

"Configuring outbound HTTP connections by using the Model Controller SPI" on page 3019
To query and administer the outbound HTTP connection, you can also use the model controller SPI.

Related concepts:

"Model SPI overview" on page 2858
Models provide information that is needed by WebSphere Portal Express to perform tasks such as content aggregation or building navigation to browse the aggregated content. The information that is aggregated is represented through models that can be accessed programmatically by using the Model SPI (read-only). The information of a model is usually persistent (stored in a database) but can also be transient (computed and stored only in memory). Models can be represented by using a tree structure (nodes have a parent-child relationship), a list structure, or a selection structure (a selected element in a tree structure).

"Controller SPI " on page 2883
You can use the Controller SPI for portal administration. It allows you to modify portal resources. It enhances the read-only portal Model SPI by adding writable aspects.

Configuring outbound HTTP connections by using configuration tasks:

Programmers can create, read, update, or delete settings of the outbound HTTP connection by using the appropriate portal configuration engine tasks.

About this task

The portal provides the following configuration tasks for these tasks:

```
create-outbound-http-connection-config
read-outbound-http-connection-config
update-outbound-http-connection-config
delete-outbound-http-connection-config
clean-outbound-http-connection-config
```

For more information about how to work with outbound HTTP connection configuration settings, read the following topics. They also provide information about to work with application-scoped configuration settings.

"Creating an outbound HTTP connection configuration profile" on page 3014
This configuration task creates an outbound HTTP connection configuration profile by using the settings that you specify in an XML document. Use this task if you want to initially create outbound HTTP settings for your configuration. You can create a global configuration or an application-scoped configuration.

"Reading an outbound HTTP connection configuration profile" on page 3015
This configuration task reads an outbound HTTP connection configuration and exports it to an XML document. System administrators can use this document to update the configuration of an outbound HTTP connection. You can read a global configuration or an application-scoped configuration.

"Updating an outbound HTTP connection configuration profile" on page 3016
This configuration task updates the configuration profile settings of the outbound HTTP connection with the settings that are specified in an XML document. Use this task to manage existing outbound HTTP settings of your configuration. You can update a global configuration or an application-scoped configuration.

"Deleting outbound HTTP connection configuration settings" on page 3017
This configuration task deletes configuration settings for an outbound HTTP connection. It deletes the settings that are specified in an XML document. You can delete settings of the global configuration or of an application-scoped configuration.

"Clearing an outbound HTTP connection configuration profile" on page 3018
This configuration task clears the complete configuration profile for an outbound HTTP connection. It deletes all settings of an outbound HTTP connection profile. Whereas the `delete-outbound-http-connection-config` configuration task deletes only those configuration settings that are listed in the specified XML document, this task purges the entire HTTP connection configuration profile.

Creating an outbound HTTP connection configuration profile:

This configuration task creates an outbound HTTP connection configuration profile by using the settings that you specify in an XML document. Use this task if you want to initially create outbound HTTP settings for your configuration. You can create a global configuration or an application-scoped configuration.

About this task

The configuration task creates the specified configuration settings if they do not exist. In contrast to the configuration task **update-outbound-http-connection-config**, this task leaves existing configuration settings unchanged. The syntax of the configuration task differs, depending on whether you want to create the settings for a global configuration or for an application-scoped configuration.

Creating the global outbound HTTP connection profile:
About this task

To create outbound HTTP connection settings for the global configuration, use the following syntax:

IBM i

```
ConfigEngine.sh create-outbound-http-connection-config -DConfigFileName=your_path/config_file.xml
```

Linux

```
./ConfigEngine.sh create-outbound-http-connection-config -DConfigFileName=your_path/config_file.xml
```

Windows

```
ConfigEngine.bat create-outbound-http-connection-config -DConfigFileName=your_path\config_file.xml
```

`your_path/config_file.xml` is the absolute path name of the XML document that contains the configuration settings for the outbound HTTP connection infrastructure that you want to add.

For more information, see Adding an outbound connection policy.

Creating an application-scoped outbound HTTP connection profile:
About this task

To create outbound HTTP connection settings for an application-scoped configuration, use the following syntax:

IBM i

```
ConfigEngine.sh create-outbound-http-connection-config   -DConfigFileName=your_path/config_file.xml
                                                         -DApplicationScopeRef=scoperef
```

Linux

```
./ConfigEngine.sh create-outbound-http-connection-config -DConfigFileName=your_path/config_file.xml
                                                         -DApplicationScopeRef=scoperef
```

Windows

```
ConfigEngine.bat create-outbound-http-connection-config  -DConfigFileName=your_path\config_file.xml
                                                         -DApplicationScopeRef=scoperef
```

`your_path/config_file.xml` is the absolute path name of the XML document that contains the configuration settings for the outbound HTTP connection infrastructure that you want to add.

`scoperef` is the context root of the application to which the configuration settings are scoped. To obtain the scope reference, proceed as follows:

1. Access the WebSphere Integrated Solutions Console.
2. Select the enterprise application for which the outbound HTTP connection is scoped.
3. Click **View Deployment Descriptor**.
4. Locate the value of the `context root` tag.

An example context root is `/PA_Banner_Ad`.

Reading an outbound HTTP connection configuration profile:

This configuration task reads an outbound HTTP connection configuration and exports it to an XML document. System administrators can use this document to update the configuration of an outbound HTTP connection. You can read a global configuration or an application-scoped configuration.

About this task

To update an outbound HTTP connection configuration, you first create the XML document by using use the configuration task `read-outbound-http-connection-config`. Then, you modify the document as required and update the outbound HTTP connection configuration by using the configuration task `update-outbound-http-connection-config`.

The syntax of the configuration task differs, depending on whether you want to read the settings for a global configuration or for an application-scoped configuration.

Reading the global outbound HTTP connection profile:
About this task

To read the outbound HTTP connection settings for the global configuration, use the following syntax:

IBM i

```
ConfigEngine.sh read-outbound-http-connection-config -DConfigFileName=your_path/config_file.xml
```

Linux

```
./ConfigEngine.sh read-outbound-http-connection-config -DConfigFileName=your_path/config_file.xml
```

Windows

```
ConfigEngine.bat read-outbound-http-connection-config -DConfigFileName=your_path\config_file.xml
```

`your_path/config_file.xml` is the absolute path name of the XML document to which you want the configuration settings to be written.

Reading an application-scoped outbound HTTP connection profile:
About this task

To read the outbound HTTP connection settings for an application-scoped configuration, use the following syntax:

IBM i

```
ConfigEngine.sh read-outbound-http-connection-config   -DConfigFileName=your_path/config_file.xml
                                                       -DApplicationScopeRef=scoperef
```

Linux

```
./ConfigEngine.sh read-outbound-http-connection-config -DConfigFileName=your_path/config_file.xml
                                                       -DApplicationScopeRef=scoperef
```

Windows

```
ConfigEngine.bat read-outbound-http-connection-config  -DConfigFileName=your_path\config_file.xml
                                                       -DApplicationScopeRef=scoperef
```

`your_path/config_file.xml` is the absolute path name of the XML document to which you want the configuration settings to be written.

`scoperef` is the context root of the application to which the configuration settings are scoped. To obtain the scope reference, proceed as follows:

1. Access the WebSphere Integrated Solutions Console.
2. Select the enterprise application for which the outbound HTTP connection is scoped.
3. Click **View Deployment Descriptor**.
4. Locate the value of the `context root` tag.

An example context root is `/PA_Banner_Ad`.

Updating an outbound HTTP connection configuration profile:

This configuration task updates the configuration profile settings of the outbound HTTP connection with the settings that are specified in an XML document. Use this task to manage existing outbound HTTP settings of your configuration. You can update a global configuration or an application-scoped configuration.

About this task

The configuration task updates the specified configuration settings. It creates the specified configuration settings if they do not exist.

The syntax of the configuration task differs, depending on whether you want to update the settings for a global configuration or for an application-scoped configuration.

To update an outbound HTTP connection configuration, you use the configuration tasks `read-outbound-http-connection-config` and `update-outbound-http-connection-config`.

Procedure

1. Run the configuration task `read-outbound-http-connection-config`. It exports the outbound HTTP connection configuration to an XML document.
2. Update the XML document that resulted from the previous step as appropriate.
3. To apply your updates to the configuration, you use the configuration task `update-outbound-http-connection-config`.

Updating the global outbound HTTP connection profile:
About this task

To update the outbound HTTP connection settings for the global configuration, use the following syntax:

IBM i

```
ConfigEngine.sh update-outbound-http-connection-config -DConfigFileName=your_path/config_file.xml
```

Linux

```
./ConfigEngine.sh update-outbound-http-connection-config -DConfigFileName=your_path/config_file.xml
```

Windows

```
ConfigEngine.bat update-outbound-http-connection-config -DConfigFileName=your_path\config_file.xml
```

`your_path/config_file.xml` is the absolute path name of the XML document that contains the configuration settings for the outbound HTTP connection infrastructure that you want to update.

Updating an application-scoped outbound HTTP connection profile:
About this task

To update the outbound HTTP connection settings for an application-scoped configuration, use the following syntax:

IBM i

```
ConfigEngine.sh update-outbound-http-connection-config   -DConfigFileName=your_path/config_file.xml
                                                         -DApplicationScopeRef=scoperef
```

Linux

```
./ConfigEngine.sh update-outbound-http-connection-config -DConfigFileName=your_path/config_file.xml
                                                         -DApplicationScopeRef=scoperef
```

Windows

```
ConfigEngine.bat update-outbound-http-connection-config  -DConfigFileName=your_path\config_file.xml
                                                         -DApplicationScopeRef=scoperef
```

`your_path/config_file.xml` is the absolute path name of the XML document that contains the configuration settings for the outbound HTTP connection infrastructure that you want to update.

`scoperef` is the context root of the application to which the configuration settings are scoped. To obtain the scope reference, proceed as follows:

1. Access the WebSphere Integrated Solutions Console.
2. Select the enterprise application for which the outbound HTTP connection is scoped.
3. Click **View Deployment Descriptor**.
4. Locate the value of the `context root` tag.

An example context root is `/PA_Banner_Ad`.

Deleting outbound HTTP connection configuration settings:

This configuration task deletes configuration settings for an outbound HTTP connection. It deletes the settings that are specified in an XML document. You can delete settings of the global configuration or of an application-scoped configuration.

About this task

The task deletes the outbound HTTP connection configuration settings as specified in an XML document. You specify the settings that you want to delete in the XML document. The task tries to remove all items that are defined in the XML document. If an item contains dependent data that is not explicitly listed in this document, then the configuration setting is not deleted. The syntax of the configuration task differs, depending on whether you want to delete the settings for the global configuration or for an application-scoped configuration.

Deleting global outbound HTTP connection settings:
About this task

To delete global configuration settings for an outbound HTTP connection, use the following syntax:

IBM i

```
ConfigEngine.sh delete-outbound-http-connection-config -DConfigFileName=your_path/config_file.xml
```

Linux

```
./ConfigEngine.sh delete-outbound-http-connection-config -DConfigFileName=your_path/config_file.xml
```

Windows

```
ConfigEngine.bat delete-outbound-http-connection-config -DConfigFileName=your_path\config_file.xml
```

`your_path/config_file.xml` is the absolute path name of the XML document that contains the configuration settings for the outbound HTTP connection infrastructure that you want to delete.

Deleting application-scoped outbound HTTP connection settings:
About this task

To delete application-scoped configuration settings for an outbound HTTP connection, use the following syntax:

IBM i

```
ConfigEngine.sh delete-outbound-http-connection-config   -DConfigFileName=your_path/config_file.xml
                                                         -DApplicationScopeRef=scoperef
```

Linux

```
./ConfigEngine.sh delete-outbound-http-connection-config -DConfigFileName=your_path/config_file.xml
                                                         -DApplicationScopeRef=scoperef
```

Windows

```
ConfigEngine.bat delete-outbound-http-connection-config  -DConfigFileName=your_path\config_file.xml
                                                         -DApplicationScopeRef=scoperef
```

`your_path/config_file.xml` is the absolute path name of the XML document that contains the configuration settings for the outbound HTTP connection infrastructure that you want to delete.

`scoperef` is the context root of the application to which the configuration settings are scoped. To obtain the scope reference, proceed as follows:

1. Access the WebSphere Integrated Solutions Console.
2. Select the enterprise application for which the outbound HTTP connection is scoped.
3. Click **View Deployment Descriptor**.
4. Locate the value of the `context root` tag.

An example context root is `/PA_Banner_Ad`.

Clearing an outbound HTTP connection configuration profile:

This configuration task clears the complete configuration profile for an outbound HTTP connection. It deletes all settings of an outbound HTTP connection profile. Whereas the `delete-outbound-http-connection-config` configuration task deletes only those configuration settings that are listed in the specified XML document, this task purges the entire HTTP connection configuration profile.

About this task

The syntax of the configuration task differs, depending on whether you want to clear the settings for the global configuration or for an application-scoped configuration. If you specify no application scope reference, the task deletes the global outbound HTTP connections profile. Otherwise, the task deletes the

application-scoped profile that you identify by the `scoperef` parameter. The configuration task deletes all settings from that specified profile.

Clearing the global outbound HTTP connection profile:
About this task

To clear all settings of the global configuration for an outbound HTTP connection, use the following syntax:

IBM i

```
ConfigEngine.sh clean-outbound-http-connection-config -DOutboundProfileType=global
```

Linux

```
./ConfigEngine.sh clean-outbound-http-connection-config -DOutboundProfileType=global
```

Windows

```
ConfigEngine.bat clean-outbound-http-connection-config -DOutboundProfileType=global
```

Clearing an application-scoped outbound HTTP connection profile:
About this task

To clear all settings of an application-scoped configuration for an outbound HTTP connection, use the following syntax:

IBM i

```
ConfigEngine.sh clean-outbound-http-connection-config   -DOutboundProfileType=scoped
                                                        -DApplicationScopeRef=scoperef
```

Linux

```
./ConfigEngine.sh clean-outbound-http-connection-config -DOutboundProfileType=scoped
                                                        -DApplicationScopeRef=scoperef
```

Windows

```
ConfigEngine.bat clean-outbound-http-connection-config  -DOutboundProfileType=scoped
                                                        -DApplicationScopeRef=scoperef
```

`scoperef` is the context root of the application to which the configuration settings are scoped. To obtain the scope reference, proceed as follows:

1. Access the WebSphere Integrated Solutions Console.
2. Select the enterprise application for which the outbound HTTP connection is scoped.
3. Click **View Deployment Descriptor**.
4. Locate the value of the `context root` tag.

An example context root is `/PA_Banner_Ad`.

Configuring outbound HTTP connections by using the Model Controller SPI:

To query and administer the outbound HTTP connection, you can also use the model controller SPI.

About this task

The information that is given here provides a brief introduction only. For a complete reference about this SPI, read the outbound HTTP connection Javadoc.

1. "Obtaining the Model SPI" on page 3020
   You can obtain the home object for the outbound HTTP connection configuration model by using JNDI lookup.

2. "Obtaining the Controller SPI"
   The Outbound HTTP Connection configuration model SPI grants read access to the configuration only. You can make updates to the outbound HTTP connection configuration model by using the Controller SPI.
3. "Viewing the configuration settings of the outbound HTTP connection" on page 3021
   WebSphere Portal Express provides several interfaces that you can use to retrieve configuration settings: `OutboundConnectionProfile`, `PolicyMapping`, `PolicyRule`, `CookieRule`, and `PolicyVariable`.
4. "Modifying configuration settings of the outbound HTTP connection" on page 3022
   You can change configuration settings of the outbound HTTP connection by using the following interfaces: `ModifiableOutboundConnectionProfile`, `ModifiablePolicyMapping`, `ModifiablePolicyRule`, `ModifiableCookieRule`, and `ModifiablePolicyVariable`.
5. "Creating and deleting configuration settings of the outbound HTTP connection" on page 3022
   The controller object also provides methods for creating or deleting configuration settings.

Obtaining the Model SPI:

You can obtain the home object for the outbound HTTP connection configuration model by using JNDI lookup.

About this task

Here is a sample code snippet:

```
import com.ibm.portal.outbound.config.*;
...

javax.naming.Context ctx = new javax.naming.InitialContext();
OutboundConnectionModelHome home = (OutboundConnectionModelHome)
            ctx.lookup("portal:service/model/OutboundConnectionModel");
OutboundConnectionModel model = home.getOutboundConnectionModelProvider().
                                GetOutboundConnectionModel();
```

The code that calls the SPI must run in the context of the portal server.

Related concepts:

"Model SPI overview" on page 2858
Models provide information that is needed by WebSphere Portal Express to perform tasks such as content aggregation or building navigation to browse the aggregated content. The information that is aggregated is represented through models that can be accessed programmatically by using the Model SPI (read-only). The information of a model is usually persistent (stored in a database) but can also be transient (computed and stored only in memory). Models can be represented by using a tree structure (nodes have a parent-child relationship), a list structure, or a selection structure (a selected element in a tree structure).

Obtaining the Controller SPI:

The Outbound HTTP Connection configuration model SPI grants read access to the configuration only. You can make updates to the outbound HTTP connection configuration model by using the Controller SPI.

About this task

Here is a sample code snippet:

```
import com.ibm.portal.outbound.config.*;
...
OutboundConnectionModel model; // refer to the previous example for information
                               // about how to instantiate this model
...

javax.naming.Context ctx = new javax.naming.InitialContext();
OutboundConnectionModelControllerHome chome = (OutboundConnectionModelControllerHome)
    ctx.lookup("portal:service/model/OutboundConnectionModelController");
OutboundConnectionModelController ctrl =
    home.getOutboundConnectionModelControllerProvider().getOutboundConnectionModelController(model);
```

The code that calls the SPI must run in the context of the portal server. Note that access to the Controller SPI is only possible for users who have administrative rights.

Locating configuration items: You can access configuration items by using `Locator` interfaces that are provided by the Outbound Connection Model. The following locators are defined at the model SPI: `OutboundConnectionProfileLocator`, `PolicyMappingLocator`, `PolicyRuleLocator` `CookieRuleLocator`, and `PolicyVariableLocator`. The following example shows how to locate configuration items using various locators:

```
import com.ibm.portal.outbound.config.*;

OutboundConnectionModel model; // refer to previous examples, how to instantiate this model
...

// Locate the global outbound HTTP connection profile:
OutboundConnectionProfile profile = model.getProfileLocator().findGlobalProfile();


// Locate the mapping that is assigned with the context path "/proxy":
PolicyMapping mapping = model.getPolicyMappingLocator(profile).findByContextPath("/proxy")


// Locate a policy rule that resides in the selected mapping and has a the following URL pattern:
PolicyRule rule = model.getPolicyRuleLocator(mapping).findByUrlPattern("http://localhost:9092/*");


// Collect a list of cookie rules that reside in the selected policy rule and start with the
// cookie name "myCookie". The items are collected in the list "filteredCookies":

List<CookieRule> filteredCookies = new ArrayList<CookieRule>();
model.getCookieRuleLocator(rule).collectAllMatching(
   new PatternMatcher<CookieRule>() {
 public int getMatchValue (CookieRule candidate) {
    return (candidate.getCookieNames()[0].startsWith("myCookie")) ? 1 : 0;
 }
   },filteredCookies);

// Locate a policy variable by its name:
PolicyVariable pvar = model.getPolicyVariableLocator(globalProfile).findByName("the.policy.var");
```

Related concepts:

"Controller SPI " on page 2883
You can use the Controller SPI for portal administration. It allows you to modify portal resources. It enhances the read-only portal Model SPI by adding writable aspects.

Viewing the configuration settings of the outbound HTTP connection:

WebSphere Portal Express provides several interfaces that you can use to retrieve configuration settings: `OutboundConnectionProfile`, `PolicyMapping`, `PolicyRule`, `CookieRule`, and `PolicyVariable`.

About this task

For a complete reference of the functions that these interfaces provide, read the Javadoc API. The following code example shows how you can retrieve configuration settings of a policy variable:

```
PolicyVariable pvar;// refer to previous examples, how to instantiate this policy variable

System.out.println ("The policy variable is "+pvar.getName()+", the type is "+pvar.getType());
if (pvar.getType() == VariableType.Endpoint) {
   System.out.println("The value is "+pvar.getEndpointValue());
} else {
   System.out.println("The values are:\n");
   for (String svalue : pvar.getDynamicPolicyValues()) {
       System.out.println(svalue);
   }
}
```

Modifying configuration settings of the outbound HTTP connection:

You can change configuration settings of the outbound HTTP connection by using the following interfaces: `ModifiableOutboundConnectionProfile`, `ModifiablePolicyMapping`, `ModifiablePolicyRule`, `ModifiableCookieRule`, and `ModifiablePolicyVariable`.

About this task

You can access these interfaces by using the method `OutboundConnectionModelController.getModifiableNode()`. Changes that you make are not immediately applied at the data backend. You must confirm the changes. The following code example illustrates how you can modify configuration settings:

```
OutboundConnectionModel model; // For information about how to instantiate this model
                               // refer to the previous examples.
OutboundConnectionModelController ctrl; // For information about how to instantiate this
                                        // controller, refer to the previous examples.
PolicyRule rule;// Refer to previous examples, how to instantiate this policy rule
System.out.println("The original URL pattern of the PolicyRule is "+rule.getUrlPattern());

// Get the Modifiable.. interface for the selected Policy rule. Change the URL pattern setting.
ModifiablePolicyRule mrule = (ModifiablePolicyRule)ctrl.getModifiableNode(rule);
mrule.setUrlPattern("http://localhost:9091/*");

// Apply the changes.
ctrl.commit();
System.out.println("The URL pattern of the PolicyRule is now "+rule.getUrlPattern());
```

Creating and deleting configuration settings of the outbound HTTP connection:

The controller object also provides methods for creating or deleting configuration settings.

About this task

You can create new items by using the `CreationContext` interface. This interface provides the controller with the mandatory initialization data. The following creation contexts are available: `OutboundConnectionProfileCreationContext`, `PolicyMappingCreationContext`, `PolicyRuleCreationContext`, `CookieRuleCreationContext`, and `PolicyVariableCreationContext`. You can implement the creation contexts either by a custom application or by the singleton class `OutboundConnectionCreationContextBuidlerFactory`.

The following code example shows how you create a new policy rule, and how you delete another policy rule:

```
OutboundConnectionModel model; // Refer to previous examples, how to instantiate this model.
OutboundConnectionModelController ctrl; // Refer to previous example, how to instantiate this.
PolicyRule rule;// Refer to previous examples, how to instantiate this policy rule.

// This singleton can be used to produce CreationContext objects.
OutboundConnectionCreationContextBuilderFactory ccf =
     OutboundConnectionCreationContextBuilderFactory.getInstance();

// Create a new policy rule.
CreationContext cc = ccf.getPolicyRuleCreationContext(mapping, "rule2", "www.testme.com/test2*");
ModifiablePolicyRule mrule2 = (ModifiablePolicyRule) ctrl.create(ModifiablePolicyRule.class, cc);

// Delete the policy rule.
ctrl.delete(rule);

ctrl.commit(); // Apply the changes.
```

Sample administration tasks:

Here you find some procedures for running administration tasks on the outbound HTTP connection settings. These tasks use the portal configuration engine interface.

"Adding an outbound connection policy"
To add a policy to the outbound HTTP connection configuration, follow the procedure that is given here.

"Listing all available configuration profiles" on page 3026
To create a file that contains all profiles that are registered at the outbound HTTP connection configuration, follow the procedure that is given here.

"Modifying outbound HTTP connection policy settings" on page 3027
To modify the settings of an existing outbound HTTP connection policy, follow the procedure that is given here.

"Removing an outbound HTTP connection policy" on page 3028
To delete an outbound HTTP connection policy, you can use different options. Some of them are described here.

"Exporting a configuration profile to a file" on page 3029
To create a backup of a specific outbound HTTP connection profile, use the procedure given here.

"Importing a configuration profile from a file" on page 3030
To restore all policies and settings for a given application scoped profile, use the import procedure given here. For example, you can use this procedure to restore a specific connection profile configuration that you backed up previously.

Adding an outbound connection policy:

To add a policy to the outbound HTTP connection configuration, follow the procedure that is given here.

Procedure

1. Decide whether you want to create a global or application-scoped policy. You can place the policy in the global profile, or in an application-scoped profile:
   - Policies that are stored in the global profile are not linked to a specific application. For details about application scoping, see *Using the Ajax proxy in portlets*.
   - Policies that are stored in an application-scoped profile are visible in the scope of the application that is associated with the profile. The connection is effective only in the scope of that application, with the following consequences:

- If the remote connection is opened through the Ajax proxy, the caller must use the Ajax proxy URL that is registered in the web module. Application-scoped policies are not visible to the Ajax proxy servlet of IBM WebSphere Portal Express.
- If the remote connection is opened through the outbound connection service, the policy is only visible if the code runs in the web module for which the profile is scoped.

2. Decide the mapping for which you want the policy to apply. The policy mapping determines the entry point of the Ajax proxy. For example, the WebSphere Portal Express Ajax proxy servlet provides the following mapping points:
   - `/proxy` is for unauthenticated access. With this mapping point, an HTTP client can access the remote content without further authentication at the portal server.
   - `/myproxy` is for authenticated access. With this mapping point, the HTTP client must authenticate before it can access the remote resource through the Ajax proxy.

   Furthermore, you can also place the policy into the default mapping. Policies that are stored in the default mapping are accessible through all servlet entry points. Usually, policies for remote URLs that have security sensitive content are stored under the `/myproxy` mapping. Otherwise, it is good practice to store the policy in the default mapping. For more information, see *Policy Mappings*.
3. Open one of the following XML templates in a text editor. Here is a template for a policy in the `/myproxy` mapping:

```
<?xml version="1.0" encoding="UTF-8"?>
<proxy-rules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="http://www.ibm.com/xmlns/prod/sw/http/outbound/proxy-config/2.0">

   <variables>
      <dynamic-policy name="MY_ADMINISTRATIVE_NAME.url">
         <value>MY_URL_PATTERN</value>
         <!-- Step 5: add additional URL patterns here, if you want to -->
      </dynamic-policy>
   <variables>

   <mapping contextpath="/myproxy"  url="*">
      <policy name="MY_ADMINISTRATIVE_NAME" url="{$MY_ADMINISTRATIVE_NAME.url}" >
         <actions>
            <method>GET</method>
            <method>POST</method>
            <!-- Step 6: add or remove HTTP methods here -->
         </actions>

         <!-- uncomment the cookie-rule section if required
         <cookie-rule name="MY_ADMINISTRATIVE_NAME.cookies.1">
            <cookie>my_cookie_name</cookie>
            <scope>user</scope>
            <handling>store-in-session</handling>
         </cookie-rule>
         -->

         <!--Step 7: uncomment this section if you require an authenticated remote connection.
            Change the metadata setting according to the type of authentication
            that you want to established.

         <meta-data>
            <name>hpaa.authtype</name>
            <value>http-basic</value>
         </meta-data>
         <meta-data>
            <name>hpaa.slotid</name>
            <value>MY_ADMINISTRATIVE_NAME.credentials</value>
         </meta-data>
         <meta-data>
            <name>forward-credentials-from-vault</name>
            <value>true</value>
         </meta-data>
         -->
```

```
      </policy>
   </mapping>

</proxy-rules>
```

Here is a template for a policy in the default mapping:

```
<policy name="MY_ADMINISTRATIVE_NAME" url="{$MY_ADMINISTRATIVE_NAME.url}" >
   <actions>
      <method>GET</method>
      <method>POST</method>
      <!-- Step 6: add or remove HTTP methods here -->
   </actions>

   <!-- uncomment the cookie-rule section if required
   <cookie-rule name="MY_ADMINISTRATIVE_NAME.cookies.1">
      <cookie>my_cookie_name</cookie>
         <scope>user</scope>
         <handling>store-in-session</handling>
      </cookie-rule>
   -->

   <!--Step 7: uncomment this section if you require an authenticated remote connection.
   Change the metadata setting according to the type of authentication that should be
   established.

   <meta-data>
      <name>hpaa.authtype</name>
      <value>http-basic</value>
   </meta-data>

   <meta-data>
      <name>hpaa.slotid</name>
      <value>MY_ADMINISTRATIVE_NAME.credentials</value>
   </meta-data>

   <meta-data>
      <name>forward-credentials-from-vault</name>
      <value>true</value>
   </meta-data>
   -->
</policy>
```

You can apply all following steps to the templates given in this step.

4. Determine an administrative name for the policy. This administrative name helps administrators to identify the policy later. For example, you can use `policy_01`. Locate all occurrences of `MY_ADMINISTRATIVE_NAME` and replace this string with the administrative name that you determined (example: `policy_01`).
5. Define a URL pattern to which you want the policy to apply. Replace `MY_URL_PATTERN` with the URL pattern to which you want this policy to apply. For example, set the URL pattern to `http://http://www.ibm.com/us/en/*`.
6. Determine the HTTP methods for which you want to make proxy requests possible. In most cases, the HTTP methods GET and POST are required. If you require a different set of supported methods, you can manage the list in the `<actions>` list of the template. Manage the `<actions>` section according to your requirements.
7. Determine the authentication method that is required to establish the remote connection. The remote host might restrict the access to the URL. If you want the remote connection to be an authenticated connection, add the metadata settings that are required for the authentication type. Also, for certain authentications extra user credentials are referenced. Make sure that the Credential Vault contains the user credentials that are referenced in the authentication settings. For details, read *Authenticating outbound HTTP connections*.
8. Save the XML document to a file. For example, save the document to the file `/tmp/create_policy.xml`.
9. Deploy the policy at the configuration profile that you selected in an earlier step.

- If you want the policy to be active globally, start the following portal configuration engine task:
  - Linux :

    ```
    ./ConfigEngine.sh update-outbound-http-connection-config
                      -DConfigFileName=/tmp/create_policy.xml
    ```

  - IBM i:

    ```
    ConfigEngine.sh   update-outbound-http-connection-config
                      -DConfigFileName=/tmp/create_policy.xml
    ```

  - Windows:

    ```
    ConfigEngine.bat  update-outbound-http-connection-config
                      -DConfigFileName=/tmp/create_policy.xml
    ```

- If you want the policy to be active in the scope of a web module only, start the following portal configuration engine task. This example scopes the policy to the web module Banner AD:
  - Linux :

    ```
    ./ConfigEngine.sh update-outbound-http-connection-config
                      -DConfigFileName=/tmp/create_policy.xml
                      -DApplicationScopeRef=/PA_Banner_Ad
    ```

  - IBM i:

    ```
    ConfigEngine.sh   update-outbound-http-connection-config
                      -DConfigFileName=/tmp/create_policy.xml
                      -DApplicationScopeRef=/PA_Banner_Ad
    ```

  - Windows:

    ```
    ConfigEngine.bat  update-outbound-http-connection-config
                      -DConfigFileName=/tmp/create_policy.xml
                      -DApplicationScopeRef=/PA_Banner_Ad
    ```

The policy was now added to the configuration. It is ready to use.

Related tasks:

"Authenticating outbound HTTP connections" on page 3031
You can protect the access to the remote host by an authentication mechanism.

Related reference:

"Using the AJAX proxy in portlets" on page 2991
Here is how you use the AJAX proxy in portlets.

"Policy mappings" on page 2994
When an outbound HTTP connection is established through the outbound connection service, the caller can specify the mapping context in an optional parameter.

Listing all available configuration profiles:

To create a file that contains all profiles that are registered at the outbound HTTP connection configuration, follow the procedure that is given here.

Procedure

Start the following portal configuration engine task:

- Linux :

  ```
  ./ConfigEngine.sh list-outbound-http-connection-profiles
                    -DListFileName=/tmp/list.xml
  ```

- IBM i:

  ```
  ConfigEngine.sh   list-outbound-http-connection-profiles
                    -DListFileName=/tmp/list.xml
  ```

- Windows:

```
ConfigEngine.bat  list-outbound-http-connection-profiles
                  -DListFileName=/tmp/list.xml
```

The task returns a list of all installed profiles. Example:

```
profile: OutboundProfileType=system
profile: OutboundProfileType=global
profile: OutboundProfileType=scoped ApplicationScopeRef=PA_Search_Center
```

Modifying outbound HTTP connection policy settings:

To modify the settings of an existing outbound HTTP connection policy, follow the procedure that is given here.

Procedure

1. Export the configuration to a file:
   - If the policy resides in the global configuration, start the following portal configuration engine task:
     - Linux :

       ```
       ./ConfigEngine.sh  read-outbound-http-connection-config
                         -DConfigFileName=/tmp/the_global_configuration.xml
       ```

     - IBM i:

       ```
       ConfigEngine.sh    read-outbound-http-connection-config
                         -DConfigFileName=/tmp/the_global_configuration.xml
       ```

     - Windows:

       ```
       ConfigEngine.bat   read-outbound-http-connection-config
                         -DConfigFileName=/tmp/the_global_configuration.xml
       ```

   - If the policy resides in the application-scoped configuration, proceed as follows:
     a. Determine the name of the application scope. To obtain this name, start the procedure for listing all available configuration profiles.
     b. Start the following configuration engine task:
        - Linux :

          ```
          ./ConfigEngine.sh read-outbound-http-connection-config
                            -DConfigFileName=/tmp/the_scoped_configuration.xml
                            -DApplicationScopeRef=THE_APPLICATION_SCOPE
          ```

        - IBM i:

          ```
          ConfigEngine.sh   read-outbound-http-connection-config
                            -DConfigFileName=/tmp/the_scoped_configuration.xml
                            -DApplicationScopeRef=THE_APPLICATION_SCOPE
          ```

        - Windows:

          ```
          ConfigEngine.bat  read-outbound-http-connection-config
                            -DConfigFileName=/tmp/the_scoped_configuration.xml
                            -DApplicationScopeRef=THE_APPLICATION_SCOPE
          ```

        where *THE_APPLICATION_SCOPE* is the name of the application scope.
2. Edit either of the files `/tmp/the_global_configuration.xml` or `/tmp/the_scoped_configuration.xml` by using a text editor or an XML editor. The configuration file contains all policies, mappings, and metadata settings that are defined.
3. Apply the changes from the previous step at the outbound HTTP connection configuration:
   - If the profile was exported from the global configuration, start the following portal configuration engine task:

– Linux :

```
./ConfigEngine.sh update-outbound-http-connection-config
                  -DConfigFileName=/tmp/the_global_configuration.xml
```

– IBM i:

```
ConfigEngine.sh   update-outbound-http-connection-config
                  -DConfigFileName=/tmp/the_global_configuration.xml
```

– Windows:

```
ConfigEngine.bat  update-outbound-http-connection-config
                  -DConfigFileName=/tmp/the_global_configuration.xml
```

- If the profile was exported from an application-scoped configuration, start the following portal configuration engine task:
  – Linux :

```
./ConfigEngine.sh update-outbound-http-connection-config
                  -DConfigFileName=/tmp/the_scoped_configuration.xml
                  -DApplicationScopeRef=THE_APPLICATION_SCOPE
```

  – IBM i:

```
ConfigEngine.sh   update-outbound-http-connection-config
                  -DConfigFileName=/tmp/the_scoped_configuration.xml
                  -DApplicationScopeRef=THE_APPLICATION_SCOPE
```

  – Windows:

```
ConfigEngine.bat  update-outbound-http-connection-config
                  -DConfigFileName=/tmp/the_scoped_configuration.xml
                  -DApplicationScopeRef=THE_APPLICATION_SCOPE
```

  where *THE_APPLICATION_SCOPE* is the name of the application scope.

Removing an outbound HTTP connection policy:

To delete an outbound HTTP connection policy, you can use different options. Some of them are described here.

About this task

To remove an outbound HTTP connection policy, proceed as follows:

Procedure

1. Export the configuration from which you want to delete the policies to a file. For details, read *Exporting a configuration profile from a file*.
2. Edit the file by using a file editor or an XML editor.
3. Locate all policies that you want to delete and remove them from the file.
4. Restore the configuration by using the modified file. For details, read *Importing a configuration profile from a file*.

Removing all policies of a configuration profile:
About this task

To remove all policies from a configuration profile, proceed as follows:

Procedure

1. Determine the name of the application scope for which you want to delete its outbound HTTP Connection profile. To obtain this name, use the procedure under *Listing all available configuration profiles*.
2. Start the following portal configuration engine task:
   - Linux :

```
./ConfigEngine.sh clean-outbound-http-connection-config
                  -DApplicationScopeRef=THE_APPLICATION_SCOPE
```

- IBM i:

```
ConfigEngine.sh   clean-outbound-http-connection-config
                  -DApplicationScopeRef=THE_APPLICATION_SCOPE
```

- Windows:

```
ConfigEngine.bat  clean-outbound-http-connection-config
                  -DApplicationScopeRef=THE_APPLICATION_SCOPE
```

where THE_APPLICATION_SCOPE is the name of the application scope.

Note: To remove the global configuration, you can also use the configuration task clean-outbound-http-connection-config instead. However, it is good practise to review the current content of the global configuration profile before you delete its content. To do so, follow the procedure for *Removing an Outbound HTTP Connection Policy* earlier in this topic.

Exporting a configuration profile to a file:

To create a backup of a specific outbound HTTP connection profile, use the procedure given here.

Procedure

Export the configuration to a file.

- To export the global configuration, start the following portal configuration engine task:
  - Linux :

```
./ConfigEngine.sh read-outbound-http-connection-config
                  -DConfigFileName=/tmp/the_global_configuration.xml
```

  - IBM i:

```
ConfigEngine.sh   read-outbound-http-connection-config
                  -DConfigFileName=/tmp/the_global_configuration.xml
```

  - Windows:

```
ConfigEngine.bat  read-outbound-http-connection-config
                  -DConfigFileName=/tmp/the_global_configuration.xml
```

  The output file /tmp/the_global_configuration.xml contains an XML export of the global configuration profile.
- To run an application scoped profile, proceed as follows:
  1. Determine the name of the application scope. To get this name, follow the procedure under *Listing all available configuration profiles*.
  2. Start the following portal configuration engine task:
     - Linux :

```
./ConfigEngine.sh read-outbound-http-connection-config
                    -DConfigFileName=/tmp/the_scoped_configuration.xml
                    -DApplicationScopeRef=THE_APPLICATION_SCOPE
```

     - IBM i:

```
ConfigEngine.sh   read-outbound-http-connection-config
                  -DConfigFileName=/tmp/the_scoped_configuration.xml
                  -DApplicationScopeRef=THE_APPLICATION_SCOPE
```

     - Windows:

```
ConfigEngine.bat  read-outbound-http-connection-config
                  -DConfigFileName=/tmp/the_scoped_configuration.xml
                  -DApplicationScopeRef=THE_APPLICATION_SCOPE
```

where THE_APPLICATION_SCOPE is the name of the application scope.

The output file /tmp/the_scoped_configuration.xml contains an XML export of the application-scoped configuration profile.

Importing a configuration profile from a file:

To restore all policies and settings for a given application scoped profile, use the import procedure given here. For example, you can use this procedure to restore a specific connection profile configuration that you backed up previously.

About this task

The following procedure assumes that the configuration that you want to restore was saved in a file /tmp/configuration.xml .

Procedure

1. Optional: If you want to restore the global configuration, skip this step. Determine the name of the application scope for which you want to restore its outbound HTTP connection profile. To obtain this name, follow the procedure given under *Listing all available configuration profiles*.
2. Start the appropriate portal configuration engine task, depending on whether you want to restore the global configuration or an application-scoped configuration:
   - To restore the global configuration, run the following configuration engine tasks:
     - Linux :

       ```
       ./ConfigEngine.sh clean-outbound-http-connection-config
                         -DOutboundProfileType=global
       ./ConfigEngine.sh update-outbound-http-connection-config
                         -DConfigFileName=/tmp/configuration.xml
       ```

     - IBM i:

       ```
       ConfigEngine.sh   clean-outbound-http-connection-config
                         -DOutboundProfileType=global
       ConfigEngine.sh   update-outbound-http-connection-config
                         -DConfigFileName=/tmp/configuration.xml
       ```

     - Windows:

       ```
       ConfigEngine.bat  clean-outbound-http-connection-config
                         -DOutboundProfileType=global
       ConfigEngine.bat  update-outbound-http-connection-config
                         -DConfigFileName=/tmp/configuration.xml
       ```

   - To restore an application-scoped configuration, run the following configuration engine tasks:
     - Linux :

       ```
       ./ConfigEngine.sh clean-outbound-http-connection-config
                         -DapplicationScopeRef=THE_APPLICATION_SCOPE
       ./ConfigEngine.sh update-outbound-http-connection-config
                         -DapplicationScopeRef=THE_APPLICATION_SCOPE
                         -DConfigFileName=/tmp/configuration.xml
       ```

     - IBM i:

       ```
       ConfigEngine.sh   clean-outbound-http-connection-config
                         -DapplicationScopeRef=THE_APPLICATION_SCOPE
       ConfigEngine.sh   update-outbound-http-connection-config
                         -DapplicationScopeRef=THE_APPLICATION_SCOPE
                         -DConfigFileName=/tmp/configuration.xml
       ```

     - Windows:

```
ConfigEngine.bat  clean-outbound-http-connection-config
                  -DapplicationScopeRef=THE_APPLICATION_SCOPE
ConfigEngine.bat  update-outbound-http-connection-config
                  -DapplicationScopeRef=THE_APPLICATION_SCOPE
                  -DConfigFileName=/tmp/configuration.xml
```

- where THE_APPLICATION_SCOPE is the name of the application scope.

Authenticating outbound HTTP connections:

You can protect the access to the remote host by an authentication mechanism.

About this task

For several authentication mechanisms, the outbound HTTP connection infrastructure provides an internal authentication support. By this support, the outbound HTTP connection service can handle the authentication protocol that is run by built-in authentication handlers. The outbound HTTP connection service supports the following authentication types:

- HTTP basic authentication
- HTTP digest authentication
- SPNEGO authentication
- Single Sign-On (SSO) by using LTPA tokens
- Form-based authentication.

"Providing user credentials for authenticated connections" on page 3032
Several authentication handlers require that user credentials are presented in the authentication process. For example, the HTTP basic authentication handler requires such user credentials. Before the outbound connection can be used, these user credentials are set in the Credential Vault.

"Establishing a basic authenticated HTTP connection" on page 3032
To establish an HTTP connection with basic authentication, you enable an outbound connection policy for HTTP basic authentication.

"Establishing a digest authenticated HTTP connection" on page 3033
To establish an HTTP connection with digest authentication, you enable an outbound connection policy for HTTP digest authentication.

"Establishing a form-based authenticated HTTP connection" on page 3034
To establish an HTTP connection with form-based authentication, you enable an outbound connection policy for form-based authentication.

"Establishing SSO connections through LTPA token" on page 3035
To establish a Single Sign-On (SSO) connection through LTPA token, you enable an outbound connection policy for the SSO connection through LTPA token.

"Establishing SSO connections through SPNEGO token" on page 3036
To establish a Single Sign-On (SSO) connection through SPNEGO token, you enable an outbound connection policy for the SSO connection through SPNEGO token.

CF03 "Establishing SSO connections through SAML 2.0 tokens" on page 3036
It is possible to establish outbound HTTP connections to remote resources that are authenticated by using the SAML 2.0 protocol. Outbound HTTP connections take care of the communication with the Identity provider (IDP) to get an authenticated connection by using SAML tokens. The SAML Authentication handler uses the HTTP POST binding protocol of the SAML specification. Therefore, it is required that both the Identity Provider and the Service Provider support the HTTP POST binding, if they are used by Outbound HTTP Connections for authentication.

Providing user credentials for authenticated connections:

Several authentication handlers require that user credentials are presented in the authentication process. For example, the HTTP basic authentication handler requires such user credentials. Before the outbound connection can be used, these user credentials are set in the Credential Vault.

About this task

For details about setting and storing credentials in the Credential vault, read the information about *Credential vault*.

The outbound HTTP connection configuration references the credentials by the slot name.

The following example procedure creates a simple credential with user ID and password in the Credential Vault:

Procedure

1. Access WebSphere Portal Express as a portal administrator.
2. Click the **Administration menu** icon. Then, click **Access** > **Credential Vault**. The Credential Vault management portlet is shown.
3. Select **Add a Vault slot**. The window for creating a vault slot is shown.
4. Choose a Vault slot name, and select the slot and vault segment to where it belongs.
5. Set the Vault slot as **Shared**, and set the user ID and password for the remote user.

Results

You created a credential with user ID and password in the Credential Vault.

Related concepts:

"Credential Vault" on page 1523
The Credential Vault is a service that stores credentials that allow portlets to log in to applications outside the realm on behalf of the user. It manages multiple identities for portlets and users.

Establishing a basic authenticated HTTP connection:

To establish an HTTP connection with basic authentication, you enable an outbound connection policy for HTTP basic authentication.

About this task

The policy rule settings in the following example code snippet enable an outbound connection policy for HTTP basic authentication. The example assumes that the connection `http://the_remote_server.com/basic-auth-protected/` is a remote site that is protected by basic authentication.

```
<policy url="http://the_remote_server.com/basic-auth-protected/*"
     basic-auth-support="true">
     <meta-data>
          <name>hpaa.authtype</name>
          <value>http-basic</value>
     </meta-data>
     <meta-data>
```

```
          <name>hpaa.slotid</name>
          <value>OutboundConnectionCredentials</value>
     </meta-data>
     <meta-data>
          <name>forward-credentials-from-vault</name>
          <value>true</value>
     </meta-data>
</policy>
```

The meanings of the settings are as follows:

- The policy attribute `basic-auth-support` enables the authentication filter.
- The value of the metadata parameter `hpaa.authtype` specifies the authentication type as HTTP basic authentication.
- The value of `hpaa.slotid` specifies the slot ID of the credential vault. For instructions about how to provide the user credentials of the remote connection, read *Providing user credentials for authenticated connections*.
- The metadata setting `forward-credentials-from-vault` specifies that the credentials of the basic authentication are gathered from the Credential Vault.
- You can use the parameter `hpaa.piid` to specify the portlet instance ID for which the credential slot is defined. This parameter is optional.

Establishing a digest authenticated HTTP connection:

To establish an HTTP connection with digest authentication, you enable an outbound connection policy for HTTP digest authentication.

About this task

The settings in the following example code snippet enable an outbound connection policy for HTTP digest authentication. The example assumes that the connection `http://the_remote_server.com/digest-auth-protected/` is a remote site that is protected by digest authentication.

```
<policy url="http://the_remote_server.com/basic-auth-protected/*"
     digest-auth-support="true">
     <meta-data>
          <name>hpaa.authtype</name>
          <value>http-digest</value>
     </meta-data>
     <meta-data>
          <name>hpaa.slotid</name>
          <value>OutboundConnectionCredentials</value>
     </meta-data>
     <meta-data>
          <name>forward-credentials-from-vault</name>
          <value>true</value>
     </meta-data>
</policy>
```

- The policy attribute `basic-auth-support` enables the authentication filter.
- The value of the metadata parameter `hpaa.authtype` specifies the authentication type as HTTP digest authentication.
- The value of `hpaa.slotid` specifies the slot ID of the credential vault. For instructions about how to provide the user credentials of the remote connection, read *Providing user credentials for authenticated connections*.
- The metadata setting `forward-credentials-from-vault` specifies that the credentials of the digest authentication are gathered from the Credential Vault.
- You can use the parameter `hpaa.piid` to specify the portlet instance ID for which the credential slot is defined. This parameter is optional.

Establishing a form-based authenticated HTTP connection:

To establish an HTTP connection with form-based authentication, you enable an outbound connection policy for form-based authentication.

About this task

The settings in the following example code snippet enable an outbound connection policy for form-based authentication. The example assumes that the site http://*the_remote_server*.com/login-form-protected/ is protected by a form-based authentication. The form is submitted by an action URL https://*the_remote_server*.com/doLogin.php.

```
<policy url="http://the_remote_server.com/login-form-protected/*" >
     <meta-data>
          <name>hpaa.authtype</name>
          <value>form</value>
     </meta-data>
     <meta-data>
          <name>hpaa.slotid</name>
          <value>OutboundConnectionCredentials</value>
     </meta-data>
     <meta-data>
          <name>forward-credentials-from-vault</name>
          <value>true</value>
     </meta-data>
     <meta-data>
          <name>form-action-url</name>
          <value>https://the_remote_server.com/doLogin.php</value>
     </meta-data>
     <meta-data>
          <name>form-field-name-user</name>
          <value>user_id</value>
     </meta-data>
     <meta-data>
          <name>form-field-name-password</name>
          <value>user_id_password</value>
     </meta-data>
     <meta-data>
          <name>form-additional-fields</name>
          <value>param1=value1,param2=value2</value>
          </meta-data>
     <meta-data>
          <name>form-session-cookies</name>
          <value>sessioncookie1,sessioncookie2</value>
     </meta-data>
</policy>
```

Set the following metadata parameters as required:

hpaa.slotid
Use this parameter to specify the slot ID of the credential vault. For information about how to provide the user credentials of the remote connection, read *Providing user credentials for authenticated connections*.

forward-credentials-from-vault
Use this parameter to specify that the credentials of the form-based authentication are gathered from the Credential Vault.

form-action-url
Use this parameter to specify the URL to which the form data is submitted.

form-field-name-user
Use this parameter to specify the ID of the HTML <input> tag that contains the user ID. In the previous example, the specified value is user_id.

form-field-name-password
Use this parameter to specify the ID of the HTML <input> tag that contains the password. In the previous example, the specified value is user_id_password.

form-additional-fields
Use this parameter to specify the names of the additional HTML input elements that are present in the form. In the previous example, the specified elements are param1=value1 and param2=value2.

form-session-cookies
Use this parameter to specify the names of the session cookies that are received from the remote server as part of Set-Cookie headers when the form is submitted and the authentication is successful. In the previous example, the specified cookies are sessioncookie1 and sessioncookie2.

Establishing SSO connections through LTPA token:

To establish a Single Sign-On (SSO) connection through LTPA token, you enable an outbound connection policy for the SSO connection through LTPA token.

About this task

The settings in the following example code snippet enable an SSO outbound connection policy through LTPA token. The example assumes that the connection http://the_remote_server.com/sso-protected/ is a remote site that is protected by an LTPA token.

```
<policy url="http://the_remote_server.com/sso-protected/*"
     basic-auth-support="true">
     <meta-data>
          <name>hpaa.authtype</name>
          <value>ltpa</value>
     </meta-data>
     <meta-data>
          <name>hpaa.slotid</name>
          <value>OutboundConnectionCredentials</value>
     </meta-data>
     <meta-data>
          <name>forward-credentials-from-vault</name>
          <value>true</value>
     </meta-data>
</policy>
```

- The policy attribute basic-auth-support enables the authentication filter.
- The value of the metadata parameter hpaa.authtype specifies the authentication type as SSO authentication by using LTPA tokens.
- The value of hpaa.slotid specifies the slot ID of the credential vault. For instructions about how to provide the user credentials of the remote connection, read *Providing user credentials for authenticated connections*. The slot ID identifies the Credential Vault slot for the user subject for which the LTPA tokens are used.
- The metadata setting forward-credentials-from-vault specifies that the credentials of the SSO authentication are gathered from the Credential Vault.

Related tasks:

"Providing user credentials for authenticated connections" on page 3032
Several authentication handlers require that user credentials are presented in the authentication process. For example, the HTTP basic authentication handler requires such user credentials. Before the outbound connection can be used, these user credentials are set in the Credential Vault.

Establishing SSO connections through SPNEGO token:

To establish a Single Sign-On (SSO) connection through SPNEGO token, you enable an outbound connection policy for the SSO connection through SPNEGO token.

The settings in the following example code snippet enable an SSO outbound connection policy through SPNEGO token. The example assumes that the connection `http://the_remote_server.com/sso-protected/` is a remote site that is protected by an SPNEGO token.

```
<policy url="http://the_remote_server.com/sso-protected/*"
     basic-auth-support="true">
     <meta-data>
          <name>hpaa.authtype</name>
          <value>spnego</value>
     </meta-data>
</policy>
```

- The policy attribute `basic-auth-support` enables the authentication filter.
- The value of the metadata parameter `hpaa.authtype` specifies the authentication type as SSO authentication by using SPNEGO tokens.

Establishing SSO connections through SAML 2.0 tokens:

It is possible to establish outbound HTTP connections to remote resources that are authenticated by using the SAML 2.0 protocol. Outbound HTTP connections take care of the communication with the Identity provider (IDP) to get an authenticated connection by using SAML tokens. The SAML Authentication handler uses the HTTP POST binding protocol of the SAML specification. Therefore, it is required that both the Identity Provider and the Service Provider support the HTTP POST binding, if they are used by Outbound HTTP Connections for authentication.

An outbound HTTP connection is activated by setting the metadata **`SSO_SAML_20_IDP`** at the connection policy or policy mapping. The value of this metadata setting is a symbolic name for the Identity provider that establishes the connection. This name is used as a prefix for another set of metadata that define the settings of the Identity provider as the following example illustrates:

```
<mapping contextpath="/myproxy" url="*">
 <policy url="http://www.myremotesite.com/RESOURCE*" >
  ...
  <!-- the following meta data setting activates the connection        -->
  <!-- for SSO connections via SAML. The symbolic name of the Identity -->
  <!-- Provider is MySampleIdentityProvider                -->
  <meta-data>
   <name>SSO_SAML20_IDP</name>
   <value>MySampleIdentityProvider</value>
  </meta-data>
 </policy>

 <!-- a second policy that establishes a SSO connection via this IDP -->
 <policy url="http://www.myremotesite.com/ANOTHER*" > <!-- another policy -->
  ...
  <meta-data>
   <name>SSO_SAML20_IDP</name>
```

```
   <value>MySampleIdentityProvider</value>
  </meta-data>
 </policy>

 <!-- the settings of the Identity provider MySampleIdentityProvider -->
 <!-- In this  example, the identity provider settings are saved in the meta data -->
 <!-- is scoped to the policy mapping "/myportal".                                 -->
 <meta-data>
  <name>MySampleIdentityProvider.IDP_HOST</name>
  <value>www.the-identity-provider.com</value>
 </meta-data>
 <meta-data>
  <name>MySampleIdentityProvider.IDP_PARAM_NAME.1</name>
  <value>SAMLRquest</value>
 </meta-data>
 <meta-data>
  <name>MySampleIdentityProvider.IDP_PARAM_VALUE.1</name>
  <value>request</value>
 </meta-data>

</mapping>
```

CF03 "Configuration settings for SAML authenticated connections"
To enable a connection policy for SAML-based authentication, the following settings must be defined in the metadata section of the policy, the policy mapping, or the default mapping.

Configuration settings for SAML authenticated connections:

To enable a connection policy for SAML-based authentication, the following settings must be defined in the metadata section of the policy, the policy mapping, or the default mapping.

SSO_SAML20_IDP

The unique name of the Identity Provider, for example, *IdpName*. This metadata enables the policy for SAML-based authentication and is used to locate the Identity Provider settings. This metadata is required to support SAML-based authentication. Refer to the following example:

```
<meta-data>
 <name>SSO_SAML20_IDP</name>
 <value>IdpName</value>
</meta-data>
```

Important: For the remaining settings, *IdpName* refers to the name of the Identity Provider that you specified in the **SSO_SAML20_IDP** setting.

IdpName.IDP_HOST

The host name or IP address of the identity provider. This setting is required. Refer to the following example:

```
<meta-data>
 <name>IdpName.IDP_HOST</name>
 <value>www.mytfim.org</value>
</meta-data>
```

IdpName.IDP_PROTOCOL

This setting defines how the Identity Provider is connected and has two possible values, either `http` or `https`. Refer to the following example:

```
<meta-data>
 <name>IdpName.IDP_PROTOCOL</name>
 <value>https</value>
</meta-data>
```

IdpName.IDP_PORT

This setting defines the TCP port that is used for the Identity Provider connection. The default value is 80. Refer to the following example:

```
<meta-data>
 <name>IdpName.IDP_PORT</name>
 <value>9443</value>
</meta-data>
```

IdpName.IDP_URI

The URI of the Identity Provider service to which the SAML authentication is submitted. If this metadata setting is not defined, the connection uses the default URI `/SAML2/SSO/POST`. Refer to the following example:

```
<meta-data>
 <name>IdpName.IDP_URI</name>
 <value>/idp/saml20/post</value>
</meta-data>
```

IdpName.IDP_TIMEOUT

The timeout value of the connection to the Identity Provider. If this metadata setting is not defined, the connection timeout is 60 seconds. Refer to the following example:

```
<meta-data>
 <name>IdpName.IDP_TIMEOUT</name>
 <value>120</value> <!-- wait 2 minutes -->
</meta-data>
```

CF05 **IdpName.IDP_AUTH_TOKEN_SOURCE**

Optional parameter that determines from where the authentication tokens for the IDP are taken. The default value is `ltpa`. The SAML authentication protocol begins with a request to the Identity provider. This request contains an authentication token, used to identify the caller at the Identity provider. The **IdpName.IDP_AUTH_TOKEN_SOURCE** parameter determines where this authentication token is taken from. Currently, two values are enabled:

ltpa If the value `ltpa` is defined, then the Ajax proxy creates an LTPA token from the user subject of the Ajax proxy connection. This LTPA token is submitted to the Identity provider to authenticate the IDP request. For most authentication scenarios that are based on Tivoli Federated Identity Manager, the `ltpa` setting is the preferred one.

cookies

If the value `cookies` is defined, then the Ajax proxy uses authentication cookies from the local connection to authenticate the IDP request. The authentication cookie names are defined in the **IdpName.IDP_AUTH_TOKEN.n** metadata settings.

Refer to the following example:

```
<meta-data>
 <name>IdpName.IDP_AUTH_TOKEN_SOURCE</name>
 <value>cookies</value>
<! -- take cookie list from IDP_AUTH_TOKEN_COOKIE.n as authentication tokens -->
</meta-data>
```

IdpName.PARAM_NAME.n and IdpName.PARAM_VALUE.n

IdpName.PARAM_NAME.n is the name of a URL query parameter to the Identity Provider. Use this setting with **IdpName.PARAM_VALUE.n**, which defines the value of a URL query parameter to the Identity Provider. For both settings, *n* is a counter beginning with 1. Refer to the following example:

```
<meta-data>
 <name>IdpName.PARAM_NAME.1</name>
 <value>RequestBinding</value>
</meta-data>
<meta-data>
 <name>IdpName.PARAM_VALUE.1</name>
 <value>HTTPPost</value>
</meta-data>
```

IdpName.IDP_AUTH_COOKIE.n

The name of the authentication cookie, where *n* is a counter beginning with 1. If this metadata setting is not defined, the default authentication cookie is SAML20. Refer to the following example:

```
<meta-data>
 <name>IdpName.IDP_AUTH_COOKIE.1</name>
 <value>SAML20</value>
</meta-data>
<meta-data>
 <name>IdpName.IDP_AUTH_COOKIE.2</name>
 <value>another_cookie</value>
</meta-data>
```

CF05 **IdpName.IDP_AUTH_TOKEN_COOKIE.n**

The name of the authentication cookie that is used to authenticate against the Identity Provider to start the SAML authentication protocol. This metadata is only effective if the metadata **IdpName.IDP_AUTH_TOKEN_SOURCE** is set to cookies. Otherwise, the settings are ignored. The following example defines the authentication cookies MSISAuthenticated, MSISAuth, and MSISAuth1.

```
<meta-data>
 <name>IdpName.IDP_AUTH_TOKEN_COOKIE.1</name>
 <value>MSISAuthenticated</value>
</meta-data>
<meta-data>
 <name>IdpName.IDP_AUTH_TOKEN_COOKIE.2</name>
 <value>MSISAuth</value>
</meta-data>
<meta-data>
 <name>IdpName.IDP_AUTH_TOKEN_COOKIE.3</name>
 <value>MSISAuth1</value>
</meta-data>
```

CF03 "Configuration settings for Tivoli Federated Identity Manager (TFIM)"
Learn about establishing an SAML-based SSO connection for Tivoli Federated Identity Manager. Tivoli Federated Identity Manager must be installed and operational before an SSO connection can be established.

CF05 "Configuration settings for Active Directory Federation Services (ADFS)" on page 3048
Learn about establishing a single-sign on (SSO) connection for Active Directory Federation Services (ADFS).

Configuration settings for Tivoli Federated Identity Manager (TFIM):

Learn about establishing an SAML-based SSO connection for Tivoli Federated Identity Manager. Tivoli Federated Identity Manager must be installed and operational before an SSO connection can be established.

For more information about the administrative settings of Tivoli Federated Identity Manager, see *Creating a generic SAML 2.0 federation with a new or existing domain* in the related links.

CF03 “Finding the Identity Provider login URL and the Partner URL (TFIM)”
In order to establish an SSO connection through Tivoli Federated Identity Manager, you must specify the Identity Provider login URL and the Partner URL. Learn how to find these values from the Tivoli Federated Identity Manager configuration if you do not already know them. If you know these values already, skip this step.

CF03 “Creating Identity Provider settings at the Outbound Connection Service configuration (TFIM)” on page 3041
Certain metadata settings such as the Identity Provider URL and the Partner URL are required to use Tivoli Federated Identity Manager Identity Provider for SSO connections through SAML 2.0 authentication protocol.

CF03 “Defining policy rules for the remote connection (TFIM)” on page 3043
Learn about how to create a policy rule for the SSO connection. Creating a policy rule is required to use the SSO connection for the Identity Provider that you registered.

CF03 “Adding the Java Authentication and Authorization Service (JAAS) login module to the Tivoli Federated Identity Manager (TFIM) server” on page 3045
The Java Authentication and Authorization Service (JAAS) login module is available as a plug-in. This plug-in sets the email address of the logged in user within the security context so that the email address can be used within Tivoli Federated Identity Manager.

Related concepts:

CF03 “Establishing single sign-on (SSO) between the portal installation and IBM Connections in SmartCloud for Social Business” on page 763
Learn about the requirements that must be fulfilled in order to successfully integrate WebSphere Portal Express with IBM Connections in SmartCloud for Social Business. You can provide a user experience where a user logs in one time into WebSphere Portal Express and then automatically sees all integrated Connections information.

Related tasks:

CF03 “Configuring single sign-on (SSO) for backend calls to IBM Connections in SmartCloud for Social Business” on page 765
The Connections integration assets use a common infrastructure that is called HTTP Outbound to complete calls to the Connections backend server. Learn about the steps that are required to configure the HTTP outbound component to complete calls to IBM Connections in SmartCloud for Social Business.

Related information:

Creating a generic SAML 2.0 federation with a new or existing domain

Finding the Identity Provider login URL and the Partner URL (TFIM):

In order to establish an SSO connection through Tivoli Federated Identity Manager, you must specify the Identity Provider login URL and the Partner URL. Learn how to find these values from the Tivoli Federated Identity Manager configuration if you do not already know them. If you know these values already, skip this step.

About this task

The following values are needed from the Tivoli Federated Identity Manager configuration in order to establish an SSO connection through Tivoli Federated Identity Manager:

The Identity Provider login URL
The login URL of the Identity Provider service.

The Partner URL
The URL of the federation partner.

The following steps describe how to obtain the Identity Provider login URL and the Partner URL from the Tivoli Federated Identity Manager configuration by using the Tivoli Federated Identity Manager admin console.

Procedure

1. Log in to the WebSphere Integrated Solutions Console of the Tivoli Federated Identity Manager server.
2. Click **Tivoli Federated Identity Manager** > **Configure Federated Single Sign-on** > **Federations**. The panel shows a list of Identity Providers.
3. Select the Identity Provider that you want to use for the SSO connection.

   Note: This Identity Provider must have a single sign-on protocol **SAML 2.0** and the role of this federation must be **Identity Provider**.
4. Click **Properties...**.
5. Select the value of the field **Single Sign-on Service URL**. Use this value as the Identity Provider service URL for your Tivoli Federated Identity Manager configuration.
6. Click **Cancel**.
7. Click **View Partners...**.
8. Select the Partner that you want to use for the SAML connection.
9. Click **Properties...**.
10. Select the value of the field **Provider ID**. Use this value as the Partner URL for your Tivoli Federated Identity Manager configuration.
11. Click **Cancel**.

Creating Identity Provider settings at the Outbound Connection Service configuration (TFIM):

Certain metadata settings such as the Identity Provider URL and the Partner URL are required to use Tivoli Federated Identity Manager Identity Provider for SSO connections through SAML 2.0 authentication protocol.

About this task

The following metadata values are required to define a Tivoli Federated Identity Manager Identity Provider, where *IdpName* is the unique name of the Identity Provider:

***IdpName*.IDP_PROTOCOL**
The protocol part of the Identity Provider login URL. Replace the value *my_idp_prot* with the value of the protocol part of the Identity Provider login URL, either `http` or `https`.

***IdpName*.IDP_HOST**
The host name part of the Identity Provider login URL. Replace the value *my_idp_host* with the Identity Provider login URL host name.

***IdpName*.IDP_PORT**
The port number of the Identity Provider login URL. Replace the value *my_idp_port* with the value of the Identity provider login URL port.

***IdpName*.IDP_URI**
The URL path of the Identity Provider login URL. Replace the value *my_idp_uri* with the path of the Identity Provider login URL.

***IdpName*.IDP_AUTH_COOKIE.1 and *IdpName*.IDP_AUTH_COOKIE.2**
Authentication tokens that are created by the federation partner.

***IdpName*.PARAM_NAME.1 and *IdpName*.PARAM_VALUE.1**
This setting selects the SAML 2.0 binding. Specify `HTTPPost` as the value.

***IdpName*.PARAM_NAME.2 and *IdpName*.PARAM_VALUE.2**
The partner URL. Replace the value *my_partner_url* with the URL of the partner that runs the Service Provider service.

***IdpName*.PARAM_NAME.3 and *IdpName*.PARAM_VALUE.3**
Defines the format of the name ID field. Specify `Email` as the value.

The following `XML` example creates the Identity Provider settings for Tivoli Federated Identity Manager at the Outbound Connection Service Configuration and uses the following values:

- The Identity Provider name is `tfim101`.
- The Identity Provider login URL is `https://idp.example.com/sps/myfederation/saml20/login`.
- The Partner URL is `https://sp.example.com/sps/myfederation/saml20/v2_0`.

Change these values according to your configuration.

Procedure

1. Create an `XML` document like the following example:

```
<?xml version="1.0" encoding="UTF-8"?>
<proxy-rules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://www.ibm.com/xmlns/prod/sw/http/outbound/proxy-config/2.0">
 <variables>
  <!--  replace values with the IdP login URL and the partner URL -->
  <endpoint name="tfim101.idp_prot">https</endpoint>
  <endpoint name="tfim101.idp_host">idp.example.com</endpoint>
  <endpoint name="tfim101.idp_port">443</endpoint>
  <endpoint name="tfim101.idp_uri">/sps/myfederation/saml20/login</endpoint>
  <endpoint name="tfim101.partner_url">https://sp.example.com/sps/myfederation/saml20/v2_0</endpo
 </variables>
 <meta-data>
  <name>tfim101.IDP_PROTOCOL</name>
  <value>my_idp_prot</value>
 </meta-data>
 <meta-data>
  <name>tfim101.IDP_HOST</name>
  <value>my_idp_host</value>
 </meta-data>
 <meta-data>
  <name>tfim101.IDP_PORT</name>
  <value>my_idp_port</value>
 </meta-data>
 <meta-data>
  <name>tfim101.IDP_URI</name>
```

```
 <value>my_idp_uri</value>
</meta-data>
<meta-data>
 <name>tfim101.PARAM_NAME.1</name>
 <value>RequestBinding</value>
</meta-data>
<meta-data>
 <name>tfim101.PARAM_VALUE.1</name>
 <value>HTTPPost</value>
</meta-data>
<meta-data>
 <name>tfim101.PARAM_NAME.2</name>
 <value>PartnerId</value>
</meta-data>
<meta-data>
 <name>tfim101.PARAM_VALUE.2</name>
 <value>my_partner_url</value>
</meta-data>
<meta-data>
 <name>tfim101.PARAM_NAME.3</name>
 <value>NameIdFormat</value>
</meta-data>
<meta-data>
 <name>tfim101.PARAM_VALUE.3</name>
 <value>Email</value>
</metadata>
<meta-data>
 <name>tfim101.IDP_AUTH_COOKIE.1</name>
 <value>LtpaToken</value>
</meta-data>
<meta-data>
 <name>tfim101.IDP_AUTH_COOKIE.2</name>
 <value>LtpaToken2</value>
</meta-data>
</proxy-rules>
```

2. After you save the XML file, run the ConfigEngine task **update-outbound-http-connection-config** to apply the Identity Provider settings at the global configuration profile.
   - AIX, HP-UX, Linux, Solaris: `./ConfigEngine.sh update-outbound-http-connection-config -DConfigFileName=XML_file -DOutboundProfileType=global`
   - IBM i: `ConfigEngine.sh update-outbound-http-connection-config -DConfigFileName=XML_file -DOutboundProfileType=global`
   - Windows: `ConfigEngine.bat update-outbound-http-connection-config -DConfigFileName=XML_file -DOutboundProfileType=global`

   where *XML_file* is the file path to the XML file.

Defining policy rules for the remote connection (TFIM):

Learn about how to create a policy rule for the SSO connection. Creating a policy rule is required to use the SSO connection for the Identity Provider that you registered.

About this task

The metadata setting **SSO_SAML20_IDP** enables the policy for SSO authentication by using SAML. Replace the default value, in this case, *tfim101* with the unique Identity Provider name.

Procedure

1. To create a policy rule for a remote connection that uses a Tivoli Federated Identity Manager Identity Provider, create an XML document like the following example:

   Notes:

   - The following example creates a policy for an SSO connection to http://www.my_remote_site.com, which is controlled by the Tivoli Federated Identity Manager Identity Provider.
   - The example includes the optional definition of a cookie handling rule **store-in-session** for the authentication tokens LtpaToken and LtpaToken2. This setting saves the authentication tokens of the remote connection in the cookie store. When a URL to the remote site is requested again, the Outbound HTTP connection service establishes an authenticated HTTP connection by using the saved authentication tokens. Reestablishing the SAML authentication procedure is therefore not needed.

```
<?xml version="1.0" encoding="UTF-8"?>
<proxy-rules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="http://www.ibm.com/xmlns/prod/sw/http/outbound/proxy-config/2.0">

  <mapping contextpath="/myproxy" url="*">

   <policy name="pol_tfim101" url="http://www.my_remote_site.com/*" >
    <actions>
     <method>GET</method>
     <method>POST</method>
    </actions>
    <cookie-rule name="co_tfim101">
     <cookie>LtpaToken</cookie>
     <cookie>LtpaToken2</cookie>
     <scope>user</scope>
     <handling>store-in-session</handling>
    </cookie-rule>
    <meta-data>
     <name>SSO_SAML20_IDP</name>
     <value>tfim101</value>
    </meta-data>
   </policy>
  </mapping>
</proxy-rules>
```

2. After you save the XML file, run the ConfigEngine task **update-outbound-http-connection-config** to apply the policy settings to the configuration profile:
   - AIX, HP-UX, Linux, Solaris: `./ConfigEngine.sh update-outbound-http-connection-config -DConfigFileName=XML_file -DOutboundProfileType=global`
   - IBM i: `ConfigEngine.sh update-outbound-http-connection-config -DConfigFileName=XML_file -DOutboundProfileType=global`
   - Windows: `ConfigEngine.bat update-outbound-http-connection-config -DConfigFileName=XML_file -DOutboundProfileType=global`

   where *XML_file* is the file path to the XML file.

Adding the Java Authentication and Authorization Service (JAAS) login module to the Tivoli Federated Identity Manager (TFIM) server:

The Java Authentication and Authorization Service (JAAS) login module is available as a plug-in. This plug-in sets the email address of the logged in user within the security context so that the email address can be used within Tivoli Federated Identity Manager.

The local Tivoli Federated Identity Manager server creates SAML tickets to interact with SmartCloud for Social Business. You can identify the user in those tickets by their email address. The default mapping rules in Tivoli Federated Identity Manager enable access to attributes within the security context of a user. It is not possible to use the email address of the user that is currently logged in. Instead of custom programming, WebSphere Portal Express provides a Java Authentication and Authorization Service (JAAS) login module implementation that must be added to your Tivoli Federated Identity Manager system. The JAAS plug-in accesses the user's email address and inserts it into the Authorization Token, so the email address can be used within the standard mapping rules. Because the plug-in uses VMM API calls to obtain the email address, the Federated Repository type needs to be configured on your Tivoli Federated Identity Manager system. The following WebSphere Application Server JAAS login modules must be enabled before you enable the plug-in:

- **`com.ibm.ws.security.server.lm.ltpaLoginModule`**
- **`com.ibm.ws.security.server.lm.wsMapDefaultInboundLoginModule`**

The preceding modules are enabled by default.

If these prerequisites are not met in your environment, or you have another way of obtaining the email address that is not stored in your User Repository, you can implement your own JAAS plug-in by using the developerWorks guidelines. For more information, see the developerWorks article *Developing a custom Java module* in the related links.

CF03 "Configuring the Java Authentication and Authorization Service (JAAS) login module"
The behavior of the JAAS login module is configurable. If you change the attribute name for the security context, make sure to adjust the mapping rule accordingly.

CF03 "Tivoli Federated Identity Manager (TFIM) mapping for the Java Authentication and Authorization Service (JAAS) login module" on page 3047
By default, the JAAS plug-in reads a user's email address from the VMM attribute with the name `mail`. The JAAS plug-in sets the `mail` attribute in the security context. If you change the name of the attribute in the security context, update the following mapping rule accordingly.

Related information:

Developing a custom Java module

Configuring the Java Authentication and Authorization Service (JAAS) login module:

The behavior of the JAAS login module is configurable. If you change the attribute name for the security context, make sure to adjust the mapping rule accordingly.

Before you begin

- Verify that the following WebSphere Application Server JAAS login modules are enabled:
  - **com.ibm.ws.security.server.lm.ltpaLoginModule**
  - **com.ibm.ws.security.server.lm.wsMapDefaultInboundLoginModule**
- Copy the JAAS plug-in to the `AppServer\lib\ext` directory of your Tivoli Federated Identity Manager installation. If your Tivoli Federated Identity Manager is clustered, complete this step on the Deployment Manager and all cluster nodes. The JAAS plug-in file is named `wp.auth.jaas` and is available in the `PortalServer/base` directory.

About this task

Copying the JAAS plug-in to the `AppServer` directory prevents the plug-in from being updated during a portal installation update. If fixes are available, make sure to update the portal installation and then replace the existing `JAR` files in the Tivoli Federated Identity Manager installations with the new `JAR` files.

Procedure

- To configure the plug-in by using the default settings, complete the following steps:
  1. Open a wsadmin shell.
  2. Run the following command:

```
$AdminTask configureLoginModule { -loginType system
      -loginEntryAlias WEB_INBOUND -loginModule com.ibm.wps.auth.jaas.EnrichAttributeLoginModu
      -useLoginModuleProxy true -authStrategy OPTIONAL  -newModule true }
$AdminConfig save
```

  3. Exit wsadmin and restart the server.
- To configure the plug-in by using an alternative VMM attribute name for the email address, complete the following steps.
  1. Open a wsadmin shell.
  2. Run the following command:

```
$AdminTask configureLoginModule { -loginType system
      -loginEntryAlias WEB_INBOUND -loginModule com.ibm.wps.auth.jaas.EnrichAttributeLoginModu
      -useLoginModuleProxy true -authStrategy OPTIONAL  -customProperties
      {"vmm_email_attribute_name=internet_mail"} -newModule true }
$AdminConfig save
```

     Where *internet_mail* is the alternative VMM attribute name for the email address.
  3. Exit wsadmin and restart the server.
- To configure the plug-in by using an alternative attribute name for the security context, complete the following steps:
  1. Open a wsadmin shell.
  2. Run the following command:

```
$AdminTask configureLoginModule { -loginType system
      -loginEntryAlias WEB_INBOUND -loginModule com.ibm.wps.auth.jaas.EnrichAttributeLoginModu
      -useLoginModuleProxy true -authStrategy OPTIONAL  -customProperties
      {"context_email_attribute_name=email"} -newModule true }
$AdminConfig save
```

     Where *email* is the alternative attribute name for the security context.
  3. Exit wsadmin and restart the server.

- To configure the plug-in by using an alternative VMM attribute name for the email address and an alternative attribute name for the security context, complete the following steps:
  1. Open a wsadmin shell.
  2. Run the following command:

```
$AdminTask configureLoginModule
        { -loginType system -loginEntryAlias WEB_INBOUND -loginModule
        com.ibm.wps.auth.jaas.EnrichAttributeLoginModule -useLoginModuleProxy true -authStra
        OPTIONAL  -customProperties {"vmm_email_attribute_name=internet_mail",
        "context_email_attribute_name=email"} -newModule true }
$AdminConfig save
```

     Where *internet_mail* is the alternative VMM attribute name for the email address and *email* is the alternative attribute name for the security context.
  3. Exit wsadmin and restart the server.
- To remove the plug-in, complete the following steps:
  1. Open a wsadmin shell.
  2. Run the following command:

```
$AdminTask
        unconfigureLoginModule { -loginType application -loginEntryAlias Portal_LTPA -loginM
        com.ibm.wps.auth.jaas.EnrichAttributeLoginModule }
$AdminConfig save
```

  3. Exit wsadmin and restart the server.

Tivoli Federated Identity Manager (TFIM) mapping for the Java Authentication and Authorization Service (JAAS) login module:

By default, the JAAS plug-in reads a user's email address from the VMM attribute with the name `mail`. The JAAS plug-in sets the `mail` attribute in the security context. If you change the name of the attribute in the security context, update the following mapping rule accordingly.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:xalan="http://xml.apache.org/xalan"
version="1.0"
xmlns:mapping-ext="com.tivoli.am.fim.trustserver.sts.utilities.IDMappingExtUtils"
extension-element-prefixes="mapping-ext"
xmlns:stsuuser="urn:ibm:names:ITFIM:1.0:stsuuser">
        <xsl:strip-space elements="*" />
        <xsl:output method="xml" version="1.0" encoding="utf-8" indent="yes" />
        <!-- Initially we start with a copy of the document. -->
        <xsl:template match="@* | node()">
            <xsl:copy>
               <xsl:apply-templates select="@* | node()" />
            </xsl:copy>
        </xsl:template>

        <!-- This will replace the principal name with the user's email. -->
        <xsl:template
        match="//stsuuser:Principal/stsuuser:Attribute[@name='name']">
            <stsuuser:Attribute name="name" type="urn:oasis:names:tc:SAML:1.1:nameid-format:emailA
            <stsuuser:Value>
               <xsl:value-of select="//stsuuser:AttributeList/stsuuser:Attribute[@name='mail']/sts
            </stsuuser:Value>
            </stsuuser:Attribute>
        </xsl:template>
</xsl:stylesheet>
```

Configuration settings for Active Directory Federation Services (ADFS):

Learn about establishing a single-sign on (SSO) connection for Active Directory Federation Services (ADFS).

For more information about the administrative settings of SAML-based authentications, see *Creating a generic SAML 2.0 federation with a new or existing domain* in the related links.

CF05 "Finding the Identity Provider login URL and the Partner URL (ADFS)" on page 3049
To establish a single sign-on (SSO) connection through Active Directory Federation Services (ADFS), you must specify the Identity Provider login URL and the Partner URL. Learn how to find these values from the ADFS configuration if you do not already know them. If you know these values already, skip this step.

CF05 "Adding cookie handling to the Active Directory Federation Services (ADFS) server" on page 3049
The Internet Information Services (IIS) server as a part of the ADFS configuration sets up the ADFS cookies by default on a specific path and a specific host. To use these cookies for single sign-on (SSO) between the portal server and the ADFS server, the cookies need to flow on requests to the portal server as well. The cookie domain and cookie path must be changed.

CF05 "Creating Identity Provider settings at the Outbound Connection Service configuration (ADFS)" on page 3050
Certain metadata settings such as the ADFS cookies, the Identity Provider URL, and the Partner URL are required to define a single sign-on (SSO) connection through Active Directory Federation Services (ADFS).

CF05 "Defining policy rules for the remote connection (ADFS)" on page 3052
Learn about how to create a policy rule for the SSO connection. Creating a policy rule is required to use the SSO connection for the Identity Provider that you registered.

Related concepts:

CF03 "Establishing single sign-on (SSO) between the portal installation and IBM Connections in SmartCloud for Social Business" on page 763
Learn about the requirements that must be fulfilled in order to successfully integrate WebSphere Portal Express with IBM Connections in SmartCloud for Social Business. You can provide a user experience where a user logs in one time into WebSphere Portal Express and then automatically sees all integrated Connections information.

Related tasks:

CF03 "Configuring single sign-on (SSO) for backend calls to IBM Connections in SmartCloud for Social Business" on page 765
The Connections integration assets use a common infrastructure that is called HTTP Outbound to complete calls to the Connections backend server. Learn about the steps that are required to configure the HTTP outbound component to complete calls to IBM Connections in SmartCloud for Social Business.

Related information:

Creating a generic SAML 2.0 federation with a new or existing domain

Finding the Identity Provider login URL and the Partner URL (ADFS):

To establish a single sign-on (SSO) connection through Active Directory Federation Services (ADFS), you must specify the Identity Provider login URL and the Partner URL. Learn how to find these values from the ADFS configuration if you do not already know them. If you know these values already, skip this step.

About this task

The following steps describe how to obtain the Identity Provider target URL to directly initiate an IDP login flow on ADFS to a specific partner.

Procedure

1. Open the ADFS Management on the ADFS server.
2. Go to **Relying Party Trusts** and select the target partner.
3. Click **Properties...**
4. Click the **Identifiers** tab and copy the **Relying party identifier**.
5. Using a browser, log in to the web interface of the ADFS server that is provided by Internet Information Services (IIS). For example, go to the following URL: `https://<host>:<port>/adfs/ls/IdpInitiatedSignOn.aspx?loginToRp=<partnerUrl>`, where *<partnerUrl>* is the value of the **Relying party identifier** that you copied in the previous step.

Results

The federation flow starts automatically after you log in to the IIS-provided web interface of the ADFS server.

Adding cookie handling to the Active Directory Federation Services (ADFS) server:

The Internet Information Services (IIS) server as a part of the ADFS configuration sets up the ADFS cookies by default on a specific path and a specific host. To use these cookies for single sign-on (SSO) between the portal server and the ADFS server, the cookies need to flow on requests to the portal server as well. The cookie domain and cookie path must be changed.

Procedure

- To change the cookie domain, open the `web.CONF` of the IIS ADFS module and add the following:

```
<configuration>

...

<system.web>

...

<httpCookies domain="your_domain" httpOnlyCookies="false" requireSSL="false" />

...
```

- To change the cookie path, an **outboundRule** on IIS is needed. To support this **outboundRule** via the IIS Management console, Application Request Routing (ARR) is needed. This enhancement creates an **outboundRule** like the following example:

```
<rewrite>
<outboundRules>
<remove name="ChangeCookiePath" />
<rule name="ChangeCookiePath">
<match serverVariable="RESPONSE_Set_Cookie" pattern="^(.*; path=/)adfs/ls" />
<conditions />
<action type="Rewrite" value="{R:1}" />
</rule>
</outboundRules>
</rewrite>
```

Creating Identity Provider settings at the Outbound Connection Service configuration (ADFS):

Certain metadata settings such as the ADFS cookies, the Identity Provider URL, and the Partner URL are required to define a single sign-on (SSO) connection through Active Directory Federation Services (ADFS).

About this task

The following metadata values are required to define an ADFS Identity Provider, where *IdpName* is the unique name of the Identity Provider:

***IdpName*.IDP_PROTOCOL**
: The protocol part of the Identity Provider login URL. Replace the value *my_idp_prot* with the value of the protocol part of the Identity Provider login URL, either `http` or `https`.

***IdpName*.IDP_HOST**
: The host name part of the Identity Provider login URL. Replace the value *my_idp_host* with the Identity Provider login URL host name.

***IdpName*.IDP_PORT**
: The port number of the Identity Provider login URL. Replace the value *my_idp_port* with the value of the Identity provider login URL port.

***IdpName*.IDP_URI**
: The URL path of the Identity Provider login URL. Replace the value *my_idp_uri* with the path of the Identity Provider login URL.

***IdpName*.IDP_AUTH_COOKIE.1**
: Authentication token that is created by the federation partner. The default value is `SamlSession`.

***IdpName*.IDP_AUTH_TOKEN_SOURCE**
: Determines where the AJAX proxy obtains the IDP authentication tokens. The IDP authentication tokens are the cookies that are required to authenticate the connection with the Identity Provider. By default, the AJAX procy creates an LTPA token from the user subject and uses this LTPA token to authenticate the connection with the Identity Provider. Because the ADFS Identity Provider does not support an LTPA-based authentication, set the value of this parameter to `cookies`. This setting lets the AJAX proxy use the authentication tokens that are defined in the metadata settings **`IdpName.IDP_AUTH_TOKEN_COOKIE.n`**.

***IdpName*.IDP_AUTH_TOKEN_COOKIE.n**
: The authentication tokens that are required for authenticating against the Identity Provider. In the example that follows, the cookies *`MSISAuth`*, *`MSISAuth1`*, and *`MSISAuthenticated`* are defined.

***IdpName*.PARAM_NAME.1 and *IdpName*.PARAM_VALUE.1**
This setting defines the partner URL. Replace the value *idp_name_partner_url* with the URL of the partner that runs the Service Provider service.

The following XML example creates the Identity Provider settings for an Active Directory Federation Services connection at the Outbound Connection Service Configuration and uses the following values:

- The Identity Provider name is *adfs01*.
- The Identity Provider login URL is *https://idp.example.com/sps/myfederation/saml20/login*.
- The Partner URL is *https://sp.example.com/sps/myfederation/saml20/v2_0*.
- The ADFS authentication cookies are *MSISAuth*, *MSISAuth1*, and *MSISAuthenticated*.

Change these values according to your configuration.

Procedure

1. Create an XML document like the following example.

```
<?xml version="1.0" encoding="UTF-8"?>
<proxy-rules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://www.ibm.com/xmlns/prod/sw/http/outbound/proxy-config/2.0
 <variables>
  <!--  replace values with the IdP login URL and the partner URL -->
  <endpoint name="adfs01.idp_prot">https</endpoint>
  <endpoint name="adfs01.idp_host">idp.example.com</endpoint>
  <endpoint name="adfs01.idp_port">443</endpoint>
  <endpoint name="adfs01.idp_uri">/adfs/ls/IdpInitiatedSignOn.asp</endpoint>
  <endpoint name="adfs01.partner_url">https://sp.example.com/sps/myfederation/saml20/v2_0</end
 </variables>
 <meta-data>
  <name>adfs01.IDP_PROTOCOL</name>
  <value>my_idp_prot</value>
 </meta-data>
 <meta-data>
  <name>adfs01.IDP_HOST</name>
  <value>my_idp_host</value>
 </meta-data>
 <meta-data>
  <name>adfs01.IDP_PORT</name>
  <value>my_idp_port</value>
 </meta-data>
 <meta-data>
  <name>adfs01.IDP_URI</name>
  <value>my_idp_uri</value>
 </meta-data>
 <meta-data>
  <name>adfs01.PARAM_NAME.1</name>
  <value>LoginToRp</value>
 </meta-data>
 <meta-data>
  <name>adfs01.PARAM_VALUE.1</name>
  <value>idp_name_partner_url</value>
 </meta-data>
 <meta-data>
  <name>adfs01.IDP_AUTH_TOKEN_SOURCE</name>
  <value>cookies</value>
 </meta-data>
 <meta-data>
  <name>adfs01.IDP_AUTH_TOKEN_COOKIE.1</name>
  <value>MSISAuth</value>
 </meta-data>
```

```
 <meta-data>
  <name>adfs01.IDP_AUTH_TOKEN_COOKIE.2</name>
  <value>MSISAuth1</value>
 </meta-data>
 <meta-data>
  <name>adfs01.IDP_AUTH_TOKEN_COOKIE.3</name>
  <value>MSISAuthenticated</value>
 </metadata>
 <meta-data>
  <name>adfs01.IDP_AUTH_COOKIE.1</name>
  <value>SamlSession</value>
 </meta-data>
</proxy-rules>
```

2. After you save the XML file, run the ConfigEngine task **update-outbound-http-connection-config** to apply the Identity Provider settings at the global configuration profile.
   - AIX, HP-UX, Linux, Solaris: `./ConfigEngine.sh update-outbound-http-connection-config -DConfigFileName=XML_file -DOutboundProfileType=global`
   - IBM i: `ConfigEngine.sh update-outbound-http-connection-config -DConfigFileName=XML_file -DOutboundProfileType=global`
   - Windows: `ConfigEngine.bat update-outbound-http-connection-config -DConfigFileName=XML_file -DOutboundProfileType=global`

   where *XML_file* is the file path to the XML file.

Defining policy rules for the remote connection (ADFS):

Learn about how to create a policy rule for the SSO connection. Creating a policy rule is required to use the SSO connection for the Identity Provider that you registered.

About this task

The metadata setting **SSO_SAML20_IDP** enables the policy for SSO authentication by using SAML. Replace the default value, in this case, *adfs01* with the unique Identity Provider name.

Procedure

1. To create a policy rule for a remote connection that uses an Active Directory Federation Services (ADFS) Identity Provider, create an XML document like the following example:

   Note: The following example creates a policy for an SSO connection to http://www.my_remote_site.com, which is controlled by ADFS.

```
<?xml version="1.0" encoding="UTF-8"?>
<proxy-rules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="http://www.ibm.com/xmlns/prod/sw/http/outbound/proxy-config/2.0">

  <mapping contextpath="/myproxy" url="*">

   <policy name="pol_adfs01" url="http://www.my_remote_site.com/*" >
    <actions>
     <method>GET</method>
     <method>POST</method>
    </actions>
    <cookie-rule name="co_adfs01">
     <cookie>LtpaToken</cookie>
     <cookie>LtpaToken2</cookie>
     <scope>user</scope>
```

```
        <handling>store-in-session</handling>
       </cookie-rule>
       <meta-data>
        <name>SSO_SAML20_IDP</name>
        <value>adfs01</value>
       </meta-data>
      </policy>
     </mapping>
    </proxy-rules>
```

2. After you save the XML file, run the ConfigEngine task **update-outbound-http-connection-config** to apply the policy settings to the configuration profile:
   - AIX, HP-UX, Linux, Solaris: `./ConfigEngine.sh update-outbound-http-connection-config -DConfigFileName=XML_file -DOutboundProfileType=global`
   - IBM i: `ConfigEngine.sh update-outbound-http-connection-config -DConfigFileName=XML_file -DOutboundProfileType=global`
   - Windows: `ConfigEngine.bat update-outbound-http-connection-config -DConfigFileName=XML_file -DOutboundProfileType=global`

   Where *XML_file* is the file path to the XML file.

Using dynamic elements in outbound HTTP connection settings:

In some cases, it is useful to control configuration settings at run time. For example, an administrator might want to have the program decide at run time which policy rule is applied. In another scenario, parts of a policy rule configuration that are known only at run time must be included in the configuration.

About this task

Enabling and disabling policy rules: Administrators can use the metadata parameter `active` to enable or disable the policy rule to which the metadata belongs. Instead of setting a static metadata value to `true` or `false`, the administrator can set a variable token. The following example uses this feature to disable policy mappings. For more details, read the information about *deactivate policy rules, mappings, or custom connection filters* under *Description of the outbound HTTP connection configuration script*.

Example

The following example deactivates multiple policy rules, depending on a single policy variable:

```
<proxy-rules ...>
     <variables>
          <endpoint name="am_i_active">false</endpoint>
     <variables>
     <mapping contextPath="/theproxy">
          <policy url="http://the_remote_server.com/*">
               ...
               <meta-data>
                    <name>active</name>
                    <value>{$am_i_active}</value>
               </meta-data>
          </policy>
          <policy url="http://another_remote_server.com/*">
               ...
               <meta-data>
                    <name>active</name>
```

```
                    <value>{$am_i_active}</value>
                </meta-data>
            </policy>
        </mapping>

</policy-rules>
```

In some cases, the decision whether the policy rules is active or inactive can be made only at run time. In this case, the policy variable must be passed to the Ajax proxy as a URL parameter. The following example shows the configuration:

```
<proxy-rules ...>
     <mapping contextPath="/theproxy">
          <policy url="http://the_remote_server.com/*">
               ...
               <meta-data>
                    <name>active</name>
                    <value>{$active_a}</value>
               </meta-data>
          </policy>
          <policy url="http://the_remote_server.com/*">
               ...
               <meta-data>
                    <name>active</name>
                    <value>{$active_b}</value>
               </meta-data>
          </policy>
     </mapping>
</policy-rules>
```

The following example activates the first rule and deactivates the second rule:

```
http://localhost/wps/theproxy/http/the_remote_server.com/main.html?active_a=true&active_b=false
```

Using programmatic extensions for outbound HTTP connections:

To extend the functions of an outbound connection, portal administrators can implement a custom outbound service filter.

About this task

For example, such access can be required in the following cases:

- A portal administrator wants to open an outbound HTTP connection to a back end system that requires a specific HTTP request header setting.
- A portal administrator wants to feed a third-party site analysis program with information about an outbound HTTP connection. In this case, the outbound connections service calls custom code whenever the outbound connection is opened to the specific remote site. The custom code feeds the third-party site analysis program with the information about the remote connection.

  "Custom outbound service filters" on page 3055
  Custom outbound service filters provide a plug point for application developers. They can use these filters to add custom function to the processing logic of a policy.

  "Using custom cookie transformation handlers" on page 3059
  You can use custom cookie transformation handlers to allow custom program extensions to get programmatic control over the outbound HTTP connection.

Custom outbound service filters:

Custom outbound service filters provide a plug point for application developers. They can use these filters to add custom function to the processing logic of a policy.

"Implementing a custom outbound connection service filter"
Application developers can implement a custom outbound connection service filter.

"Registering a custom outbound connection service filter" on page 3056
For a custom outbound connection filter to take effect, you must register it.

"Configuration settings for custom outbound connection filters" on page 3057
The configuration of a custom outbound connection filter definition includes a metadata section. Developers can use it as a store for filter-specific customization data.

"Built-in configuration settings for custom outbound connection filters" on page 3059
The following built-in metadata setting for custom connection filter is defined.

Implementing a custom outbound connection service filter:

Application developers can implement a custom outbound connection service filter.

About this task

Custom outbound connection service filters must implement the interface `com.ibm.mashups.proxy.ext.OutboundServiceFilter`. They are instantiated from a factory class `com.ibm.mashups.proxy.ext.OutboundServiceFilterFactory`.

The following listing is an example of a custom outbound service filter:

```
package sample.test.filters;

import com.ibm.mashups.proxy.Context;
import com.ibm.mashups.proxy.configuration.AjaxProxyConfiguration;
import com.ibm.mashups.proxy.ext.ConnectionStatusEvent;
import com.ibm.mashups.proxy.ext.FlowControl;
import com.ibm.mashups.proxy.ext.OutboundServiceFilter;
import com.ibm.mashups.proxy.ext.OutboundServiceFilterFactory;

public class MyFilterFactory implements OutboundServiceFilterFactory {

    /**
      * This method is called whenever an outbound HTTP conenction
      * is opened for a url for which this custom filter
      * is defined in its configuration settings.
      * The method receives the URL of the outbound connection,
      * the proxy configuration, and the runtime context.
      * The method must return a filter object that implements the
      * OutboundServiceFilter interface.
      */
      @Override
      public OutboundServiceFilter newOutboundServiceFilter(URL url,
                        AjaxProxyConfiguration proxyConfig,
                        Map<String,String> metadata,
                        Context runtimeCtx) throws IOException {
            return new MyCustomFilter();
      }
}

public class MyCustomFilter implements OutboundServiceFilter {
     /*
```

```
 * This method is called for outbound HTTP connections
 * for which this filter has been configured.
 * The method receives the HttpURLConnection object
 * that connects against the remote server.
 * The method is called immediately before the connect() method
 * is called against that HttpURLConnection object. This means that
 * the implementation of this method can retrieve or modify the setup
 * parameters and the general request properties of this connection objects.
 * The filter can abend the connection, by invoking errorHandler.setError()
 * or errorHandler.setRedirect().
 */
 @Override
 public void preConnect(HttpURLConnection urlConnection,
                        FlowControl errorHandler)
 {
     // implement your custom code here.
 }
/*
 * This method is called for outbound HTTP connections for that this
 * filter has been configured.
 * The method receives the HttpURLConnection object that has connected
 * against the remote server.
 * The method is called immediately after connect() method is called
 * against that HttpURLConnection object, which means, the implementation
 * of this method is allowed to retrieve or modify the response header
 * fields and the remote content.
 * The filter can abend the connection, by invoking errorHandler.setError()
 *or errorHandler.setRedirect().
 */
 @Override
 public void postConnect(HttpURLConnection urlConnection,
              Map<String, List<String>> modifiableResponseHeaders,
               FlowControl errorHandler) {

     // implement your custom code here.
 }

 /**
  * This method is called whenever the outbound connection service ran
  * into an error. The method receives an eventData object that
  * contains information about the problem that occurred, such as the name,
  * the id and the parameter array of a resource bundle that contains the
  * error message which describes the problem, as well as information about
  * the failing http connection.
  * The filter can use this method to override the error handling, by
  * invoking  by invoking errorHandler.setError() or
  * errorHandler.setRedirect(). For example, the error handler can
  * override the error message or override the HTTP status of the
  * outbound connection.
  */
 @Override
 public void handleError(ConnectionStatusEvent eventData,
                   FlowControl errorHandler) {
     // implement your custom code here.
 }
```

Make sure that the filter class can be loaded at run time. For example, place the class file in the `/WEB-INF/lib` directory of the web module that contains the code that establishes the outbound HTTP connection.

Registering a custom outbound connection service filter:

For a custom outbound connection filter to take effect, you must register it.

About this task

Custom connection filters for outbound HTTP connections are defined at a policy setting. The following XML snippet shows how you register a custom connection filter at a policy setting:

```
<policy url="http://the_remote_server.com/path/*" >
    <actions>
        <method>GET</method>
    </actions>
    <filter-chain>
        <filter-factory>
            <classname>sample.test.filters.MyFilterFactory
            </classname>
        </filter-factory>
    </filter-chain>
</policy>
```

The `<filter-chain>` section contains a list of class names that denote the factory class for custom HTTP connection filters. The outbound connection service loads and runs the filters in the order in which they appear in the configuration.

Configuration settings for custom outbound connection filters:

The configuration of a custom outbound connection filter definition includes a metadata section. Developers can use it as a store for filter-specific customization data.

Example: A developer implemented a custom filter named `MyUserAgentRedirectionFilter`. The developer uses the filter to cause a redirect to a different URL if the user agent matches with a user agent pattern. You must define the user agent pattern and the redirection URL in the configuration. The following code snippet illustrates how the filter factory fetches settings for the redirection URL and the user agent patterns from the configuration. The sample code snippet assumes that the configuration setting for the user agent pattern is in a metadata setting with the name `my_user_agent`, and the redirection URL is in a metadata setting with the name `my_redirection_url`.

```
public class MyUserAgentRedirectionFilterFactory implements OutboundServiceFilterFactory {

    @Override
    public OutboundServiceFilter newOutboundServiceFilter(URL url,
                AjaxProxyConfiguration proxyConfig,
                Map<String,String> metadata,
                Context runtimeCtx) throws IOException {

        // fetch the settings from the filter configuration.
        String redirection_url = metadata.get("my_redirection_url");
        String  ua_pattern = metadata.get("my_user_agent");
        return new MyCustomFilter(url, ua_patterns);
    }
}
```

The following outbound connection filter code sample checks whether the user agent string of the current outbound connection matches with the pattern that is defined in the configuration:

```
public class MyUserAgentRedirectionFilter implements OutboundServiceFilter {

    private final String redirectionUrl;
    private final String pattern;

    public  MyUserAgentRedirectionFilter(String url, String ua_pattern) {
        redirectionUrl = url;
        patterns = ua_patterns;
    }
```

```
    @Override
    public void preConnect(HttpURLConnection conn, FlowControl ctrl)
    {
        String userAgentString = conn.getRequestProperty("User-Agent");
        if (this.shoudRedirect(userAgentString)) {
            ctrl.setRedirect(redirectionUrl);
        }
    }

    @Override
    public void postConnect(HttpURLConnection conn,
                Map<String, List<String>> modifiableResponseHeaders,
                FlowControl ctrl) { }

    @Override
    public void handleError(ConnectionStatusEvent eventData,
                FlowControl ctrl) { }


    // helper method: check if this filter should apply.
    // returns true if the given user agent matches with the given pattern

    private boolean shouldRedirect(String ua) {
        return Pattern.compile(pattern).matcher(ua).matches();
    }
}
```

Finally, you provide the configuration. In the example scenario, you want to use the user agent redirection filter to redirect outbound HTTP requests to the remote host `http://www.my_remote_server.com/` if the client runs on an android device or iPhone device. You want outbound requests that are submitted from android devices to be redirected to `http://android.my_remote_server.com/index.html`, and requests from an iPhone device to be redirected to `http://iphone.my_remote_server.com/index.html`. The following configuration snippet gives an example:

```
<policy url="http://www.my_remote_server.com/*" >
   <actions>
      <method>GET</method>
   </actions>
   <filter-chain>
      <filter-factory>
         <classname>sample.test.filters.MyUserAgentRedirectionFilterFactory</classname>
       <meta-data>
            <name>my_redirection_url</name>
            <value>http://iphone.my_remote_server.com/index.html</value>
         </meta-data>
         <meta-data>
            <name>my_user_agents</name>
            <value>.*iPhone.*</value>
         </meta-data>
      </filter-factory>

      <filter-factory>
         <classname>sample.test.filters.MyUserAgentRedirectionFilterFactory</classname>
       <meta-data>
            <name>my_redirection_url</name>
            <value>http://android.my_remote_server.com/index.html</value>
         </meta-data>
         <meta-data>
            <name>my_user_agents</name>
            <value>.*Android.*Mobile.*</value>
         </meta-data>
      </filter-factory>
   </filter-chain>
</policy>
```

Built-in configuration settings for custom outbound connection filters:

The following built-in metadata setting for custom connection filter is defined.

active The built-in metadata setting `active` activates or deactivates a custom connection filter. If you want to disable the connection filter but keep the definition in the configuration, set this flag to `false`. Example: The portal administrator wants to temporarily disable the redirection handling of a connection filter.

```
<filter-chain>
    <filter-factory>
        <classname>sample.test.filters.MyUserAgentRedirectionFilterFactory
        </classname>
        <meta-data>
            <name>my_redirection_url</name>
            <value>http://iphone.my_remote_server.com/index.html</value>
        </meta-data>
        <meta-data>
            <name>my_user_agents</name>
            <value>.*iPhone.*</value>
        </meta-data>
        <meta-data>
            <name>active</name>
            <value>false</value>
        </meta-data>
    </filter-factory>
</filter-chain>
```

Using custom cookie transformation handlers:

You can use custom cookie transformation handlers to allow custom program extensions to get programmatic control over the outbound HTTP connection.

About this task

You define custom cookie transformation handlers in a cookie rule. The interface provides two methods for getting programmatic control in the following two phases of the outbound HTTP connection:

- Before the HTTP request header is sent to the remote connection. The transformation handler receives a cookie that applies to the cookie rule.
- After the response header is read. The transformation handler receives a cookie that was set in a `Set-Cookie` header definition.

Application developers can use a custom cookie transformation handler to modify the name, path, domain, or value of the affected cookie, or to produce analytic or statistical information about the cookie.

"Implementing a custom cookie transformation filter"
Application developers can implement a custom cookie transformation filter.

"Registering a custom cookie transformation filter" on page 3061
For a custom cookie transformation filter to take effect, you must register it.

Implementing a custom cookie transformation filter:

Application developers can implement a custom cookie transformation filter.

About this task

To do so, they must provide an implementation of the interfaces `com.ibm.mashups.proxy.ext.CookieTransformer` and

com.ibm.mashups.proxy.ext.CookieTransformerFactory. The factory class creates and returns an instance of the cookie transformer object. The following sample code uses the cookie transformation filter to write the name and value of a certain cookie to a log. The sample code uses System.err as a Writer object.

```
package test.sample;
import com.ibm.mashups.proxy.ext.CookieTransformer;
import com.ibm.mashups.proxy.ext.CookieTransformerFactory;

public class SampleCookieXformerFactory implements CookieTransformerFactory {
 /**
  * Produce and return a new CookieTransformer instance.
  * @param url is the URL for which the cookie transformer is to be opened.
       * @param the metadata that are associated with the transformer object.
       * @return CookieTransformer the custom cookie transformation handler.
         * Must not be null.
         */
 @Override
 public CookieTransformer newCookieTransformer(URL url,
                                  Map<String, String> metadata) {
     return new SampleCookieTransformer (url);
 }
}


package test.sample;
import com.ibm.mashups.proxy.ext.CookieTransformer;
import com.ibm.mashups.proxy.ext.CookieTransformerFactory;
/**
 * The custom cookie transformation handler logs ingoing and outgoing
 * cookies, for which the cookie transformer handler was defined.
 *
 */
public class SampleCookieTransformer implements CookieTransformer (

 // sample code: use System.err as output stream.
 private PrintStream getWriter () {
           return System.err;
 }

 private URL theURL;

        /**
         * Initialize a sample cookie transformation handler.
         * @param url the URL for which the outbound connection was opened.
         **/
        public SampleCookieTransformer (URL url) {
     this.theURL = url;
        }

    /**
     * Process an outgoing cookie.
     * This method is called before the outbound HTTP connection is submitted
     * to the remote host.
     * The sample code writes the name and value of the cookie and the URL
     * to which the connection was submitted to a PrintStream writer.
     * @param cookie the cookie instance.
     */
 @Override
 public void transformOutbound(HttpCookie cookie) {
            getWriter().println (Cookie "+cookie.getName() +" sent to URL
                                 "+theURL+". value="+cookie.getValue());
        }


    /**
     * Process an ingoing cookie.
```

```
     * This method is called after an HTTP response header was read.
     * The response header contains a Set-Cookie statement for a cookie
     * to which this connection handler applies.
     * The sample code writes the name and value of the affected cookie
     * and the URL from where the cookie was set to a PrintStream writer.
     * @param cookie the cookie instance.
     */
 @Override
 public void transformInbound(HttpCookie cookie) {
     getWriter().println (Cookie "+cookie.getName() +" received from URL
                               "+theURL+". value="+cookie.getValue());
        }

}
```

A custom cookie transformation handler is an extension of the outbound HTTP connection service. Therefore, the extension handler code must be accessible by the class loader of the WebSphere Application Server on which WebSphere Portal Express runs. For example, place the code in the following directory:

- Linux: *PortalServer_root*/shared/app
- IBM i: *PortalServer_root*/shared/app
- Windows: *PortalServer_root*\shared\app

Registering a custom cookie transformation filter:

For a custom cookie transformation filter to take effect, you must register it.

About this task

You define custom cookie transformation handlers at a cookie rule. The following xml snippet shows how you register a custom cookie transformation filter:

```
<cookie-rule>
    <cookie>TransformationCookie</cookie>
    <handling>store-in-session</handling>
    <transformation>
        <classname>test.sample.SampleCookieXformerFactory</classname>
    </transformation>
</cookie-rule>
```

The `transformation` section contains a class name that specifies the factory class for custom cookie transformation filters.

Migration:

WebSphere Portal Express Version 8.5 provides a migration process for the change from the Ajax proxy of previous portal versions to the new outbound HTTP connection.

If you upgrade your WebSphere Portal Express from a previous version to Version 8.5, the outbound HTTP connection infrastructure attempts to migrate existing Ajax proxy configuration settings to the outbound HTTP connection service configuration. It moves all settings from the Ajax proxy configuration into database settings for the outbound HTTP connection service. This migration takes place as follows:

Migration of global configuration settings:
In previous WebSphere Portal Express versions, global proxy configuration settings were in the `proxy-config.xml` file of the Ajax Proxy Configuration web module. The outbound HTTP connections infrastructure migrates

global proxy configuration settings when the portal Version 8.5 outbound HTTP connection service is accessed for the first time after the portal upgrade. The global configuration settings are updated in the system configuration profile.

Migration of customized global configuration settings:
In previous WebSphere Portal Express versions, custom global proxy configuration settings were in the WP Configuration Service Resource Environment Provider (REP) property named `proxy.config.file`. The outbound HTTP connections infrastructure migrates these custom global configuration settings when the portal Version 8.5 outbound HTTP connection service is accessed for the first time after the portal upgrade. Custom global configuration settings are updated into the global configuration profile.

Migration of application-specific proxy configurations
Application-specific configurations are imported into the outbound HTTP connection configuration when the web module for the application is deployed. The deployment program scans for the file `/WEB-INF/proxy-config.xml`. If this file exists, the migration program creates a scoped configuration profile that relates to the web module that is created.

Important: In previous WebSphere Portal Express versions, the `proxy-config.xml` file held the configuration of the Ajax proxy. Changes that you make to this file became effective when you restarted the web module. In contrast, the outbound HTTP connection infrastructure makes only the necessary updates to the `proxy-config.xml` file. Changes to this file do not become effective until one of the following events occur:

- For global or customized global configuration settings: The outbound HTTP connection service is started for the first time after the portal upgrade.
- For application-specific proxy configurations: The web module that contains the `proxy-config.xml` is deployed.
- One of the portal configuration tasks `create-outbound-http-connection-config` or `update-outbound-http-connection-config` is started.

Related tasks:

"Configuring outbound HTTP connections by using configuration tasks" on page 3013
Programmers can create, read, update, or delete settings of the outbound HTTP connection by using the appropriate portal configuration engine tasks.

Live text for click-to-action

IBM WebSphere Portal Express supports a live text API for user controlled data transfer between components. With live text, a component on a page can declare sources and targets for data transfer. For example, this can be a portlet or a navigation element. When the user clicks on a source element, the portal displays a menu listing targets that match the selected source. When a menu item is selected, the portal invokes the corresponding target passing it the source data. This process is called Click-to-Action (C2A).

Both sources and targets are specified by HTML markup that apply live text elements. This means that the HTML markup contains special attributes that marks the HTML fragment as a click-to-action tag.

- Sources of click-to-action menus provide a data item that is potentially relevant for other components on the page. The data item has an associated type name that is used to determine the targets that can handle this data item.

- Targets specify the type name that they are interested in and a display title for the click-to-action menu item that represents the target entry. They provide a handler for the received data either as JavaScript code or as a server-side URL to which the data is sent when the menu item is selected.

As both sources and targets are defined by live text elements, all components that contribute HTML markup to a page can provide sources and targets. This includes IBM portlets, JSR standard portlets, and theme and skin components, as well as WCM content or Web clippings. Portlets can register their portlet action or render URLs as server-side targets, but you can also define targets within one portal page that point to another page or even to a Common Gateway Interface (CGI) handler outside the portal.

Click-to-action treats all source data as unstructured text. You can encode any information in the source value, but the target handler is responsible for decoding the received data appropriately.

Click-to-action also integrates with the server side portlet communication programming model: If a portlet provides the following server side communication targets, they will automatically be made available for click-to-action on all pages that contain the portlet:

- Portlet events declared by a JSR 286 portlet in `portlet.xml` with a payload type of `java.lang.String`.
- Cooperative portlet actions declared by JSR 168 portlets in a WSDL deployment descriptor with an input property of class `java.lang.String`.

Therefore portlets that use these declarations do not need to generate semantic tagging markup for click-to-action targets.

Administrators can turn off the automatic generation of click-to-action targets by specifying a portlet preference `com.ibm.portal.c2a.target.generation = false`.

"Live text formats"
Application of live text elements to sources and targets is based on special HTML classes that are attached to elements of the HTML markup.

"Integrating click-to-action targets with the person menu" on page 3066
Target actions for live text based click-to-action can also integrate with the person menus that are generated for hCard microformats found in the page markup.

"Relation to cooperative portlet wiring" on page 3066
You can use both live text elements and cooperative portlet wires to exchange data between portlets on a page.

"Comparison of the new features with click-to-action in IBM portlets" on page 3067
The user experience of live text elements based click-to-action is similar to that of the click-to-action JSP tags that are available for IBM portlets.

Live text formats:

Application of live text elements to sources and targets is based on special HTML classes that are attached to elements of the HTML markup.

Note: The class attribute can be attached to any HTML element and that it can have multiple values, which are separated by spaces. Therefore, you can annotate any HTML element in your output with a click-to-action class, even if it has already a class attribute for CSS formatting.

Sources for Live text are span () or division ( ) elements that are annotated with a c2a:source class. Targets are HTML form (<form>) elements that are annotated with a c2a:target class. Both sources and targets have mandatory and optional properties. These properties are provided by annotated subelements of the main source or target tag. The property value is the entire content of the annotated property element with leading and trailing white spaces removed Example:

```
<b class="c2a:value">
    johndoe@acme.com
</b>
```

represents a c2a:value property with the value johndoe@acme.com. In many cases some of the source or target content is only relevant for the click-to-action framework but not for the user. In such cases, you might want to hide that part of the source or target content from rendering. To do this action, use the HTML attribute style="display:none".

Source tag

The source tag must be an HTML span () or division ( ) elements that are annotated with a c2a:source class. The properties of a source tag are as follows:

Table 456. Source tag properties

Property	Content	Mandatory
c2a:typename	Namespaced type name that describes the format and semantics of the data that is provided.	Yes
c2a:value	Actual data that is passed to the target operation.	Yes
c2a:anchor	Optional anchor point that you can use for displaying the **Click-to-action** hover menu. If you do not specify this property, the content of the property c2a:value is used as the default.	No
c2a:display	Markup that is inserted as a header into the menu that is generated for the source.	No

When the portal finds one or more matching targets for a source tag, the c2a:anchor subelement is marked as an active click-to-action source. And hovering over the element displays a click-to-action menu trigger. You can control the way that source anchors are displayed in the browser by the CSS file styles.css that is located under the directory c:\IBM\WebSphere\PortalServer\ui\wp.tagging.liveobject\semTagEar\Live_Object\Framework.ear\Live_Object_Framework.war\ui. The decoration for elements that offer a hover menu is defined by the following styles by default:

```
.hasHover { border-bottom: 1px dotted #306bc4; }
img.hasHover { padding:1px; border:1px dotted #306bc4; }
```

The value of the property c2a:typename is used to match sources to targets. This value can have an XML namespace associated to avoid unintended type name clashes. As plain HTML does not support the use of standard XML namespaces and namespace prefixes, a namespaced name is represented as namespaceURI#localPart.

Source tag example:

```
<div class="c2a:source someotherclass">
    <span class="c2a:typename"
          style="display:none">http://www.ibm.com/xmlns/prod/datatype#email822</span>
    <p>some content that is not relevant to click-to-action, also to make clear
```

```
       that property elements do not have to be direct children
        <b class="c2a:value">johndoe@acme.com</b>
    </p>
    <img class="c2a:anchor" src="c2aHoverTrigger.jpg" />
    <p class="c2a:display" style="display:none;">
        <b><c>This is a sample click-to-action source</c></b><br>
        <b><c>You can add an optional header to your action menu</c></b>
  </p>
</div>
```

Target tag

The target tag must be an HTML form (<form>) that is annotated with a c2a:target class. The properties of a target tag are as follows:

Table 457. Target tag properties

Property	Content	Mandatory
c2a:typename	Namespaced type name that describes the format and semantics of the data that is expected.	Yes
c2a:action-label	The string that you want to be displayed for the operation in the click-to-action menu.	Yes
c2a:action-param	This property must be an HTML input (<input>) element, designates the place where the data is passed.	No

When the portal finds one or more matching targets for a source tag, the c2a:anchor subelement is marked as an active click-to-action source. And hovering over the element displays a click-to-action menu trigger. You can control the way that source anchors are displayed in the browser by the CSS file styles.css that is located under the directory c:\IBM\WebSphere\PortalServer\ui\wp.tagging.liveobject\semTagEar\Live_Object\Framework.ear\Live_Object_Framework.war\ui. The decoration for elements that offer a hover menu is defined by the following styles by default:

```
.hasHover { border-bottom: 1px dotted #306bc4; }
img.hasHover { padding:1px; border:1px dotted #306bc4; }
```

When a target is selected from a click-to-action menu, the corresponding HTML form is processed as if a user submitted it. The input field that was annotated as c2a:action-param is filled with the source data. Then, JavaScript onsubmit handlers are started, and finally the portal sends the form to its target URL. The target URL must point to the server-side target for click-to-action. If you want to implement a client-side click-to-action handler, use an onsubmit handler on the form. The JavaScript handler can retrieve the form and the input value from the page Document Object Model (DOM). But for convenience, the source data is also passed in the global variable window.ibm.portal.c2a.event.value.

Target tag example for a server-side handler:

```
<FORM class="c2a:target" action="/myapp/do.something">
    <span class="c2a:typename"> http://www.ibm.com/xmlns/prod/datatype#email822</span>
    <p class="c2a:action-label">Show inbox</p>
       Email: <input type="text" class="c2a:action-param" name="someInputName">
    <input type="submit">
</FORM>
```

Target tag example for a client-side handler:

```
<FORM class="c2a:target" onsubmit="doSomething(this);return false"
      action="javascript:void(0)" style="display:none">
    <span class="c2a:typename"> http://www.ibm.com/xmlns/prod/datatype#email822</span>
    <p class="c2a:action-label">Show inbox</p>
    <input type="text" class="c2a:action-param" name="someInputName">
</FORM>
```

Integrating click-to-action targets with the person menu:

Target actions for live text based click-to-action can also integrate with the person menus that are generated for hCard microformats found in the page markup.

Target actions that you want to be displayed in a person menu need to declare the special type name as follows:

`http://www.ibm.com/xmlns/prod/websphere/portal/v6.1/livetext#hcard`

Unlike normal click-to-action sources, the person data that is available in the person menu is complex structured information. Therefore the value of the variable `window.ibm.portal.c2a.event.value` is not a string, but a JavaScript object with multiple nested properties that represent available information about the person. Refer to the person tag documentation about available attributes. The value that is passed in a target input field annotated with `c2a:action-param` is a string representation of this object in JavaScript Object Notation (JSON) format.

If you want target actions to receive only individual fields of the person record, you can indicate this by appending the field selector to the type name. For example, a target action that is declared with the following type name will receive only the `email.internet` nested property of the person record.

`http://www.ibm.com/xmlns/prod/websphere/portal/v6.1/livetext#hcard.email.internet`

If the specified field is not available for a given source, then the target action is not displayed in the person menu.

Relation to cooperative portlet wiring:

You can use both live text elements and cooperative portlet wires to exchange data between portlets on a page.

However, the execution flow for both methods is different:

Flow for cooperative portlet wiring:

1. The source portlet renders an action link.
2. The user clicks a link to execute the source portlet action.
3. The source portlet action is executed on the server, producing the source data and triggering the wires.
4. The wires fire target portlet actions that are executed with the source data.

Flow for click-to-action

1. The source portlet renders a source HTML tag that contains the source data.
2. The user clicks on the source element to open a menu and selects the target action from the menu
3. The source data is sent to the server where a target portlet action is executed with the source data.

The following table lists differences that will help you decide which method for data exchange is appropriate for your application:

Table 458. Comparing click-to-action to portlet wires

Click to action	Portlet wires	
Is based on dynamic presence of targets on a page.	Is based on static administrator-defined wires.	
Triggers a single target selected from a menu.	Triggers all defined targets that have been wired up.	
Can only handle string data.	Can pass complex data types.	

Notes:

1. You can write portlets that support both methods; any portlet action declared in a cooperative portlet WSDL that defines an input parameter is available as a target for both click-to-action and portlet wiring. For sources, you can add a preference that lets the administrator control whether click-to-action live text should be emitted, as demonstrated by the cooperative portlet JSR shipping example.
2. You can also combine both methods: a target action that is triggered by click-to-action can have wires attached that will propagate to other portlet actions on the server within the same request. consequently, a single click-to-action menu selection can trigger multiple portlet actions on the server.

Related concepts:

"Portlet communication" on page 3069
IBM WebSphere Portal Express supports multiple ways for portlets to exchange or share information.

Comparison of the new features with click-to-action in IBM portlets:

The user experience of live text elements based click-to-action is similar to that of the click-to-action JSP tags that are available for IBM portlets.

However, the underlying technologies are different, and the functional possibilities of live text go beyond click-to-action JSP tags. This results in important distinctions:

Table 459. Comparison of the new features to existing click-to-action

	New: Live text for click-to-action	Click-to-action JSP tags for IBM portets
How sources and targets are declared.	Sources and targets can be declared by any HTML fragment on a page, including portlets, theme components or simple HTML clippings. JSR portlet actions declared in a cooperative portlet WSDL are automatically registered as targets.	Sources and targets can only be declared within IBM portlets. Sources are declared by using JSP tags and targets are declared with a cooperative portlet WSDL.
How targets are handled	Targets can register client side (Javascript) or server side (FORM submission) handlers for click-to-action.	Target actions are always handled with a server call.

Table 459. Comparison of the new features to existing click-to-action (continued)

	New: Live text for click-to-action	**Click-to-action JSP tags for IBM portets**
Where menu contents are computed	Menu contents are computed on the client when the user clicks on a source. This results in improved performance and smaller markup size.	Menu contents are computed on the server when the JSP tags are rendered.
Whether menus can be customized	Menu appearance can be customized for each source by specifying a HTML menu head.	Menu appearance cannot be customized.
Whether cross-page actions are possible	Sources and targets must be defined on the same page. Target URLs can lead to a different page in the portal when they are selected.	Cross-page actions can be selected from sources on all pages.
Whether multiple targets are allowed	Only a single target is available for one user action.	Source data can be distributed to multiple targets with a single click.
Whether portlet wiring is supported	No wiring support is available.	Users can automate actions by pressing **Ctrl** and clicking on a menu item.

Related concepts:

"Portlet communication" on page 3069
IBM WebSphere Portal Express supports multiple ways for portlets to exchange or share information.

Client-side aggregation reference

Refer to programming model guidelines for client-side mode or server-side mode.

"Programming model guidelines for server-side mode"
Learn which widgets you can use in server-side mode and how to adapt widgets for server-side mode.

Programming model guidelines for server-side mode

Learn which widgets you can use in server-side mode and how to adapt widgets for server-side mode.

Widgets you can use in server-side mode

In general, all widgets programmed against the iWidget specification can be rendered in server side mode. However, a key difference to client side rendering is that full page refreshes happen frequently and not only if a user action triggers them explicitly. For example, each time a user triggers a portlet action, this results in a full page refresh that fetches the updated page from the server. This also means that the document object model (DOM) is regenerated as a result of a standard user interaction. Therefore, to write a widget that works in server side mode, apply the guidelines listed in the following:

- State always needs to be stored using onNavState changed events and not by storing it in the dom. Otherwise it is gone after a page refresh.

Furthermore, some of the predefined events that the iWidget specification specifies behave slightly different when you use them in server side rendering mode:

- `onRefreshNeeded`: This event is not sent after a page refresh.

- `onSizeChanged`: This event is only sent if the skin used with it supports it.
- `onNewWire`:
  - This event can be sent by a client side user interface component for widgets on the same page.
  - This event is not sent if the wire model of the page was changed externally.
  - This event is not sent after a page refresh.
- `onRemoveWire`:
  - This event can be sent by a client side user interface component for widgets on the same page.
  - This event is not sent if the wire model of the page was changed externally.
  - This event is not sent after a page refresh.

Options of interactions between widgets and portlets

iWidgets cannot:

- Send events to JSR168 or JSR286 portlets.
- Receive events from JSR168 or JSR286 portlets.
- Share render parameters with JSR286 portlets.

iWidgets can:

- Send events to and receive events from other iWidgets.
- Share render parameters with other iWidgets on the same page by using shared item sets.

Portlet communication

IBM WebSphere Portal Express supports multiple ways for portlets to exchange or share information.

Select the topic about the programming technique that is most appropriate for your application requirements from the following links.

"Public render parameters" on page 3070
Public render parameters allow JSR 286 portlets to share navigational state information. They are specially useful for coordinating the multiple navigation or viewer portlets that display different information items that are all related to the same parameter name. The portal stores all portlet render parameters, including public render parameters, as an encoded part of the current portal URL. Therefore, public render parameters are correctly preserved by typical browser navigation actions such as the Back button and bookmarking.

"Advanced URL generation for data exchange" on page 3072
For data exchange, IBM WebSphere Portal Express supports cross-portlet links

"Standard portlets publish and subscribe mechanisms" on page 3074
Both JSR 168 and 286 portlets provide communication capabilities that enable you to pass information from one portlet to another. For JSR 286 portlets, these capabilities are included in the JSR 286 standard. For JSR 168 portlets, an extension defined by IBM provides these capabilities. Portlet developers define these capabilities. Portal administrators then determine whether or not these capabilities are used to pass information between portlets.

"Known issues and restrictions related to standard portlets publish and subscribe mechanisms" on page 3094
Review this information for a list of known issues and restrictions with portlet communication.

"Special purpose techniques for data exchange" on page 3095
IBM WebSphere Portal Express supports special purpose techniques for data exchange.

"Shared portlet sessions" on page 3095
The following communication methods are based on shared state between multiple portlets. This means that two or more portlets read and write to the same data.

Related concepts:

"URL generation in WebSphere Portal" on page 2835
Generating Portal URLs correctly is one of the most important tasks in programming a WebSphere Portal Express based application. There are several programming tools and techniques available for generating WebSphere Portal Express URLs in custom code. The following section introduces the programming tools available and discusses when it is most appropriate to use each of the tools.

"Struts Portlet Framework" on page 2943
Learn how to write Struts applications that can be deployed in WebSphere Portal Express. Know special considerations when developing such applications and additional concepts, such as portlet modes, multiple device support, and portlet communication, that might need to be addressed by the Struts application.

Public render parameters

Public render parameters allow JSR 286 portlets to share navigational state information. They are specially useful for coordinating the multiple navigation or viewer portlets that display different information items that are all related to the same parameter name. The portal stores all portlet render parameters, including public render parameters, as an encoded part of the current portal URL. Therefore, public render parameters are correctly preserved by typical browser navigation actions such as the Back button and bookmarking.

Notes:

- Public render parameters are generated between JSR portlets during the render phase and not the action phase. For more information, see *JSR 286 portlet events based communications* in the related links.
- You can set shared render parameters as well as private render parameters during URL generation by using the state API. For more information, see *Advanced URL generation for data exchange*.
- Public render parameters can also be shared with remote portlets by using the WSRP V 2.0 protocol.
- Information about render parameters is normally encoded into the URL. Therefore their names and values should be as short as possible in order to not exceed the URL length restrictions set by your browser.

Related concepts:

"Advanced URL generation for data exchange" on page 3072
For data exchange, IBM WebSphere Portal Express supports cross-portlet links

Related tasks:

"Creating a web content page with the XML configuration interface" on page 2074
As with other portal pages, you can create a web content page with the XML configuration interface (**`xmlaccess`** command). Page definition is similar to a standard portal page. However, there is an additional page parameter that specifies the site area that is associated with the web content page.

Related reference:

"JSR 286 portlet events based communications" on page 3075
Portlet events provide a powerful and flexible publish/subscribe mechanism for communication between JSR 286 portlets. They can be used to exchange complex data between portlets and to trigger portlet activity such as updates to back end systems. In the portal, they can also interoperate with other communication mechanisms such as Cooperative portlets and click-to-action.

Concepts of the Java Portlet Specification 2.0: Programming details for public render parameters are defined in the Java Portlet Specification 2.0. They are read as request parameters and set in render URLs or on an ActionResponse. For most purposes, public render parameters behave exactly like "normal" or private render parameters. The only difference is that the parameter identifier is explicitly declared by the programmer in the `portlet.xml` deployment descriptor, along with one or more namespaced public names. As a result, a portal can decide which parameters of two portlets must map to the same information.

Here is an example declaration in the deployment descriptor:

```
<portlet-app xmlns:x="http://acme.com/portlet">
   <portlet>
      ...
      <supported-public-render-parameter>custID</supported-public-render-parameter>
   </portlet>
   <public-render-parameter>
      <identifier>custID</identifier>
      <qname>x:customerID</qname>
   </public-render-parameter>
</portlet-app>
```

This snippet declares a public render parameter that is accessed with the parameter key `custID`. Here is an example code snippet for reading the parameter:

```
String customerID = renderRequest.getParameter("custID");
```

The parameter is shared with the global name, which consists of the namespace and the localname, and the `customerID` in the namespace `http://acme.com/portlet`. Therefore, if another portlet declares a parameter with the same global name - regardless of the local identifier that it uses - that parameter can be mapped to the same data by the portal.

Portlets can declare globalized display names and descriptions for public render parameters in the application resource bundle. However, WebSphere Portal Express Version 8.5 does not have a user interface that requires this information for display.

How to control parameter sharing in the portal: The portal implements the sharing of parameters by placing them in scopes: Two or more public render parameters from different portlets are mapped to the same data in the portlet URL, if and only if both of the following conditions apply:

- The parameters declare the same global name that includes the namespace.
- The parameters are placed in the same scope.

The default scope for all public parameters is the global scope. Therefore, all public parameters with the same global name are shared by all portlets across all pages.

For some use cases, it can be required to limit parameter sharing. For example, this case can occur if you have two pairs of navigator or viewer portlets on two different pages, where each pair must be coordinated. However, the pairs must not interfere across pages so that the navigator on the first page does not influence the viewer on the second page. For such cases, the portal makes it possible to limit the

sharing scope for public render parameters on a page basis. The parameter sharing scope for a page is controlled by a page parameter `param.sharing.scope`. You can set it from the **Page Properties** view under **Advanced options** > **I want to set parameters**. If you set a value for this parameter, portlets on the page share their public render parameters only with other portlets on the same page or on pages with the same scope.

If you want to prevent all render parameters that are used on a page from being shared with other pages, you can set the parameter `param.sharing.scope` to the reserved value `ibm.portal.sharing.scope.page`. Specifying this value produces the same result as using the unique object ID of the page, as generated internally by the portal, for the value of the `param.sharing.scope` parameter.

How to set the scope for specific public render parameters: The setting `param.sharing.scope` mentioned earlier sets the scope for all public render parameters. In addition, it is also possible to set the scope for specific public render parameters. The WebSphere Portal Express template pages use this approach. For example, the Basic page template specifies the following parameters:

```
<parameter
   name="param.sharing.scope.{http://www.ibm.com/xmlns/prod/datatype/content/resource-collections}"
   type="string" update="set"><![CDATA[ibm.portal.sharing.scope.page]]></parameter>
<parameter
   name="param.sharing.scope.{http://www.ibm.com/xmlns/prod/datatype/content}"
   type="string" update="set"><![CDATA[ibm.portal.sharing.scope.page]]></parameter>
<parameter
   name="param.sharing.scope.{http://www.ibm.com/xmlns/prod/websphere/portal/v7.0/portal-contextual-portal}"
   type="string" update="set"><![CDATA[ibm.portal.sharing.scope.page]]></parameter>
<parameter
   name="param.sharing.scope.{http://www.ibm.com/xmlns/prod/websphere/portal/v8.0/portal-contextual-portal}"
   type="string" update="set"><![CDATA[ibm.portal.sharing.scope.page]]></parameter>
<parameter
   name="param.sharing.scope.{http://www.ibm.com/xmlns/prod/websphere/portal/publicparams}path-info"
   type="string" update="set"><![CDATA[ibm.portal.sharing.scope.page]]></parameter>
```

This snippet from a template sets the page scope for all public render parameters that are contained in the following name spaces:

`http://www.ibm.com/xmlns/prod/datatype/content/resource-collections`
`http://www.ibm.com/xmlns/prod/datatype/content`
`http://www.ibm.com/xmlns/prod/websphere/portal/v7.0/portal-contextualportal`
`http://www.ibm.com/xmlns/prod/websphere/portal/v8.0/portal-contextualportal`

It also sets the scope for the `path-info` parameter that is defined in the namespace

`http://www.ibm.com/xmlns/prod/websphere/portal/publicparams`

Limitations: The following are known limitations to public render parameters:

1. The sharing scope always applies to all parameters of all portlets on a page. You currently cannot control sharing of public render parameters at a more granular level than the page level.
2. The optional `<alias>` elements of public render parameter declarations are ignored. Currently, sharing of public render parameters is only controlled by the `<name>` or `<qname>` elements of the parameter declarations

Advanced URL generation for data exchange

For data exchange, IBM WebSphere Portal Express supports cross-portlet links

Cross-portlet links work for both IBM and JSR portlets. They use the state API to generate a URL that points to a render parameter of a specific portlet. With the advanced URL generation APIs for portlets, you can have a portlet create a link that points to another portlet, possibly even on a different page, and pass data to that portlet. If it is at all possible, use the publish/subscribe methods. Use explicit

cross-portlet links only where other techniques are not applicable. For example, this can be for communication between IBM and JSR portlets.

For details, refer to the documentation about the portalRenderURL tag and the PortalURLGenerationService. Such links can also be created from a theme JSP that uses the portal URL generation tag.

For advanced use cases that require more control over the generated URLs, you can also use the portal state API.

If the pure render parameter approach is not applicable, generating URLs and transporting the data to be communicated is suggested. There are three options available: JSP tag URL generation, PortalURLGenerationService, and state API-based URL generation.

JSP tag URL generation
This approach is only supported for JSR portlets and allows the user to generate URLS that transport information from within portlet JSPs. For more information, see *JSP tags for standard portlets* in the related links.

PortalURLGenerationService based URL generation
Use this service to generate render URLs to other portlets. For more information, see *Interface PortalURLGenerationService* in the related links. This option might be available only if specific portlets are targeted and if render URLs need to be created for the target portlets in question. For more information, see *Leveraging WebSphere Portal V6 programming model: Part 2. Advanced URL generation in themes and portlets* in the related links.

State API-based link generation
Generating state API-based URLs allows the transfer of data to portlets developed by using the JSR168 API or JSR286 API. State API-based link generation is the most powerful way of modifying the navigational state and thus share information between portlets. In addition to the portlet state aspects such as portlet mode and window state, state API-based link generation also enables you to set private or public render parameters for one or multiple portlets. For more information, see *Leveraging WebSphere Portal V6 programming model: Part 2. Advanced URL generation in themes and portlets*.

Notes:

- To set private render parameters for a given portlet, request the PortletAccessorController from the PortletAccessorFactory. For more information, see *Interface PortletAccessorController* in the related links.
- To modify shared state information such as public render parameters for JSR286 portlets, you can also request the SharedStateAccessorController from the PortletAccessorFactory. For more information, see *Interface SharedStateAccessorController* in the related links.
- So far, the URLs being generated modify navigational state information only and do not carry any action semantic, which is necessary to modify the persistent backend state. PortletTargetAccessorController is available from the PortletAccessorFactory and enables the targeting of a specific portlet's processAction method. PortletTargetAccessorController triggers a processAction on the specific target portlet that the generated URL points to.
- Standard portlets provide powerful publish and subscribe mechanisms for exchanging information through an action phase invoked between portlets.

During the invocation, portlets can exchange complex data and trigger portlet activity such as updates to back-end systems.

Related concepts:

"Public render parameters" on page 3070
Public render parameters allow JSR 286 portlets to share navigational state information. They are specially useful for coordinating the multiple navigation or viewer portlets that display different information items that are all related to the same parameter name. The portal stores all portlet render parameters, including public render parameters, as an encoded part of the current portal URL. Therefore, public render parameters are correctly preserved by typical browser navigation actions such as the Back button and bookmarking.

Related reference:

"JSP tags for standard portlets" on page 3133
The standard portlet API defines several tags that can be used in portlet JSPs to access the portlet request and response and to generate URLs.

Related information:

http://www-128.ibm.com/developerworks/websphere/library/techarticles/0612_behl/0612_behl.html

http://www-128.ibm.com/developerworks/websphere/library/techarticles/0603_behl/0603_behl.html

Interface PortalURLGenerationService

Leveraging WebSphere Portal V6 programming model: Part 2. Advanced URL generation in themes and portlets

Interface PortletAccessorController

Interface SharedStateAccessorController

Standard portlets publish and subscribe mechanisms

Both JSR 168 and 286 portlets provide communication capabilities that enable you to pass information from one portlet to another. For JSR 286 portlets, these capabilities are included in the JSR 286 standard. For JSR 168 portlets, an extension defined by IBM provides these capabilities. Portlet developers define these capabilities. Portal administrators then determine whether or not these capabilities are used to pass information between portlets.

"Define portlet communication capabilities"
Standard portlets provide powerful publish and subscribe mechanisms for exchanging information by using an action such as semantic invocation. These mechanisms can be used to exchange complex data between portlets and to trigger portlet activity such as updates to back-end systems.

"Triggering communication" on page 3090
These communication methods are based on directed communication links that pass information from a source portlet to a target portlet.

Define portlet communication capabilities:

Standard portlets provide powerful publish and subscribe mechanisms for exchanging information by using an action such as semantic invocation. These mechanisms can be used to exchange complex data between portlets and to trigger portlet activity such as updates to back-end systems.

"JSR 286 portlet events based communications"
Portlet events provide a powerful and flexible publish/subscribe mechanism for communication between JSR 286 portlets. They can be used to exchange complex data between portlets and to trigger portlet activity such as updates to back end systems. In the portal, they can also interoperate with other communication mechanisms such as Cooperative portlets and click-to-action.

"JSR 168 IBM extension for cooperative portlets" on page 3077
Cooperative portlets represent an IBM specific API for publish/subscribe communication between portlets.

"Interoperability between JSR 286 portlet events and JSR 168 cooperative portlets" on page 3087
By concept, cooperative portlets are similar to JSR 286 portlet events. Both concepts describe publish/subscribe communication patterns that are based on typed information that is published and received by portlets and propagated via communication links.

JSR 286 portlet events based communications:

Portlet events provide a powerful and flexible publish/subscribe mechanism for communication between JSR 286 portlets. They can be used to exchange complex data between portlets and to trigger portlet activity such as updates to back end systems. In the portal, they can also interoperate with other communication mechanisms such as Cooperative portlets and click-to-action.

For more information about the portal event distribution mechanism, read the section about the portal *Event broker*.

Concepts of the Java Portlet Specification 2.0

Programming details for portlet events are defined in the Java Portlet Specification 2.0. Portlets can publish events by using the `response.setEvent()` call and receive events in the `processEvent` method. The `GenericPortlet` class provides a default event handling mechanism that dispatches events based on the `@ProcessEvent` annotation.

As a portlet programmer, declare all events that a portlet can publish or receive in the `portlet.xml` deployment descriptor.

Example declaration in the deployment descriptor:

```
<portlet-app xmlns:x="http://acme.com/portlet"
             xmlns:std="http://somestandardsbody.org/interop-events">
    <portlet>
        ...
        <supported-processing-event>
            <qname>x:address.showForUpdate</qname>
        </supported-processing-event>
        <supported-publishing-event>
            <qname>x:address.updated</qname>
        </supported-publishing-event>
    </portlet>
    <event-definition>
        <qname>x:address.showForUpdate</qname>
        <alias>std:address</alias>
        <value-type>com.acme.portlets.common.Address</value-type>
    </event-definition>
    <event-definition>
        <qname>x:address.updated</qname>
```

```
        <alias>std:address</alias>
        <value-type>com.acme.portlets.common.Address</value-type>
    </event-definition>
</portlet-app>
```

This declares a publishing and a processing event with a Java payloads of class `com.acme.portlets.common.Address` and global names `address.showForUpdate` and `address.updated` in the namespace `http://acme.com/portlet`. In addition, the alias `address` in the namespace `http://somestandardsbody.org/interop-events` of some presumed standardization organization indicates that these events are compatible with any events of another portlet that has the same alias and can therefore work with input or output values of the provided address type. Here is a coding example:

```
public class MyPortlet extends GenericPortlet {
    public static String NAMESPACE = "http://acme.com/portlet";

    @ProcessAction(name="address.updated")
    public void addressUpdated(ActionRequest request, ActionResponse response) {
      ...
      Address myAddress = ...;
      response.setEvent(new QName(NAMESPACE, "address.updated"), myAddress);
    }
    @ProcessEvent(name="address.showForUpdate")
    public void addressUpdated(EventRequest request, EventResponse response) {
      Address myAddress = (Address) request.getEvent();
      ...
    }
```

Controlling event distribution in the portal

The Java Portlet Specification intentionally leaves open how events are passed between portlets, but only specifies how they are published and received. In IBM WebSphere Portal Express Version 8.5, event distribution is based on the same event broker and wiring techniques that are used to connect cooperative portlets. That means that when you place a portlet on a page, it will initially not be able to publish or receive any events. You must use the portlet wiring tool link to connect the events declared by the portlet to outputs or inputs of other portlets.

Portlets can declare localized display names and descriptions for portlet events in the application resource bundle. Provide at least a display name, as the portlet wiring tool needs this information to display event sources and targets properly.

The portlet `EventDistributionService` provides the option to have user interface elements displayed dependent on whether a given event is wired to targets or not. The wiring information provided by this portlet service is available for all requests in which the property `com.ibm.portal.portlet.Constants.FEATURE_EVENT_DISTRIBUTION_SERVICE` is set and available. For incoming WSRP requests this property is not available.

Support for complex event types

As noted previously, portlet events support complex Java types as payloads. The Java Portlet Specification requires that such payload classes support Java serialization as well as XML serialization by using Java XML Binding (JAXB) annotations. The portal supports transfer of such payloads between portlets, even if they are deployed in different WAR files. However, if the payload class is packaged as part of the portlet WAR file, it is necessary to serialize and deserialize values during the transport because both portlets use different class loaders, which will result in less than optimal run time performance.

To obtain optimal performance for transferring complex Java payloads, at the cost of a more complicated deployment process, remove the payload classes from the WAR file and deploy them in a class loader that is shared by both portlets. You can do this by using the IBM WebSphere Application Server shared libraries, or by placing the shared classes in the *wp_profile_root*/PortalServer/config directory.

Related concepts:

"Public render parameters" on page 3070
Public render parameters allow JSR 286 portlets to share navigational state information. They are specially useful for coordinating the multiple navigation or viewer portlets that display different information items that are all related to the same parameter name. The portal stores all portlet render parameters, including public render parameters, as an encoded part of the current portal URL. Therefore, public render parameters are correctly preserved by typical browser navigation actions such as the Back button and bookmarking.

JSR 168 IBM extension for cooperative portlets:

Cooperative portlets represent an IBM specific API for publish/subscribe communication between portlets.

You can use cooperative portlet support with either IBM or JSR 168 portlets, but you cannot use it for communication between portlets of different API types.

For development of new portlets, use the standards based JSR 286 portlet events instead. They provide equivalent and compatible functionality. For more information about moving from cooperative portlets to JSR 286 events, refer to the documentation about interoperability between events and cooperative portlets.

"Cooperative portlets overview"
Cooperative portlets communicate by using properties that are produced and consumed by portlet actions. Each action on a portlet can be associated with a single input property and zero or more output properties. An additional WSDL deployment descriptor defines which portlet actions are enabled for communication. It also provides information about how the property value is transferred to or from the action, for example as a request attribute or a JSR 168 render parameter.

"Cooperative portlet programming model" on page 3078
View information on how properties and actions are registered, how properties are generated and received, and some available APIs.

"Packaging, deploying and compiling cooperative portlets" on page 3079
When packaging, deploying, and compiling cooperative portlets, refer to these considerations on aspects of the process such as creating deployment descriptors and packaging the WAR file.

"WSDL reference for cooperative portlets" on page 3081
WSDL is often used in the context of web services, in order to define interfaces implemented by a web service. The elements in the WSDL used by click-to-action are described, along with extensions to the <binding> element and the WSDL Extensions schema.

Cooperative portlets overview:

Cooperative portlets communicate by using properties that are produced and consumed by portlet actions. Each action on a portlet can be associated with a single input property and zero or more output properties. An additional WSDL deployment descriptor defines which portlet actions are enabled for

communication. It also provides information about how the property value is transferred to or from the action, for example as a request attribute or a JSR 168 render parameter.

This means that cooperative portlets do not require extra APIs for communication, but can use the same interfaces that handle normal action processing. Action processing in a target portlet often does not actually need to distinguish between an action initiated by a portlet URL and an action initiated by the transfer of a property value from another portlet.

Note: Cooperative portlet WSDLs are not evaluated for JSR 286 portlets, and therefore JSR 286 portlets do not support this model of re-using portlet actions for portlet communication.

For more detailed information about the cooperative portlet programming model refer to the documentation about the cooperative portlet programming model. For more information about how to declare actions and properties in cooperative portlet WSDL descriptors.

Packaging considerations for cooperative portlets

When you write cooperative portlets, you need to observe some special guidelines for packaging and deployment descriptors. In the case of standard portlets, the function of the wrapper is provided through a portlet filter, and no extra run time code has to be included in the portlet WAR file.

Cooperative portlet programming model:

View information on how properties and actions are registered, how properties are generated and received, and some available APIs.

Registering properties and actions

The declarative method can be used with standard portlets and is the only registration implementation supported for standard portlets.

Declarative (WSDL). Target portlets associate their actions with an input property which has been declared as an XML type. The actions are declared using WSDL, with a custom binding extension which specifies the mapping from the abstract action declaration to the actual action implementation. WSDL is a language used to define collections of abstract operations, together with the input and output parameters (called `parts` in WSDL) for each operation.

Associated with each action is zero or one input parameters and zero or more output parameters. Each input or output parameter is associated with exactly one property. Each property is associated with an XML type. The input property's type is used for matching the action to output properties which are produced by portlets. The output parameters are produced as a result of executing the action, and can be wired to other actions subject to the type matching constraint mentioned previously.

Generating property values

Action bindings

The WSDL custom binding extension provides mapping information from the abstract operation declaration to the actual action implemented by the portlet. The custom binding extension defines how information is delivered

to the action implementation and received from the action implementation. Actions can generate output properties to transfer information, and the binding used in the implementation must match the parameter binding declared in the WSDL file.

Receiving property values

Action bindings

Action bindings may also be used to receive property values. The bindings used to provide the property values to the action implementation will match the input parameter bindings specified in the WSDL file.

Advanced programmatic APIs

For standard portlets, the property broker provides additional APIs to give developers more control over inter-portlet communication. The following functionality can be implemented using the programmatic APIs:

-
  - Activate or deactivate actions for a session. Only active actions can be triggered through wires or through Click-to-Action menus.
  - Determine if a property is wired, and if it is a cross-page wire. It can also determine if the wire is actually active (the target action is active). This knowledge may be used to render links for triggering wired actions.

Note: Both sources and targets must register their input and output properties and actions for wire creation to be possible.

The following package define the programmatic APIs provided by the property broker: com.ibm.portal.propertybroker.service.PropertyBrokerService (this is an interface).

Packaging, deploying and compiling cooperative portlets:

When packaging, deploying, and compiling cooperative portlets, refer to these considerations on aspects of the process such as creating deployment descriptors and packaging the WAR file.

Creating the deployment descriptors

A WAR must have a `web.xml` and a `portlet.xml` file in order to comply with J2EE specifications. For standard portlets, `web.xml` needs to contain only servlet information, not portlet information. If the standard portlet does not contain servlets, `web.xml` must still be present, though the content of the file will be empty.

The `portlet.xml` file must specify the location of the WSDL file associated with each portlet. To achieve this, you must modify `portlet.xml` to add a configuration parameter to each concrete portlet that exposes actions to the property broker through the WSDL file.

- Standard portlet examples

  Sample `web.xml` file for standard portlet with empty content:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
                         "http://java.sun.com/dtd/web-app_2_3.dtd">
<web-app>
    <display-name>Shipping Portlets - Standard Version</display-name>
</web-app>
```

Sample standard portlet.xml file

```
<?xml version="1.0"?>
<portlet-app
   xmlns="http://java.sun.com/xml/ns/portlet/portlet-app_1_0.xsd"
   version="1.0"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://java.sun.com/xml/ns/portlet/portlet-app_1_0.xsd
                       http://java.sun.com/xml/ns/portlet/portlet-app_1_0.xsd">
    <portlet>
        <description xml:lang="en">
           Displays details for a particular order
        </description>
        <portlet-name>StandardOrderDetail</portlet-name>
        <display-name xml:lang="en">Standard Order Detail</display-name>
        <portlet-class>
           com.ibm.wps.portlets.shipping.OrderDetailPortlet
        </portlet-class>
        <expiration-cache>3600</expiration-cache>
        <supports>
            <mime-type>text/html</mime-type>
        </supports>
        <supported-locale>en</supported-locale>
        <resource-bundle>nls.StandardOrderDetail</resource-bundle>
        <portlet-preferences>
            <preference>
                <name>
                   com.ibm.portal.propertybroker.wsdllocation
                </name>
                <value>/wsdl/OrderDetail.wsdl</value>
            </preference>
            <preference>
                <name>com.ibm.portal.context.enable</name>
                <value>true</value>
            </preference>
        </portlet-preferences>
    </portlet>

 ...

</portlet-app>
```

Note: For standard portlets, you must use a special <preference> entry to specify the application portlet class.

```
<portlet-preferences>
   <preference>
      <name>com.ibm.portal.propertybroker.wsdllocation</name>
      <value>/wsdl/OrderDetail.wsdl</value>
   </preference>
</portlet-preferences>
```

WAR file considerations

Once the code and deployment changes have been made for using the property broker, additional libraries and files must be packaged along with the application. After you package the WAR file, it is ready to be installed. Use the following table to package the files in the correct location.

Table 460. Files to package and their correct location

Portlet type	Standard portlets
File name	WSDL file
Path	relative to the root of the WAR file or an absolute URL

Additional considerations for compiling

If you are using your own development environment, make sure to add the `wp.propertybroker.standard.api.jar` file to your class path for standard portlets. These JAR files can be found in the *PortalServer_root*`/base/wp.propertybroker.standard.api/shared/app` directory. For other JAR file requirements, read Creating a simple portlet.

WSDL reference for cooperative portlets:

WSDL is often used in the context of web services, in order to define interfaces implemented by a web service. The elements in the WSDL used by click-to-action are described, along with extensions to the `<binding>` element and the WSDL Extensions schema.

Usually, a SOAP binding is used to specify the concrete realization of the interface by a web service which supports the SOAP protocol. The following shows how Click-to-Action uses some of the elements in the WSDL document. The extensions to the <binding> element are described and the complete schema for WSDL Extensions is provided.

- **<types>**

  For cooperative portlets, this declares the data type of the data to be transferred. The data type is declared using XML Schema Datatypes (XSD, see the XSD specification). There may be multiple types defined in the document.

  The following shows a declaration for the TrackingIDType:

  ```
  <types>
    <xsd:simpleType name="TrackingIDType">
      <xsd:restriction base="xsd:string">
      </xsd:restriction>
    </xsd:simpleType>
  </types>
  ```

- **<message>**

  Input messages can contain only one part.

- **<operation>**

  Provides an abstract definition of a Click-to-Action operation. Note the restriction on input messages mentioned previously.

- **<portType>**

  Defines an abstract collection of operations. The operations must be defined in the document. One operation corresponds to each action on the Click-to-Action enabled portlet. Only actions that are to be enabled for Click-to-Action should be declared.

- **<binding>**

  The binding element is extended to associate portlet actions with operations. For each operation, the portlet action name must be provided. The portlet action name may be specified using the **name** attribute of the action tag in the binding section of the WSDL file. If it is omitted, the **name** attribute from the operation tag is used as the portlet action name.

  For each operation parameter, the action parameter name must be provided. The portlet parameter name may be specified using the **name** attribute of the parameter tag in the binding section of the WSDL file. If it is omitted, the **name** attribute of the part tag associated with the param tag is used as the portlet parameter name.

Further, the **boundTo** attribute may be used to specify where the parameter will be bound. Choices are request-parameter, request-attribute, session-attribute, or action-attribute.

The <binding> element includes cooperative portlet extensions, which are described in Extensions to the <binding> element.

If you are familiar with WSDL, you might notice that the service section (enclosed by the <service> element in WSDL) is not used in the C2A action declaration file. This is because the file is associated with a specific portlet implementing the operations defined in the file through external means (an entry in the portlet.xml file associated with the portlet).

Extensions to the <binding> element

The <binding> element has been extended to support cooperative portlets. Each extension element is prefixed with `portlet:`, which refers to the C2A namespace, `http://www.ibm.com/wps/c2a`. The `portlet:` prefix is used to identify the extension elements in this section, but a different name may be used for the prefix as long as it refers to the C2A namespace. See WSDL Extension Schema

- <portlet:binding>

  This must be the first child element of the WSDL <binding> element. Its presence identifies the section as a C2A binding extension for portlet action invocation. The element has the following attribute.

  – **style**

    Deprecated. Specifying `style="struts"` indicates that Struts actions are being declared. Instead, you should use `type=struts` on the <portlet:action> element for this purpose.

- <portlet:action>

  This must be the first child element of any WSDL <operation> element in the binding section. It contains the following attributes:

  – **name**

    the portlet action name. If this is omitted, the name of the associated operation element is used as the portlet action name.

  – **caption**

    a short string about the action suitable for displaying in the portlet user interface. For translated captions, indicate in dotted format the name of the key in the resource bundles from which the caption is to be obtained.

  – **type**

    Indicates one of the following values:

    - **default** indicates a DefaultPortletAction object is used. This object is deprecated but still supported for migration purposes.
    - **simple** indicates a simple portlet action String is used.
    - **struts** indicates a Struts action is used.
    - **standard** indicates a standard (JSR) portlet action is used.
    - **standard-struts** indicates a struts action is used with a standard (JSR) portlet.

  – **actionNameParameter**

    (Standard only) The name of the request parameter which will be used to supply the action name.

  – **description**

a text description of the action. For translated descriptions, indicate in dotted format the name of the key in the resource bundles from which the description is to be obtained.

– **activeOnStartup**

Indicate either "true" or "false". If false, the action must be programmatically activated in each session. If true (default), the action is active as soon as the portlet is initialized.

– **selectOnMultipleMatch**

In the case that multiple actions match based on the data type for the portlet, this attribute can be used to indicate which action to trigger. Multiple matching actions can occur when a user broadcasts an action on a source. If not specified, the default is false. In the case where no single action is set to true, the data is not delivered to the target.

In the following example from `TrackingC2A.wsdl`, There are two actions declared: trackingDetails and routingDetails. **selectOnMultipleMatch** is used to indicate that the trackingDetails should be used in the case of a multiple match.

```
<binding name="TrackingBinding" type="tns:Tracking_Service">
   <portlet:binding/>
   <operation name="trackingDetails">
      <portlet:action caption="Tracking Details"
               description="Get tracking details for specified tracking id"
               selectOnMultipleMatch="true"/>
      <input>
          <portlet:param name="trackingId" partname="trackingId"/>
      </input>
      <output>
          <portlet:param name="customerName" partname="custName"
                         boundTo="session"/>
      </output>
   </operation>
   <operation name="routingDetails">
      <portlet:action caption="Routing Details"
               description="Get routing details for specified tracking id"/>
      <input>
          <portlet:param name="trackingId" partname="trackingId"/>
      </input>
   </operation>
</binding>
```

- <constant:param>

  This is used to set the value returned by the `getConstantParameters` method of the action. Each constant parameter will be bound as a request parameter with the specified name and value when the action is invoked. More than one constant parameter can be specified.

  – **name**

  the name of the parameter.

  – **value**

  specifies the parameter value.

  The following shows an example for using the constant parameter.

  ```
  <portlet:constant-params>
       <portlet:constant-param name="defaultMonth" value="January"/>
    </portlet:constant-params>
  ```

- <portlet:param>

  This must appear as a child element in the <input> or <output> subelements of the <operation> element in the binding. It specifies the parameters consumed (if enclosed in the <input> element) or produced (if enclosed in the <output>

element) by the portlet action. At present, we restrict the number of parameters consumed to at most one. The number of parameters produced can be any number. It has the following attributes:

- **name**

  the name of the parameter. If omitted, the name attribute value of the part element is used as the parameter name.

- **partname**

  refers to a part element of the input or output for the operation. It may be omitted if the input or output has a single part.

- **boundTo**

  specifies where the parameter value is bound. Currently, this attribute can specify one of the following values only:

  - `request-parameter` : This specifies that the value is bound as a parameter in the PortletRequest object. This is the default value if the boundTo attribute is omitted. Note that for output parameters, a different value should usually be specified as the default PortletRequest implementation provided by WebSphere Portal Express does not allow parameters to be set during action processing.
  - `request-attribute` : This specifies that the value is bound as an attribute in the PortletRequest object.
  - `session` : This specifies that the value is bound to the PortletSession object.
  - `action` : This specifies that the value is bound to a portlet action.
  - `render-parameter:` (JSR only) This specifies that the value is bound as a render parameter in the ActionResponse (output parameters only).

- **presentIfNullValue**

  specifies is null values for the parameter are significant for the purposes of wired action invocation. Value can be either "true" or "false", with "false" being the default. If "true", the presence of the parameter will cause the wired action to be invoked even if the value of the parameter is null. If "false", the presence of the parameter will cause the wired action to be invoked only if the parameter value is not null. Can be specified for either input or output parameters (i.e, the treatment of null parameter values can be controlled at either the source or the target).

WSDL Extension Schema

The following is the schema for the extensibility elements introduced for the portlet action invocation using Click-to-Action. Lines have been broken to allow viewing on the printed page.

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsd:schema xmlns="http://www.w3.org/2001/XMLSchema"
               xmlns:xsd="http://www.w3.org/2001/XMLSchema"
               targetNamespace="http://www.ibm.com/wps/c2a"
               xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
               xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xsi:schemaLocation="http://schemas.xmlsoap.org/wsdl/
                                   http://schemas.xmlsoap.org/wsdl/
                                   http://www.w3.org/2001/XMLSchema
                                   http://www.w3.org/2001/XMLSchema.xsd"
               xmlns:portlet="http://www.ibm.com/wps/c2a">
   <!-- The binding  element indicates that the binding section
        contains custom extensions describing a mapping of operations to
        portlet actions. This must be placed within a WSDL
        binding element. -->
<element name="binding" type="portlet:bindingType"/>
```

```
<complexType name="bindingType">
   <!-- DEPRECATED, retained for compatibility with version 4. Use the
   type attribute with the action element instead. Set value to be
   "struts" for a portlet implemented using the struts framework. -->
   <attribute name="style" type="string" use="optional"/>
</complexType>
<!-- The action element is used to provide all the information
  about the portlet action necessary for the property broker to
  invoke it. This must be placed within an wsdl:operation element
  in the wsdl:binding section. -->
<element name="action" type="portlet:actionType"/>
<complexType name="actionType">
    <sequence>
        <!-- The constant-params element is used to provide constant
             parameters associated with the action. These are bound
             as request parameters when the action is invoked -->
        <element name="constant-params" type="portlet:constantParamsType"
             minOccurs="0" maxOccurs="1"/>
        </sequence>
         <!-- The name of the action.
              Will be set as the portlet action name.
              If omitted, the name of the corresponding operation
              element will be used as the action name. -->
        <attribute name="name" type="string" use="optional"/>
        <!-- The type of the action. Currently recognized values
             are default, simple, struts, and standard,
             with default being the default.
             The values specify whether the legacy DefaultPortletAction,
             legacy simple portlet action, legacy portlet
             struts action, or standard (JSR-168) portlet
             action mechanism is used to invoke the portlet action.
             The use of DefaultPortletAction is
             deprecated, and the default is set to this
             value for backwards compatibility.-->
        <attribute name="type" type="string" use="optional"
           default="default"/>
        <!-- A short name for  the action which is displayed to the
          user.If a message resource file is
          provided for the portlet,
          the value is used as a key to retrieve a translated string -->
        <attribute name="caption" type="string" use="optional"/>
        <!-- A description for  the action which is
             displayed to the user.
             If a message resource file is provided for the portlet, the
             value is used as a key to retrieve a translated string -->
        <attribute name="description" type="string" use="optional"/>
        <!-- If more than one portlet action can simultaneously process
             a property value, only those with the
             invokeOnMultipleMatch attribute
             set to true will be invoked -->
        <attribute name="selectOnMultipleMatch" type="boolean"
           use="optional" default="false"/>
        <!-- If the activeOnStartup attribute is true,
             the action may be selected for invocation
             at any time unless  programmatically
             deactivated on a per-session basis.
             If the activeOnStartup is false, the action may not be
             selected for invocation, unless
             programmatically activated on a per-session basis -->
        <attribute name="activeOnStartup" type="boolean"
           use="optional" default="true"/>
        <!-- The actionNameParameter attribute is used to specify
             the name of a request parameter whose value will carry
             the action name. It is recommended that all actions of
             a portlet use the same value of this parameter. This is
             used to identify the specific action which is being executed
             on the portlet, as the processAction method in JSR 168
```

```
          does not explicitly pass any information identifying the
          action. If this attribute is omitted, the
          actionNameParameter value defaults to
          com.ibm.portal.propertybroker.action -->
     <attribute name="actionNameParameter" type="string"
        use="optional"
        default="com.ibm.portal.propertybroker.action"/>
</complexType>
<complexType name="constantParamsType">
     <!-- The constant-params element is used to provide constant
          parameters associated with the action. These are bound
          as request parameters when the action is invoked -->
     <sequence>
     <!-- The constant-param element is used to provide a name
          and a value which will be bound as a request parameter
          when the portlet's action is invoked -->
     <element name="constant-param" type="portlet:constantParamType"
      minOccurs="0" maxOccurs="unbounded"/>
     </sequence>
</complexType>
<complexType name="constantParamType">
     <!-- The constant-param element is used to provide a name
          and a value which will be bound as a request parameter
          when the portlet's action is invoked -->
     <!-- The name of the constant-parameter -->
     <attribute name="name" type="string" use="required"/>
     <!-- The value of the constant-parameter -->
     <attribute name="value" type="string" use="required"/>
</complexType>
<!-- The param element is used to indicate information
          about the input or output property associated with the
          action and the mechanism for passing parameters to or from
          the action. It must be placed within a
          wsdl:input or wsdl:output element. -->
<element name="param" type="portlet:paramType"/>
<complexType name="paramType">
<!-- The param element indicates how the parameter is to be
     delivered to the portlet action or retrieved from
     it once the action has executed -->
     <!-- The name of the parameter which will be used on
     action invocation. If omitted, the partname is used.
     This name will also be set as the name of the
     corresponding property. -->
     <attribute name="name" type="string" use="optional"/>
     <!-- The name of the corresponding part. May be omitted only if
          there is a single parameter for the action,
          in which case it will be inferred from
          the associated operation definition -->
     <attribute name="partname" type="string" use="optional"/>
     <!-- Where to place the parameter prior to invoking the action
          (in case of in parameters), or where to look for it
          after invoking the action (in case of out parameters).
          request-attribute, request-parameter, session are
          allowed for both standard and legacy portlets. A value of
          action is allowed for legacy portlets only, and a value
          of render-parameter is allowed for standard portlets only
     -->
     <attribute name="boundTo" type="string"
      default="request-parameter"/>
     <!-- If the parameter is found but its value is null, this attribute
     determines if the parameter is deemed to be present for brokered
     communication purposes. If the value is set to true, the parameter
     is deemed to be present and the null value will be propagated. The
     default is false.
     -->
     <attribute name="presentIfNullValue" type="string"
      default="false" use="optional"/>
```

```
        <!-- The java class for the parameter.
             The default is java.lang.String.
             If a non-default value is provided, it will be
             used in conjunction with type and namespace to
             restrict matches
        -->
        <attribute name="class" type="string"
         default="java.lang.String"/>
        <!-- A short name for the parameter which is displayed
             to the user. If a message resource file is
             provided for the portlet,
             the value is used as a key to retrieve a
             translated string -->
        <attribute name="caption" type="string" use="optional"/>
        <!-- A description for  the parameter which is displayed to the
             user. If a message resource file is provided for the
             portlet, the value is used as a key to retrieve a
             translated string -->
        <attribute name="description" type="string" use="optional"/>
    </complexType>
</xsd:schema>
```

Interoperability between JSR 286 portlet events and JSR 168 cooperative portlets:

By concept, cooperative portlets are similar to JSR 286 portlet events. Both concepts describe publish/subscribe communication patterns that are based on typed information that is published and received by portlets and propagated via communication links.

Because the concepts are quite similar, the portal supports data exchange between JSR 168 cooperative portlets and JSR 286 portlets that support events. This means that you can extend an existing setup using JSR 168 cooperative portlets with new JSR 286 portlets and that you can smoothly migrate individual cooperative portlets to the new API without losing communication possibilities.

The following table lists the similarities:

Table 461. Similarities between events and cooperative portlets

Aspect of similarity	Cooperative portlets	JSR 286 events
Event declaration	in a WSDL deployment descriptor	as part of `portlet.xml`
Sending data	as a side effect of action processing, for example by setting a request attribute that is declared as an output property	by using the portlet API `setEvent()` call on an action or event response
Receiving data	as a portlet action with input properties, for example, request parameters	by using the portlet API `processEvent()` callback

Writing interoperable portlets

In both models, portlets declare their external inputs and outputs so that they can be matched for communication. The outputs and inputs are matched based on an XML qualified name, which is associated with an XML namespace to make it globally unique. For JSR 286 portlets, the matching information is denoted by the event name while for cooperative portlets, the matching is based on XML data

type declared or referenced in the WSDL. In addition, inputs and outputs carry data that has a specific Java language data type.

Consequently, it is possible to connect a cooperative portlet action to a JSR 286 processing event if these two conditions are met:

- The XML type name of the action's output property matches the name or an alias of the processing event.
- The Java type of the output property is the same as the Java type of the procesing event payload.

You can connect JSR 286 publishing event to a cooperative portlet action if these two conditions are met:

- The name or an alias of the publishing event matches the XML type name of the action's input property.
- The Java type of the publishing event payload is the same as the Java type of the input property.

These rules also have the following implications:

- A JSR 286 event without an associated Java type (no `<value-type>` declaration) cannot be connected with a cooperative portlet.
- Exchanged data types must be primitive types or they must be Java and JAXB serializable (because JSR 286 requires this) and deployed in a shared class loader (because cooperative portlets do not support cross-class loader marshalling).

Example for migrating deployment descriptors

The following cooperative portlet WSDL declares a portlet action **OrdersForMonth** that takes a month name as input and produces an order ID and a customer ID as outputs.

```
   <definitions name="OrderDetail_Service"
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  targetNamespace="http://www.ibm.com/wps/c2a/examples/shipping"
  xmlns:shipping="http://www.ibm.com/wps/c2a/examples/shipping"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<types>
  <xsd:schema targetNamespace="http://www.ibm.com/wps/c2a/examples/shipping">
    <xsd:simpleType name="MonthType">
      <xsd:restriction base="xsd:string"/>

    </xsd:simpleType>
    <xsd:simpleType name="OrderIDType">

      <xsd:restriction base="xsd:string"/>

    </xsd:simpleType>
    <xsd:simpleType name="CustomerIDType">

      <xsd:restriction base="xsd:string"/>

    </xsd:simpleType>
  </xsd:schema>
</types>
<message name="OrderMonthRequest">
  <part name="order_month" type="shipping:MonthType"/>
</message>
<message name="OrderMonthResponse">
  <part name="order_Id" type="shipping:OrderIDType"/>
  <part name="customer_Id" type="shipping:CustomerIDType"/>
</message>
<portType name="OrderMonth_Service">
  <operation name="order_Month">
     <input message="shipping:OrderMonthRequest"/>
     <output message="shipping:OrderMonthResponse"/>
  </operation>
</portType>
```

```
<binding name="OrderMonthBinding" type="shipping:OrderMonth_Service">
  <portlet:binding/>
  <operation name="order_Month">
    <portlet:action name="OrdersForMonth" type="standard"/>
    <input>
      <portlet:param name="month" partname="order_month" class="java.lang.String"/>
    </input>
    <output>
      <portlet:param name="orderId" partname="order_Id"/>
      <portlet:param name="customerId" partname="customer_Id"/>
    </output>
  </operation>
</binding>
</definitions>
```

The containing portlet can be converted to a JSR 286 portlet that exposes the same external behavior. Each action that has an input property becomes a processing event that can be sent to the portlet. Each output property of an action becomes a publishing event that can be emitted by the portlet. The wiring behavior can be maintained by declaring an alias for each event that has the same name as the WSDL data type of the corresponding input or output parameter: For example, the `customerId.published` event declares an alias `shipping:CustomerIDType` so that it can be wired to any processing event or portlet action that takes a customer ID as input.

```
<portlet-app xmlns="http://java.sun.com/xml/ns/portlet/portlet-app_2_0.xsd"
xmlns:shipping="http://www.ibm.com/wps/c2a/examples/shipping"
version="1.0">
<portlet ...>
...
   <supported-processing-event>
      <qname>shipping:OrdersForMonth</qname>
   </supported-processing-event>
   <supported-publishing-event>
      <qname>shipping:orderId.published</qname>
   </supported-publishing-event>
   <supported-publishing-event>
      <qname>shipping:customerId.published</qname>
   </supported-publishing-event>
</portlet>
<event-definition>
  <qname>shipping:OrdersForMonth</qname>
  <alias>shipping:MonthType</name>
  <value-type>java.lang.String</value-type>
</event-definition>
<event-definition>
  <qname>shipping:orderId.published</qname>
  <alias>shipping:OrderIDType</name>
  <value-type>java.lang.String</value-type>
</event-definition>
<event-definition>
  <qname>shipping:customerId.published</qname>
  <alias>shipping:CustomerIDType</name>
  <value-type>java.lang.String</value-type>
</event-definition>
</portet-app>
```

Note:

- The fact that the cooperative portlet properties use `java.lang.String` as data type and that the corresponding events must be declared with `<value-type>java.lang.String` depends only on the `class="java.lang.String"` attribute of the <portlet:param>WSDL element. If no class attribute is present, `java.lang.String` is also used as default value. The XML type declarations in the WSDL are not used to determine the Java type of a property.
- The declaration of cooperative portlet actions does not allow multiple input parameters. Actions or events that require multiple inputs must instead be declared with a single compound input property or event payload that combines all required data.

Triggering communication:

These communication methods are based on directed communication links that pass information from a source portlet to a target portlet.

The portlet programmer defines which information a portlet can send or receive in order to participate in the communication. At run time data is passed by an event broker component across communication links that are created by portal administrators or portal users. When data is sent across a link, the target portlet is explicitly invoked to process the received information and can perform arbitrary updates as a side-effect. This is called the push model. Previously cached markup is discarded as a result of receiving an event.

Message-based communication allows more programmatic and administrative control than shared state, but also creates more overhead. If you need to coordinate many portlets and create many connections, consider using shared state instead.

"Communication with persistent wires"
You can use portlet wires for communication between portlets. Portlet wires are persistent data links.

"Communication with dynamic menus" on page 3092
You can use dynamic menus for communication between portlets.

"Runtime behavior" on page 3092
The portal event broker subsystem provides support for inter-portlet communication with active notifications using portlet events or the cooperative programming model.

Communication with persistent wires:

You can use portlet wires for communication between portlets. Portlet wires are persistent data links.

The following two communication techniques are based on persistent portlet wires. You have to configure the wires in a separate step before the portlets can communicate:

Portlet events

This supports JSR 286 portlets only. The Java Portlet Specification 2.0 defines a model for communication by publishing and subscribing based on portlet events. This model is supported in WebSphere Portal Express Version 8.5. Portlet events can have complex objects as payload, if they provide an XML binding. With portlet events, WebSphere Portal Express Version 8.5 also supports communication links between remote (WSRP V2.0) portlets.

Cooperative portlets

Only JSR 286 portlets support this communication technique. WebSphere Portal Express Version 8.5 still supports the IBM specific cooperative portlet programming API that was provided by previous releases and is known as the Property Broker. However, if you develop new portlets, use the portlet events based on the JSR 286 standard instead. It provides equivalent and compatible functionality. For more information about moving from cooperative portlets to JSR 286 events, refer to Interoperability between events and cooperative portlets.

Note that you can create communication links in both directions between JSR 286 portlets and JSR 168 portlets by using the cooperative portlet API. However, that API does not support communication links between IBM and JSR 168 portlets.

"Portlet wires"
Portlet Wires are used to direct the information flow between portlets that communicate using JSR 286 Portlet events or the WebSphere Portal Cooperative portlets API.

Portlet wires:

Portlet Wires are used to direct the information flow between portlets that communicate using JSR 286 Portlet events or the WebSphere Portal Cooperative portlets API.

A wire connects a JSR 286 publishing event or an output property of a cooperative portlet to a JSR 286 processing event or a cooperative portlet action of another portlet. When the source portlet fires an event or produces a property, and that source event or property has outgoing wires, the information is propagated to the target portlet(s). At the same time the corresponding handler code, `processAction` or `processEvent` is invoked. Conversely, if an event is produced that is not wired to any targets, the event is simply discarded.

Creating wires is a part of page administration and requires appropriate access permissions. It is separated from the portlet development or deployment process, so that the portlet developer does not need to know the actual structure of inter-portlet communication. Communicating portlets can be developed independent of each other, as long as they agree on the same data type and semantics for data exchange. Wires between portlets are usually created by using the portlet wiring tool. It is available as a tab in the Edit Page user interface.

Creating wires: You can create wires only between portlets of the same API type, either standard API or IBM API based; for communication between IBM and Standards-based portlets, you need to use other means, such as cross page links. However, you can create wires between JSR 168 portlets and JSR 286 portlets, This allows you a smooth upgrade from the proprietary collaborative portlet API to the new standard based portlet events.

To create a wire between two portlets, the output that the source portlet declares must match the input that the target portlet declares. This match can be on either of the following two levels:

1. The XML name that describes the semantic content of the data:
   a. For JSR 286 events this is represented by the event names declared in `portlet.xml`.
   b. For collaborative portlets, this is represented by the XML name of the property type declared in the collaborative portlet WSDL.
2. The actual Java class representation of the data.

By default, the wiring tool requires a match of the semantic XML name. To avoid the requirement for a coordinated global namespace, JSR 286 portlets can declare multiple alias names for an event. This allows a portlet to connect to multiple other portlets that use different naming conventions. If the semantic XML name in the portlet section of the `portlet.xml` ends with a period (.), it works as a wildcard character. You can use it to match other semantic XML names defined in the portlet application section of the `portlet.xml` that have the same prefix. This

matching scheme can be useful for handling a group of similar portlet events that have the same payload format in a uniform way.

In a case where you know that two portlets are using the same data format but have declared different XML names, for example, for common formats such as email addresses, you can switch to matching on payload type, which allows you to create wires between sources and targets regardless of the XML semantic name. This can also be useful for very generic target portlets that can accept any string as input.

In addition to the portlet wiring tool, you can also create wires by using the portal configuration interface (XmlAccess). Note that this interface does not perform any consistency checks on declared output or input data. Of course, the option to create a wire does not necessarily mean that the portlets will communicate as expected. For example, if the portlets make different assumptions about the format of the payload, this leads to run time exceptions when the wire is executed.

Remote portlet support: The portal also allows you to create wires between remote portlets that use the WSRP 2.0 protocol for event transfer. You can wire remote portlets that have been integrated into the portal and placed on portal pages, even if they were consumed from different Producers. You can also wire remote portlets to local standard based portlets.

Payload data for remote events is transported as XML content. Therefore local portlets that want to communicate with remote portlets must either declare event payloads with appropriate XML serialization definitions by using the Java XML Binding framework (JAXB) or process the raw XML strings. If the remote portlet Producer is also a WebSphere Portal Express Version 8.5 or another JSR 286 compliant portal and local and remote portlets are using the same JAXB definitions, the correct XML translations happens automatically.

Communication with dynamic menus:

You can use dynamic menus for communication between portlets.

The following two communication techniques are based on dynamically generated menus that allow portal users to select a target destination for displayed data at run time:

Live text based click-to-action
: This technique works for JSR 168 and JSR 286 portlets only. The portal supports client side creation of dynamic action menus based on live text. This means that source data items on a portal page can be tagged with semantic HTML to serve as menu anchors. You can provide tagged source data by any type of HTML fragment on a page. This includes portlets, themes, or external content inside a rendering portlet. You can also define targets by using semantic HTML. In addition to such targets explicitly coded in HTML, the portal makes portlet events or cooperative portlet actions for portlets on the page automatically available as targets.

Runtime behavior:

The portal event broker subsystem provides support for inter-portlet communication with active notifications using portlet events or the cooperative programming model.

The event broker is based on a loosely coupled publish/subscribe communication model: participating portlets declare the messages or data types that they can publish and consume, but do not need to know about the actual communication structure.

Depending on the programming model that you use, sources and targets are defined and implemented in a different way (refer to the documentation about interoperability between events and cooperative portlets), but on the level of concepts described here, they are treated the same way.

Note: The following explanations apply to standard based portlets.

How messages are distributed between portlets is determined by portlet wires that are defined as part of page editing. Portlet wires link a defined output of one portlet on a page to a defined input of another portlet on the same page or on a different page.

The wiring step is separate from portlet development and deployment. This allows independent development of source and target portlets, as long as both portlets use common data types and semantics for information exchange. Portlets can operate stand-alone, and, as partner portlets for the communication are added by editing pages, they can exchange information and react in a coordinated manner, and thereby improve the user experience. Conversely, as portlets are removed, the remaining portlets still work correctly.

Action/event and render phases: The processing cycle of a portal request is divided into an action/event phase which processes user input and a render phase which generates markup output. The activity of the event broker subsystem occurs during the action/event phase. Event processing is usually initiated by a portlet action that is encoded in the current request URL. For example, this can be a portlet action URL, or a URL generated by a click-to-action menu. For requests that do not require portlet activity, but just produce output, the action/event phase can be skipped entirely.

If the request specifies a portlet action, that action will be run and may itself publish output, either as a JSR 286 event or as a cooperative portlet property. If that output is wired to the input of one or more other portlets, a call to the processing methods of these portlets is put into the event queue. When the first portlet action is completed, the next event is taken from the queue and the target portlet is invoked. In the course of the event processing, the target portlet can trigger further communication calls that are then also queued. This process repeats until the queue is empty.

This allows the synchronization of multiple portlets within a single request-response cycle. For example, in a context of a customer order for some goods, all of the following can happen:

1. Transfer of the order ID to the Order Details portlet.
2. This first step also triggers the transfer of the tracking ID for the order to the Tracking Details portlet
3. The Tracking details portlet in turn triggers the transfer of the customer name that is associated with the order to the Customer Details portlet.
4. Consequently, all three portlets display information that pertains to the same customer order.

To avoid infinite loops, event distribution will also stop when a maximum limit of portlet calls is reached; this is considered an error situation and should be avoided.

Note: Event processing is completely sequential and never nested within one client request; each target event or action is fully processed before the next one is invoked. The portal guarantees that the order in which events are delivered to a single target preserves the order in which the events were published. However, for performance reasons, events for different targets may be re-ordered to minimize context switches.

Cross-page communication: Wires can link source output from a portlet on one page to target events of other portlets on different pages. All targets, whether on the same page or on different pages, are processed within the action/event phase of one request cycle. A redirect to the target page can be sent only after the entire action/event phase has been completed. For details about wiring and cross-page wires, refer to the documentation about portlet wires.

Note: This behavior implies that the target portlet that receives event or action calls from a cross-page wire can be on a different page than the current page. In other words the "current page" at execution time can be different from the page that contains the target portlet. Portlet code that relies on retrieving programmatic information about the current page context during the action/event phase should be prepared to handle this case.

Known issues and restrictions related to standard portlets publish and subscribe mechanisms

Review this information for a list of known issues and restrictions with portlet communication.

The following known issues and restrictions exist with portlet communication.

- The pop-up menu functionality requires browsers with JavaScript on the client.
- Complex Java types can only be transferred between portlets if they are deployed in a shared class loader. Use basic Java types, such as `java.lang.String` or `java.util.HashMap` for compound data. Custom classes must be installed in a IBM WebSphere Application Server shared library available to both portlets or must be directly installed into the directory *`wp_profile_root`*`/PortalServer/config`.

  Note: JSR 286 events with XML bindings allow you to transfer complex data types between portlets in different class loaders. However, to avoid the performance penalty incurred by XML serialization, follow the previous recommendations.
- When you import sample Struts WAR files including the file `pbstrutsExample.war` into Rational Application Developer Version 6, it reports broken links. You can ignore this warning message.
- Rational Application Developer lists WS-I warnings for wsdl:binding errors. WS-I compliance warning messages will be displayed in the Problems view for Click-to-Action enabled portlets. You can ignore them in the Click-to-Action WSDL resources. To prevent these warning messages from being displayed, follow these steps:
  1. Select **Window**, then **Preferences**, then **Workbench**, and then **Capabilities**.
  2. Expand **Web Service Developer** and ensure that **Web Services Development** is selected.
  3. Click **Apply**.
  4. Select **Window**, then **Preferences**, then **Web Services**, and then **WS-I Compliance**.
  5. Set the compliance levels to **Ignore** compliance.

6. Click **Apply**.

Important: To remove the compliance warnings from the Problems view open the project's pop-up menu in the Project Explorer and select Run Validation.

- If multiple cross-page wires marked with the `switch page` flag are triggered at the same time, all wire targets are invoked. However, the target page to which a user gets directed cannot be predicted unambiguously.
- Cross-page wires for standard portlets are executed before redirecting to the target page. Therefore cross-page target portlets must not assume that they are executed on the target page.
- Wires created on a root page do not apply to derived pages that you create by referencing the root page. While such pages do inherit the content of the original page, this inheritance does not apply to wires; you have to explicitly create the wires for the derived page.
- Cooperative Java Server Faces (JSF) portlets that you developed by using older versions of Rational Application Developer Version 6 can contain code that does not work with IBM WebSphere Portal Express Version 8.5. If this occurs, replace the file `jsf-portlet.jar` in the failing WAR file with the version in the latest fix pack level.
- Cooperative portlet action declarations with multiple input parameters configured in the WSDL file are not supported.

Special purpose techniques for data exchange

IBM WebSphere Portal Express supports special purpose techniques for data exchange.

Cookies

The Java Portlet Specification 2.0 adds support for portlets to read and set HTTP cookies. You can share cookies between portlets and other Web applications to track information about the current browser interaction without requiring a server side session and without consuming server resources.

Custom JavaScript data exchange

As portlets are Web components inside an HTML page, you can use any client side techniques to read, write or exchange data within the markup that your portlet provides. This includes sharing data via the page document object model (DOM) or use of client side JavaScript code or libraries.

Related information:

http://www-128.ibm.com/developerworks/websphere/library/techarticles/0612_behl/0612_behl.html

http://www-128.ibm.com/developerworks/websphere/library/techarticles/0603_behl/0603_behl.html

Shared portlet sessions

The following communication methods are based on shared state between multiple portlets. This means that two or more portlets read and write to the same data.

There is no direction imposed for the data flow. Portlets are not explicitly called to receive data, but receive data updates implicitly when they read shared information that has been updated (pull model).

The portal supports the following methods for portlet communication based on shared state:

Portlet session sharing
This applies to JSR standard API portlets only. All standard portlets and servlets that are deployed within the same Web application have access to the same Web module HTTP session. Portlet session attributes are normally namespaced per portlet window to avoid accidential collisions, but it is possible to access attributes at the actual HTTP session level by using the `PortletSession.APPLICATION_SCOPE` constant. Portlets that rely on shared session attributes need to be developed and deployed together in a single WAR file.

Public render parameters
This applies to JSR 286 standard API portlets only. The Java Portlet Specification 2.0 for JSR 286 allows portlets to share navigational state information that is stored as render parameters. This method of data sharing is especially useful for coordinating the view state of multiple portlets that display different information items that are all related to the same parameter name, such as a **`customerID`**. In this case, the parameter should be represented as a shared render parameter. Shared render parameters are defined per portlet in the portlet application's `portlet.xml`. A similar common scenario is the coordination between a navigator and a viewer portlet. Public render parameters provide a simple programming model and allow bookmarking of the shared state and **Back** button support. Public render parameters can also be shared with remote portlets via the WSRP V 2.0 protocol. This is supported by IBM WebSphere Portal Express Version 8.5 and later versions.

Note: Information about render parameters is normally encoded into the URL. Therefore their names and values should be as short as possible in order to not exceed the URL length restrictions that many browsers have.

Dynamic user interfaces

Learn about dynamic user interfaces that include dynamic pages, dynamic portlets, dynamic UI configuration, dynamic UI properties, and shared dynamic UIs. Get an overview of how to develop a dynamic UI configuration.

Dynamic user interfaces, or *dynamic UIs*, are portlets or pages that are dynamically created based on the definition of an existing page or portlet definition. A dynamic UI can be launched only by a portlet using the Dynamic UI Manager API. Because of its dynamic nature, the interface is not persisted in the portal database and has a maximum lifetime of the user's session with the portal. The interface can also be closed prior to the end of the session either programmatically or by the user.

Dynamic UIs are suited for applications in which users might need to have several instances of a page or portlet open for multitasking. Consider the travel request scenario in business process integration as an example. If several marketing representatives need to travel to a conference, their manager would receive multiple travel requests. With static pages, the manager must complete one request before proceeding to the next one. Using dynamic UIs, the manager can open several requests simultaneously and navigate between these and other pages in the portal. Business process integration is an example of a *dynamic UI configuration*.

Dynamic pages and portlets contain many of the same properties as static pages and portlets. For example, the user can navigate between static and dynamic

pages, or change the window state of a dynamic portlet. However, dynamic UIs are not stored in the portal database, so they have less impact on server performance. Therefore, dynamic UIs cannot be administered. For example, you cannot view or update a dynamic page using **Manage Pages** or assign a unique name to a dynamic portlet or page.

A dynamic UI is independent of its point of origin. For example, if a portlet launches a dynamic page, the launched page is maintained even if the originating portlet or page is deleted. A dynamic UI is a copy of a page or portlet definition at the time the instance is created and is not affected by subsequent changes to the definition. By default, the dynamic UI acquires the title and description of its page or portlet definition. However, it is possible to programmatically overwrite the title and description when the dynamic UI is launched.

Dynamic pages

A dynamic page is an instance of an existing static page, called a *page definition*. The page definition serves as a template from which all dynamic instances are created. Users must have view rights to the page definition in order to open a dynamic instance of that page. The page definition can be created by the administrator using XMLAccess or **Manage Pages**. The dynamic page copies the layout and content (containers, portlets, and wires) and portlet preferences provided by the page definition. Changes to the page definition do not affect the layout or content of dynamic pages that have already been launched. The layout and content of the dynamic page instance cannot be customized.

When a portlet launches a dynamic page, the new page instance appears in the navigation under a node created just for containing dynamic pages. This node is called an *extension node*. An example of an extension node is the container for task pages for business process integration. A single portal can have multiple extension nodes, one for each dynamic UI configuration. However, extension nodes are intended to hold only dynamic pages. Static nodes added under the content topology of an extension node are not visible. Dynamic pages are not visible to anonymous users, even if anonymous users have view rights to the page definition.

When a dynamic page is closed which a user currently has selected, the user is automatically redirected to another open dynamic page in the same configuration. If no sibling page exists, the redirection is determined by one of the following.

1. The page indicated by the setDefaultRedirectPage() method of the DynamicUIManagementFactoryService interface.
2. If setDefaultRedirectPage() is not used, the user is redirected to the page indicated by the defaultRedirectPage property in the `DynamicUIManagerFactoryServiceImpl`. This property is prefixed by the unique name of the extension node for the dynamic UI configuration. For example, two dynamic UI configurations could set this property as follows.

   ```
   my.dynui.config.test1.defaultRedirectPage=page1.unique.name
   my.dynui.config.test2.defaultRedirectPage=page2.unique.name
   ```

3. If the defaultRedirectPage property is not set, the user is redirected to the extension node for the configuration.

In most cases the page definition should be hidden from users by placing it in a sub tree of the content hierarchy that is not accessible by the portal navigation. In the business process configuration, for example, task page definitions are created under a **Task Page Definitions** node that is accessible only to the administrator.

Dynamic portlets

Dynamic portlets include portlets on a dynamic page and portlets that are dynamically added to the page. Dynamic portlets can receive properties from the portlet that launches the dynamic UI. The task processing portlet, which is a component of business process integration, is an example of a dynamic portlet.

When a portlet is dynamically added to an existing dynamic page, it is added to special containers that have been designated as launch areas. Each dynamic portlet that is added to the page is placed in the column with the least number of portlets. If all of the content of a dynamic page is locked, any attempt to launch a dynamic portlet on that page throws an `AddUIElementException`.

Note:

- It is possible for a user to access edit mode for a dynamic portlet. However, the preferences are not preserved (cannot be persisted).
- For a user to launch a dynamic portlet, the user must have view rights to the portlet definition from which the dynamic portlet would be launched.
- Dynamic portlets cannot be moved on a dynamic page.

Dynamic UI configuration

A dynamic UI configuration includes the settings and software components of a portal that support different application frameworks for dynamic UI. For example, a portal site can take advantage of the configuration provided by business process integration as well as support another dynamic UI configuration for a different purpose. The launching portlet can invoke a specific dynamic UI configuration by the unique name assigned to its extension node. Dynamic pages are assigned to the extension node for the configuration for which it is created.

Dynamic UI properties

Upon launch, a portlet can pass properties to a dynamic UI using the property broker API. Properties can be set during or after launching the dynamic UI. The purpose of these properties is to convey any information to the target portlets needed to fulfill their task, for example, a calendar entry to be processed or a document to be opened.

Properties passed to a dynamic page are available to all portlets on the page. Properties passed to a dynamic portlet are available only for that portlet. However, the dynamic portlets must declare their intention to receive dynamic properties using a preference setting in their descriptor. For example, in a business process configuration, the My Tasks portlets sends the TaskID, ReturnPageID. and TaskUIHandle properties to the dynamic page. The task processing portlet sets the **`com.ibm.portal.pagecontext.enable`** preference to `true` so that it will receive these properties. For IBM portlets, this is set as a configuration parameter. Also, IBM portlets must modify the servlet class entry of the `web.xml` file, specifying the com.ibm.wps.pb.wrapper.PortletWrapper class.

If page properties are set multiple times before the context is delivered, only the latest parameter set is distributed to the portlets on the target page.

Shared dynamic UIs

A *shared dynamic UI* is a dynamic page or dynamic portlet in which only one shared instance can exist for a user at a given time. A portlet can explicitly launch a shared portlet or shared page using the addSharedPage() or addSharedPortlet() methods. When a shared dynamic UI instance already exists, it is reused. If not, a new instance is created and marked as either the shared portlet for a given page, or the shared page for a dynamic UI instance.

If a dynamic UI is launched as shared, subsequent launching of the UI using the non-shared methods (addPortlet() and addPage()) creates a new instance. To ensure the same shared instance is used, the launching portlet must consistently use the shared methods.

The scope of a shared portlet or shared page instance is restricted to a dynamic UI configuration. This means, for example, that a portlet can add a shared dynamic page in dynamic UI configuration A. When another portlet also tries to create a shared instance in dynamic UI configuration B, a new one is created.

"Overview: Developing a dynamic UI configuration"
Get an overview of the main tasks involved in creating a dynamic UI configuration.

Overview: Developing a dynamic UI configuration

Get an overview of the main tasks involved in creating a dynamic UI configuration.

About this task

The following describes a general overview of the main tasks involved in creating a dynamic UI configuration.

- Planning
- Creating the extension node for containing dynamic pages
- Developing the portlet definition for dynamic portlets
- Developing the page definition for dynamic pages
- Developing the dynamic UI launching portlet
- Providing controls for closing dynamic UIs.
- Testing the configuration
- Deploying the configuration

Note: Only standard portlets can launch dynamic UIs. IBM portlets, however, can be launched as dynamic portlets and receive properties. Standard portlets can also launch dynamic pages that contain a mixture of standard and IBM portlets.

Procedure

1. Planning. Determine the requirements of the dynamic UI configuration. The following are some of the questions that can help you in the planning process.
   - What is the layout of the page definition?
   - Should the page definitions be viewable by users? If not, where in the portal topology should they be placed?
   - Where should the extension node be placed in the portal topology?
   - What information needs to be passed to each dynamic UI? How will the portlets use that information?

- How will the UI be closed? Should you use tags to allow the user to close dynamic pages and dynamic portlets? If not, is it the responsibility of the calling portlet or a dynamic portlet to close the UI?

2. Creating the extension node for dynamic pages.
   a. Create the page to be used for containing dynamic UIs.
   b. Assign a unique name to the node. To do this, use the Unique names portlet.
   c. Open a command prompt.
   d. Change to the *wp_profile_root*/ConfigEngine directory.
   e. Run the following configuration task:
      - Linux: `./ConfigEngine.sh action-enable-page-as-extension-node-wp.dynamicui.config -DPageUniqueName=`*`pageUniqueName`* `[-DVirtualPortalContext=`*`virtualPortalContext`*`]`
      - IBM i: `ConfigEngine.sh action-enable-page-as-extension-node-wp.dynamicui.config -DPageUniqueName=`*`pageUniqueName`* `[-DVirtualPortalContext=`*`virtualPortalContext`*`]`
      - Windows: `ConfigEngine.sh action-enable-page-as-extension-node-wp.dynamicui.config -DPageUniqueName=`*`pageUniqueName`* `[-DVirtualPortalContext=`*`virtualPortalContext`*`]`

      where *pageUniqueName* is the unique name of the extension node and *virtualPortalContext* is the context of the virtual portal that the specified page is part of. The `-DVirtualPortalContext` flag is needed only in the case where a virtual portal is used. This command designates the page as an extension node. If you subsequently need to remove the extension node designation, run the command using the `action-disable-page-as-extension-node-wp.dynamicui.config` task.

3. Developing the portlet definition for dynamic portlets. Any existing portlet can be used as the definition for a dynamic portlet. For standard portlets, be sure to set the ID attribute for the `<portlet-app/>` element in `portlet.xml`. The launching portlet uses this value, plus the `<portlet-name/>`, to obtain the object ID of the portlet definition. For IBM portlets, the uid attribute of the `<portlet-app/>` element is used. Determine what properties need to be passed to the portlet.

   Portlet deployment descriptor considerations

   To receive portlet properties, the **`com.ibm.portal.context.enable`** preference parameter must be set in the `portlet.xml` with a value of `true`. By default, this value is set to false.

   - For standard portlets:

```
<portlet-preferences>

   ...
    <preference>
    <name>com.ibm.portal.context.enable</name>
    <value>true</value>
   </preference>
   ...
</portlet-preferences>
```

   - For IBM portlets

```
...
   <config-param>
      <param-name>com.ibm.portal.context.enable</param-name>
      <param-value>true</param-value>
   </config-param>
...
```

Additionally, you must also modify the file `web.xml` for IBM portlets to receive portlet properties. The servlet class entry should specify the com.ibm.wps.pb.wrapper.PortletWrapper class. See"Packaging, deploying and compiling cooperative portlets" on page 3079 for more information and an example.

Considerations for retrieving context

Standard portlets receive page properties on the processAction() method using the com.ibm.portal.context request attribute. The value of this attribute is a Map storing the context entries. Each property value is obtained by name. The name and value of each property value is determined by the requirements of your dynamic UI configuration. Use the following code sample to get a value for the property with the name **propertyName**:

```
public void processAction (ActionRequest request, ActionResponse response)
            throws PortletException, java.io.IOException
{

   // perform application specific action handling
   ...

   // perform page context processing

   String specialAction = request.getParameter("com.ibm.portal.action");

   if (specialAction != null &&
       specialAction.equals("com.ibm.portal.context.receive"))
   {
      //this indicates context was passed to the launched page
      java.util.Map contextMap = (java.util.Map)
                    request.getAttribute("com.ibm.portal.context");

      Object propertyValue =  (Object) contextMap.get(<propertyName>);

      portletSession.setAttribute(<propertyName>, propertyValue);

   }

}
```

IBM portlets must implement the PropertyListener interface that provides the setProperties() method, which provides the page properties as an array of PropertyValue objects.In this sample, a loop is used to scan the array for the task properties.

```
public void setProperties(PortletRequest request, PropertyValue contextArray[])
{

   Object propertyValue;

   for (int i = 0; i < contextArray.length; i++)
   {
      String propertyName = contextArray[i].getProperty().getName();

      if(propertyName.equals(<propertyName>))
      {
         propertyValue = (Object)contextArray[i].getValue();
      }

   }

   request.getSession().setAttribute(<propertyName>, propertyValue);

}
```

IBM portlet considerations

To receive portlet properties, the **com.ibm.portal.context.enable** configuration parameter is specified in the portlet.xml. This parameter plays the same functional role as the standard portlet preference. Additionally, you must also modify the `web.xml` file to receive portlet properties. The servlet class entry should specify the com.ibm.wps.pb.wrapper.PortletWrapper class. See the "Packaging, deploying and compiling cooperative portlets" on page 3079 section for more information.

4. Developing the page definition for dynamic pages.
   a. Create the layout and content of the page definition from which dynamic pages are launched. For development purposes, you can use Manage Pages as you work with the visual layout.
   b. Assign a unique name to the page using the Unique names portlet. The portlet uses this name to lookup the object ID of the page definition.
5. Developing the dynamic UI launching portlet
   a. Using JNDI lookup, obtain a reference to the required portlet services.

```
private DynamicUIManagementFactoryService dynamicUIManagerFactoryService;
private RedirectURLGeneratorFactoryService redirectService;
PropertyFactory propertyFactory;
...

// Obtain a reference to the Dynamic UI Management Factory service
PortletServiceHome dynamicUIManagerFactoryServiceHome = (PortletServiceHome)
   ctx.lookup("portletservice/com.ibm.portal.portlet.service.dynamicui.DynamicUIManagementFactoryService");
dynamicUIManagerFactoryService = (DynamicUIManagementFactoryService)
   dynamicUIManagerFactoryServiceHome.getPortletService(DynamicUIManagementFactoryService.class);

// Obtain a reference to the property factory
PortletServiceHome serviceHome = (PortletServiceHome)
   ctx.lookup("portletservice/com.ibm.portal.propertybroker.service.PropertyFactory");
   propertyFactory =
     (PropertyFactory)serviceHome.getPortletService(com.ibm.portal.propertybroker.service.PropertyFactory.class);

// If the dynamic UI should be displayed immediately upon launch,
// obtain a reference to the RedirectURLGeneratorFactory service
   PortletServiceHome redirectServiceHome = (PortletServiceHome)
      ctx.lookup("portletservice/com.ibm.portal.portlet.service.state.RedirectURLGeneratorFactoryService");
   redirectService = (RedirectURLGeneratorFactoryService)
   redirectServiceHome.getPortletService(RedirectURLGeneratorFactoryService.class);
```

   b. Obtain the object ID of the page definition or portlet definition. This can be done using either a unique name (for a page or portlet definition) or portlet name + portlet-app ID (portlet definition only).

Using a unique name:

```
Context ctx = new InitialContext();

...

Name uniqueName = new CompositeName("portal:uniquename");

uniqueName.add(yourUniqueName);
ObjectID oidForName = (ObjectID) ctx.lookup(uniqueName);
```

Using portlet name + porlet-app id:

```
Context ctx = new InitialContext();

...

Name portletName =
     new CompositeName("portal:config/portletdefinition");
portletName.add(appID);
portletName.add(portletName);

ObjectID portletDefOID = (ObjectID) ctx.lookup(portletName);
```

c. Obtain an instance of the DynamicUICtrl interface from the factory. The factory expects the Render/ Action Request/Response followed by the configuration name (String) as input parameters. The DynamicUICtrl that is returned is parameterized with the request/response. The DynamicUICtrl must be obtained once per request and should not be stored.

```
DynamicUICtrl DynamicUICtrl =
    dynamicUIManagerFactoryService.getDynamicUICtrl(request, response, extensionNode);
```

d. Create the properties that need to be passed to the dynamic UI.

```
PropertyController property1 = propertyFactory.createProperty(config);
property1.setName(propertyKey);
property1.setClassname("java.lang.String");
property1.setDirection(Direction.OUT);

PropertyValue value =
    propertyFactory.createPropertyValue(request, property1, propertyValue);
PropertyValue[] propertyValues = new PropertyValue[1];
propertyValues[0] = value;
```

e. Launch the dynamic UI using the addPage() or addPortlet methods and passing the object ID of the page/portlet definition. Example of launching a page:

```
DynamicUICtrl.addPage(pageDefinitionID,
                new LocalizedImpl(title,description), propertyValues);
```

f. Optional: Navigate the user to the newly launched page using a redirect.

```
RedirectURLGenerator redirector =
                    redirectService.getURLGenerator(request, response);
EngineURL redirectURL = redirector.createPortletURL(launchedPortlet);

response.sendRedirect(redirectURL.toString());
```

6. Providing controls for closing dynamic UIs. In many cases you can allow the user to explicitly close a dynamic UI from a theme or skin. These icons are enabled by using the `<portal:closePage/>` and `<portal:closePortlet/>` tags. See "Tags used by the portal JSPs" on page 2790 for more information.

   As an alternative, the launching portlet can close a dynamic UI using the removePage() or removePortlet() methods of the DynamicUICtrl interface, providing the object ID of the dynamic UI as input on the call. If removing a dynamic portlet, the calling portlet must be on the same page.

7. Testing the configuration. Each dynamic UI configuration has its own requirements, design, and implementation, and therefore, a different set of test cases. However, you should always verify the following general behaviors.
   - Pages and portlets are created and removed successfully.
   - Each time a dynamic UI is launched, a new instance of the page or portlet is created or, in the case of shared dynamic UIs, the user is returned to an existing instance.
   - Properties are passed and processed by the dynamic portlets as expected.
   - As portlets are added and removed from a dynamic page, the page maintains a balanced layout.
8. Deploying the configuration When the dynamic UI configuration is ready to be promoted to a production level server, create the XMLAccess scripts that contain the necessary page and portlet definitions. You can use the Manage Pages portlet to export the page configuration to XMLAccess, including any wires or unique names used by the configuration.

Collaborative Services API and the person tag

Collaborative Services are a set of methods and JavaServer Page tags that allow developers who are writing portlets for WebSphere Portal Express or other

application servers to add Lotus collaborative functionality to their portlets. The services can be used to develop new custom portlets, or to add collaborative functionality (for example, menus or person links indicating online status) to existing portlets.

The Collaborative Services include a JavaServer Page tag language descriptor (TLD) for a **person** tag.

When added to your custom portlet, the person tag causes people's names to appear as hyperlinks, and the **Click here for Person Card** option to display when the user moves the cursor over an active (underlined) person's name. Clicking this option displays the Person card. If WebSphere Portal Express cannot identify the person name, it displays the name as plain text and the **Click here for Person Card** option is not available.

By default, the Person card includes the action **Profile**. The **Send Mail** action displays if the user has an email address. If Sametime is installed and enabled to work with the portal, the person tag adds an icon that indicates the person's online status and additional actions:

- **Chat**
- **Add as Sametime Contact**

Note: The **person** tag provides only default actions, but you can add your own custom actions to the person menu in any portlet.

You can configure the amount of time that the Person card displays by modifying the **personTagTimeout** custom property.

You can also customize the Person card so that only the business card fields display by default, letting users choose whether to expand the information shown.

The tag library for Collaborative Services that includes the **person** tag is installed on the portal server in the following locations:

Windows
PortalServer_root\people\people.impl\persontag\taglib\shared\app\WEB-INF\tld

Linux *PortalServer_root*/people/people.impl/persontag/taglib/shared/app/WEB-INF/tld

IBM i *PortalServer_root*/people/people.impl/persontag/taglib/shared/app/WEB-INF/tld

Note: You do not need to copy or move the `.tld` file anywhere within your portal project; you need only refer to its location in your portal installation. You need a reference to it in the JSP file for every portlet that you deploy that uses the person tag.

"Integrating the Business card and online status in a custom portlet" on page 3105
If IBM WebSphere Portal Express is configured to work with IBM Sametime, you can integrate the business card and online awareness in a custom portlet. Person names then appear with a dynamic status indicator. Users of the portlet can move the cursor over the name of an active person and then click **Click for Business Card**.

Integrating the Business card and online status in a custom portlet

If IBM WebSphere Portal Express is configured to work with IBM Sametime, you can integrate the business card and online awareness in a custom portlet. Person names then appear with a dynamic status indicator. Users of the portlet can move the cursor over the name of an active person and then click **Click for Business Card**.

Before you begin

The Business card includes person details in a collapsible section. If a photo of the person is available in the repository, the Business card also displays the photo. The photo enlarges when you hover over it with the mouse. The Business card also includes options for viewing the profile of a person, chatting, and more.

The Business card is provided by using the person tag of the Collaborative Services API. There are two methods to integrate the Business card, using the Person tag of the collaborative API and using the live text microformat.

Procedure

Integrating the Business card and online status in a custom portlet

1. Add the following line to your page:
   ```
   <%@taglib uri="/WEB-INF/tld/people.tld"prefix="pa"%>
   ```
2. Include a statement in your page similar to the following line:
   ```
   <pa:person value="CN=John Smith,OU=SALES,O=ACME" valueType="LDAPDN" displayName="John Smith" isActive="true/false" />
   ```
   The *displayName* and *isActive* attributes are optional. The *isActive* attribute controls the formation of the Business card for a users name, depending on the activity of the user.
   - If a user is active and selectable, then the *isActive* attribute is true.
   - If a user is inactive and non-selectable, then the *isActive* attribute is false.

The supported types for *valueType* are:

- EMAIL

  ```
  <pa:person value="wpsadmin@acme.com" valueType="EMAIL" displayName="John Smith"  isActive="true
  ```

- LDAPDN

  ```
  <pa:person value="CN=John Smith,OU=SALES,O=ACME" valueType="LDAPDN" displayName="John Smith" is
  ```

- MEMBERDN

  ```
  <pa:person value="CN=John Smith,OU=SALES,O=ACME" valueType="MEMBERDN" displayName="John Smith"
  ```

- OBJID

  ```
  <pa:person value="Z9eAe1JRU6N5FDRRANP14GRR47Q5CC38ANPLCI3" valueType="OBJID" displayName="John
  ```

- CN

  ```
  <pa:person value="wpsadmin" valueType="CN" displayName="John Smith"&nbp; isActive="true/false"
  ```

- WMMID

  ```
  <pa:person value="wpsadmin" valueType="WMMID" displayName="John Smith"  isActive="true/false" /
  ```

Where OBJID corresponds to the *objectId* of the user.

Integrating the Business card using live text microformat.

3. Add the following lines to your page:
   a. Business card with online status:

      ```
      <span class='vcard X-sametime-resolve'>
      ```

   b. Business card without online status:

      ```
      <span class='vcard'>
      ```

4. Include this line to add a user display name.

   ```
   <a class='fn' href='javascript:SemTagMenu.a11y(event)' style='color:black; text-decoration:none;'
      onclick='return false;'>{displayName}</a>
   ```

5. Any of the class `'email'`, `'uid'` and `'objectId'` are sufficient as long as they resolve to a unique user.

   ```
   <span class='email' style='display:none;'>{email id}span class='email' style='display:none;'>{email id}</span>

   <span class='uid' style='display:none;'>{LDAPDN}span class='uid' style='display:none;'>{LDAPDN}</span>

   <span class='userObjectId' style='display:none;'>{objectId}span class='userObjectId' style='display:none;'>{objectId}
   </span></span>
   ```

Example

In the following example, `<%@taglib uri="/WEB-INF/tld/people.tld" prefix="pa" %>` interacts with Collaborative Services. The first % statement is a reference to the Collaborative Services tag library. The `pa:person value="wpsadmin@acme.com" valueType="EMAIL"` statement uses the Collaborative Services person tag to display the name of a person as a live link.

```
<%@taglib uri="/WEB-INF/tld/people.tld" prefix="pa" %>
<%@taglib uri="/WEB-INF/tld/portlet.tld" prefix="portletAPI" %>
<portletAPO:init/>
Hello example
<h1>Collaborative Services Hello JSP</h1>
<br />
<h2>Hello,<pa:person value="wpsadmin@acme.com" valueType="EMAIL" /> !</h2>
```

Customizing Person card actions through the theme

You can use the theme to add items to the Person card's **More actions** menu in any portlet that uses the AJAX person tag.

Before you begin

About this task

Procedure

1. Locate the theme file which contains the <body></body> tags of the page. This is most often Default.jsp, or theme.html. For more information on locating your theme's files, see Location of theme resources.
2. Open the file in a text editor.
3. To add a menu item, add XML code to the end of the file, just before </body></html> tags. The following sample code adds a menu item called **Test Action**.

```
<div class="com.ibm.portal.action" style="display:none;">
 <span class="action-id">test.action1</span>
 <span class="action-impl">/javascript/TestAction.js</span>
 <span class="action-context">person</span>
 <span class="action-label">Test Action</span>
 <span class="action-description">This is a test for extending Person menu</span>
 <span class="action-showif">TestAction.showif</span>
 <span class="action-url">javascript:TestAction.execute(@@@ARGS@@@)</span>
 <span class="action-order">0</span>
</div>
```

4. Save the changes to the file.
5. Create a JavaScript file that executes the action you want to occur when the associated menu item is selected. The following sample code generates an alert when the menu item **Test Action** is selected.

```
var TestAction = {
  showif: function(person) {
    return true;
  },
  execute: function(person) {
    alert("TestAction executed for: " + person.fn);
  }
}
```

   Paste the JavaScript file (TestAction.js) in a directory where it can be accessed by a URL. For example you could put the TestAction.js file in either a javascript directory in the root of your web server (/javascript/TestAction.js) or a directory in your theme WAR file */yourthemecontext*/javascript/TestAction.js.
6. If the file you modified in step 1 was not Default.jsp, update the file's timestamp by making a change, removing the change, and then saving the file.
7. To verify that the new action was added, open a portlet that supports people awareness (for example, the People Finder) and then view the Person card for a selected user. Verify that the option you added (for example **Test Action**) is available from the **More actions** menu and that it works as expected.

Setting display duration for the Person card

You can configure the Person card to display longer than the default number of milliseconds by modifying the personTagTimeout custom property in the WebSphere Integrated Solutions Console.

Before you begin

About this task

Procedure

1. Select the appropriate console, depending on your environment:
   - If running stand-alone, use the local WebSphere Integrated Solutions Console.
   - If running in a cluster, use the console of the Deployment Manager.
2. Start the WebSphere Integrated Solutions Console by entering the URL in the location field of a Web browser:

   `http://example.com:admin_port/ibm/console`

   where *example.com* is the name of your server and *admin_port* is the port assigned to the WebSphere Integrated Solutions Console.
3. In the navigation, click **Resources** > **Resource Environment** > **Resource Environment Providers**.
4. Locate and click the resource **WP PeopleService**.
5. Under Additional Properties, click **Custom Properties**.
6. Locate **personTagTimeout** and then set its value in milliseconds.
7. Click **Apply** and then save the settings.
8. Restart the portal server.

What to do next

Related concepts:

"People awareness" on page 959
People awareness makes people's names appear as hyperlinks that users can click to display information about the individual and gain access to actions for contacting and working with him or her. If the administrator has configured an IBM Sametime server to work with WebSphere Portal Express, users can see each other's online presence in their person links according to the status options they have set in their Sametime client (for example, whether the person is active, away, offline, or does not want to be disturbed). The person's online status appears only if Sametime is enabled.

Customizing the look of the Person card

You can change elements of the **Click for Person Card** option and the appearance of the Person card such as text color, font, background color, and so on, by modifying Cascading Style Sheet (CSS) definitions specified in the `styles_people.jspf` file located in the theme directory.

Before you begin

About this task

Procedure

1. Locate your theme's Cascading Style Sheet (CSS) files. The location can differ between themes, but the CSS files are often in a `/css` or `/styles` folder within your theme. For more information on locating your theme's files, see Location of theme resources.
2. Locate and open the CSS file that contains the Person Card styles. The file may differ by theme, but is often either `styles_people.jpsf`, `styles_ibm.jspf`, or `portalLegacy.css`. You can locate the specific file by searching file contents of

the CSS folder for one of the CSS style definitions listed in the following table (for example, `personMenu`), or by loading the portal theme in a Web browser and using Web development tools to inspect the CSS style definitions loaded in the page.

3. Find the following style definitions in the file and modify them as needed:

Modify this style definition	To do this
`.menu_drop_icon`	Change the look and feel of the **Click for Person Card** option.
`.hyperlink`	Change the appearance of the person name hyperlink.
`.photoCard img`	Change the style of the image that displays in the business card section of the Person card.
`.businessCard .cardname`	Change the style of the name that displays in the business card section.
`.businessCard li`	Change the style of other user details that display in the business card section.
`.personMenuActions`	Change the style of the actions such as **Profile** and **Send Mail** that display on the Person card.
`.personMenu`	Change the style of the container that holds the business card and action items as a single unit.

4. Touch the timestamp on the theme's `styles.jsp` so that it will be recompiled by the JSP compiler. You can touch the timestamp by editing the file, adding a blank line, and saving the file.
5. Clear the browser's cache and cookies.
6. Call the Person card and verify your changes.

What to do next

Making business card fields expand and collapse

You can configure the Person card to display the business card fields for a selected name. Users can click a small control to expand the rest of the profile or hide it from view. You enable or disable this control by modifying the **showExpandableSection** custom property in the WebSphere Integrated Solutions Console.

Procedure

1. Select the appropriate console, depending on your environment:
   - In a stand-alone environment, use the local WebSphere Integrated Solutions Console.
   - If running in a cluster, use the console of the Deployment Manager.
2. Start the WebSphere Integrated Solutions Console by entering the URL in the location field of a web browser:

   `http://`*`example.com`*`:`*`admin_port`*`/ibm/console`

   Where *example.com* is the name of your server and *admin_port* is the port assigned to the WebSphere Integrated Solutions Console.
3. In the navigation, click **Resources** > **Resource Environment** > **Resource Environment Providers**.

4. Locate and click the resource **WP PeopleService**.
5. Under Additional Properties, click **Custom Properties**.
6. Locate the custom property **showExpandableSection** and set the value as needed:

   `true` enables the control, allowing users to click either **Show More** to expand the business card, or **Show Less** to collapse the business card.

   `false` hides the control.
7. Click **Apply** and then save the settings.
8. Restart the portal server.

Logging for Collaborative Services

The Lotus Collaborative Services APIs use the WebSphere JRas facility for logging error information to the WebSphere Integrated Solutions Console or to log files.

Before you begin

For Diagnostic Trace, specify the following value to log information that is related to the Collaborative Services:

```
com.lotus.cs.cslog=all=:com.lotus.ap.portlets.
*=all=:com.ibm.wkplc.people.portal.taglib.
*=finest
```

The trace file can be found in the following location:

- IBM i: *wp_profile_root*`/logs/WebSphere_Portal/trace.log`
- Linux: *wp_profile_root*`/logs/WebSphere_Portal/trace.log`
- Windows: *wp_profile_root*`\logs\WebSphere_Portal\trace.log`

IBM Portlet API

The IBM Portlet API is no longer supported starting with WebSphere Portal Express Version 8.5.0. You must convert portlets based on the IBM Portlet API to the Standard Portlet API. Learn how to convert IBM Portlets to Standard API portlets.

"Converting IBM portlets (IBM i Linux Windows)"
You can convert your basic IBM portlets and IBM portlets that use the Struts Portlet Framework to the standard portlet API.

Related reference:

"Standard portlet API" on page 2934
The Java Portlet Specification addresses the requirements of aggregation, personalization, presentation, and security for portlets running in a portal environment.

Converting IBM portlets (IBM i Linux Windows)

You can convert your basic IBM portlets and IBM portlets that use the Struts Portlet Framework to the standard portlet API.

- "Converting basic IBM portlets to the standard"
- "Converting IBM portlets that use the Struts Portlet Framework" on page 3117

Converting basic IBM portlets to the standard

This topic describes some of the more common changes (but not all) that are required to convert an IBM portlet to a standard portlet. Many conversion tasks

depend on the amount of complexity in the portlet code. You must become familiar with the Java Portlet Specification to determine any remaining changes that are not covered in this topic.

- Changing Java source
- Changing JSP source
- Changing the portlet deployment descriptor

Changing Java source

1. Change import statements to use the standard packages.

   Change this:

   ```
   import org.apache.jetspeed.portlet.*;
   import org.apache.jetspeed.service.*;
   ```

   to this:

   ```
   import javax.portlet.*;
   import com.ibm.portal.portlet.service.*;
   ```

2. Change class inheritance to use GenericPortlet. Notice that the ActionListener is not implemented.

   Change this:

   ```
   public class SamplePortlet extends PortletAdapter implements ActionListener{
   ...
   }
   ```

   to this:

   ```
   public class SamplePortlet extends GenericPortlet{
   ...
   }
   ```

3. Change objects that are used for all render methods. In the standard portlet API, the PortletRequest and PortletResponse define common functions for the RenderRequest and RenderResponse subclasses. These subclasses are the arguments for all implementations of the render() method, including doView(), doEdit(), and doHelp().

   Change this:

   ```
   public void doView(PortletRequest request, PortletResponse response) {
      ...
   }
   ```

   to this:

   ```
   public void doView(RenderRequest request, RenderResponse response)
        throws PortletException, IOException{
        ...
   }
   ```

4. Change the actionPerformed() method. In the standard portlet API, this method is replaced by the processAction() method, which does not require the portlet to implement a listener. The processAction() methods accepts the ActionRequest and ActionResponse as arguments, which extend the PortletRequest and PortletResponse.

   Change this:

   ```
   public void actionPerformed(ActionEvent event) throws PortletException{
   ...
   }
   ```

   to this:

   ```
   public void processAction(ActionRequest request, ActionResponse response)
        throws PortletException, IOException{
        ...
   }
   ```

5. Change how the response content type is set. In the standard portlet API, the MIME type of the output that is returned in the response must be set before including the JSP. IBM portlets declare the MIME type

using the contentType attribute of JSP's page directive. Therefore, this change makes the contentType setting in the JSP unnecessary.

Add the following code before including the JSP:

```
response.setContentType("text/html");
```

6. Change JSP includes. In the standard portlet API, JSPs are included by a request dispatcher's include() method. In the portlets' render method, set the MIME type of the output before returning it in the response.

   Change this:

   ```
   PortletContext context = getPortletConfig().getContext();
   context.include("/jsp/View.jsp", request, response);
   ```

   to this:

   ```
   response.setContentType("text/html");
   PortletContext context = getPortletConfig().getPortletContext();
   context.getRequestDispatcher("/jsp/View.jsp").include( request, response);
   ```

7. Change classes where user data is stored. In the standard portlet API, user data is stored in a PortletPreferences object, rather than the PortletData object that is available using the IBM portlet API. Notice the different getter methods that are used with the request object and setter methods that are used for the data object.

   Change this:

   ```
   PortletData portData = request.getData();
   portData.setAttribute("userName", userName);
   portData.store();
   ```

   to this:

   ```
   PortletPreferences prefs = request.getPreferences();
   prefs.setValue("userName",request.getParameter("username"));
   prefs.store();
   ```

   Some preferences are read-only and can be modified only by an administrator. See Change configuration parameters to preferences for more information.

8. Change the method that is used for namespace encoding. For example, if the portlet uses encodeNamespace() to return a unique string to be prefixed to a JavaScript variable name within the content that is generated by the portlet, the portlet can use getNamespace().

   Change this:

   ```
   PortletResponse.encodeNamespace()
   ```

   to this:

   ```
   RenderResponse.getNamespace()
   ```

9. Change how portlet URLs are generated. For example, a portlet's doEdit() method might save the URI to the edit mode to pass to the JSP. The portlet must instantiate a PortletURL object using the createRenderURL() method.

   Change this:

   ```
   // Save URI for the edit page
   PortletURI editURI = response.createURI();
   ...
   // Preserve the edit page URI in the request to make
   // it accessible by the edit JSP
   request.setAttribute("editURI", editURI.toString());
   ```

   to this:

   ```
   // Save URI for the edit page
   PortletURL editURL = response.createRenderURL();
   ...
   ```

```
// Preserve the edit page URI in the request
// to make it accessible by the edit JSP
request.setAttribute("editURL", editURL.toString());
```

The standard portlet API does not have an equivalent method for createReturnURI(). If the URL is intended to call the portlets' action method, however, the portlet should use the createActionURL() method.

Changing JSP source

1. Change the tag library to use the standard tag library.

   Change this:

   ```
   <%@ taglib uri="/WEB-INF/tld/portlet.tld" prefix="portletAPI" %>
   ```

   to this:

   ```
   <%@ taglib uri="http://java.sun.com/portlet" prefix="portlet" %>
   ```

2. Change references to API objects. In the standard portlet API, the <portlet:defineObjects /> JSP tag makes the RenderRequest, RenderResponse, and PortletConfig objects available to JSP files. After making this change, all references in the JSP to the PortletRequest and PortletResponse should be changed to the corresponding RenderRequest and RenderResponse.

   Change this:

   ```
   <portletAPI:init />
   ...
   <%
   PortletData prefs = portletRequest.getData();
   %>
   ```

   to this:

   ```
   <portlet:defineObjects />
   ...
   <%
   PortletPreferences prefs = renderRequest.getPreferences();
   %>
   ```

3. Change JSP tags that are used for namespace encoding. For example, if the portlet uses <portletAPI:encodeNamespace/> to uniquely qualify the name of a text input field, this tag must be changed as follows.

   Change this:

   ```
   <input name="<portletAPI:encodeNamespace value='name'/>"  type="text" >
   ```

   to this:

   ```
   <input name="<portlet:namespace/>name"  type="text" >
   ```

4. Change how portlet URLs are generated. If the portlet JSP creates a URL to itself, it should specify which method gets control using the <portlet:actionURL/> or <portlet:renderURL/> tags. Any parameters passed on the URL are specified using the <portlet:param/> tag.

   Change this:

   ```
   <a href="<portletAPI:createURI>
               <portlet:URIParameter name='action' value='search'/>
            </portlet:createURI>" >
   ```

   to this:

   ```
   <a href="<portlet:actionURL>
               <portlet:param name='action' value='search'/>
            </portlet:actionURL>" >
   ```

5. Change resource bundles. The <portletAPI:text/> tag of the IBM Portlet API has been deprecated and should be replaced in all portlets by the JSTL equivalent. See Using JSTL in portlet JSPs for more information.

Change this:

```
<portletAPI:text key="my.label" bundle="nls.myproperties"/>
```

to this:

```
<fmt:setBundle basename="nls.myproperties"/>
...
<fmt:message key="my.label"/>
```

6. Change how resources are invoked from the JSP. For example, if the JSP displays an image, it should use the encodeURL() method of the appropriate response object and, in addition, add the context path of the portlet from the request.

   Change this:

   ```
   <img src='<%= portletResponse.encodeURL("images/photo01.jpg") %>'
     alt="photo">
   ```

   to this:

   ```
   <img src='<%= renderResponse.encodeURL(renderRequest.getContextPath() +
               "/images/photo01.jpg") %>' alt="photo">
   ```

Changing the portlet deployment descriptor

The following steps describe some of the differences between the portlet deployment descriptors of the IBM Portlet API and the Java Portlet Specification. However, the order of the elements in the standard portlet descriptor is important and strictly enforced during deployment. You should use a tool, such as Rational Application Developer, that performs validation as you develop the portlet deployment descriptor.

1. Remove the DOCTYPE declaration. The portlet descriptor for the standard portlets uses an XML schema, which will be added in the next step.

   Remove this:

   ```
   <!DOCTYPE portlet-app-def PUBLIC "-//IBM//DTD Portlet Application 1.1//EN"
               "portlet_1.1.dtd ">
   ```

2. Remove the <portlet-app-def/> element. The first-level element in the standard portlet descriptor is <portlet-app/>.

   Remove this:

   ```
   <portlet-app-def>
   ....
   </portlet-app-def>
   ```

3. Update the <portlet-app/> element.
   - Add the schema definition and namespace declarations.
   - Remove the major-version and minor-version attributes.
   - Set the version attribute to the required version of the Java Portlet Specification. Currently, version 1.0 is the only supported specification version.
   - Change the uid attribute to id.

   Change this:

   ```
   <portlet-app uid="com.mycompany.samples.MyPortletApp.001c"
                major-version="1" minor-version="0">
   ```

   to this:

   ```
   <portlet-app
        xmlns="http://java.sun.com/xml/ns/portlet/portlet-app_1_0.xsd"
        version="1.0"
        id="com.mycompany.samples.MyPortletApp.001c"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   ```

```
 xsi:schemaLocation=
    "http://java.sun.com/xml/ns/portlet/portlet-app_1_0.xsd
     http://java.sun.com/xml/ns/portlet/portlet-app_1_0.xsd">
```

4. Remove all <concrete-portlet-app/> elements and their contents. Save any required information, such as configuration parameters and language definitions, for use in the portlet definition.
5. Update the <portlet/> element. Remove the href, minor-version, and major-version attributes.

   Change this:

   ```
   <portlet id="com.mycompany.samples.MyPortlet.110x"
           href="WEB-INF/web.xml#com.mycompany.samples.MyPortlet.href110x"
           major-version="1" minor-version="0">
   ```

   to this:

   ```
   <portlet id="com.mycompany.samples.MyPortlet.110x">
   ```

6. Move the definition of the portlet class from the `web.xml` to the `portlet.xml`.

   Remove this from the `web.xml`:

   ```
   <servlet id="com.mycompany.samples.MyPortlet.001c">
       <servlet-name>MyPortlet</servlet-name>
       <servlet-class>com.mycompany.samples.MyPortlet</servlet-class>
   </servlet>
   <servlet-mapping
            id="ServletMapping_com.mycompany.samples.MyPortlet.001c">
       <servlet-name>MyPortlet</servlet-name>
       <url-pattern>/MyPortlet/*</url-pattern>
   </servlet-mapping>
   ```

   Add this to the portlet.xml:

   ```
       <portlet-class>com.mycompany.samples.MyPortlet</portlet-class>
   ```

7. Change how caching is defined. Use the <expires/> value from the IBM portlet descriptor as the value for the <expiration-cache/> element in the standard descriptor. There is no equivalent in the standard descriptor to the <shared/> element. See Caching portlet output for more information.

   Change this:

   ```
   <cache>
      <expires>-1</expires>
      <shared>no</shared>
   </cache>
   ```

   to this:

   ```
   <expiration-cache>-1</expiration-cache>
   ```

8. Change the content of the <supports/> element.
   - Change supported markups to MIME types.
   - Use the <portlet-mode/> element.

   The standard portlet descriptor allows you to declare only MIME types. In some cases, two markup types use the same MIME type. For example, both HTML and cHTML use 'text/html' as the MIME type. For standard portlets, WebSphere Portal Express accepts the value of a `wps.markup` initialization parameter as the markup type.

   Change this:

   ```
   <supports>
      <markup name="html">
         <view />
         <edit />
      </markup>
   </supports>
   ```

to this:

```
<init-param>
   <name>wps.markup</name>
   <value>html,chtml</value>
</init-param>
...
<supports>
   <mime-type>text/html</mime-type>
   <portlet-mode>VIEW</portlet-mode>
   <portlet-mode>EDIT</portlet-mode>
</supports>
```

Be sure to place the initialization parameters before the <expiration-cache> element.

9. Remove window state elements. Normal, maximized, and minimized window states are supported by default and not declared in the standard portlet deployment descriptor.

   Remove this:

   ```
   <allows>
      <maximized/>
      <minimized/>
   </allows>
   ```

10. Change configuration parameters to preferences. In the standard portlet descriptor, preferences can be changed by users in any of the standard modes, or they can be declared as read-only and modified only by an administrator.

    Change this:

    ```
    <config-param>
        <param-name>Location</param-name>
        <param-value>Antartica</param-value>
    </config-param>
    ```

    to this:

    ```
    <portlet-preferences>
          <preference>
              <name>Location</name>
              <value>Antartica</value>
              <read-only>true</read-only>
          </preference>
    </portlet-preferences>
    ```

11. Change localized settings.

    a. Remove the <default-locale/> element. In the standard portlet descriptor, the first locale listed in the descriptor is the default. If no locale is specified, then English is used as the default.

    b. Create resource bundles for each supported language containing the title, short title, and keywords for the portlet. Use the following parameter names:

       ```
       javax.portlet.title = My Portlet Title
       javax.portlet.short-title = Title
       javax.portlet.keywords = portlets, JSR 168, portal
       ```

    c. Declare the resource bundle in the portlet descriptor as in the following example.

       ```
       <resource-bundle>nls.MyPortlet</resource-bundle>
       ```

       In this example, the default resource bundle `MyPortlet.properties` is in the `/WEB-INF/nls` subdirectory of the WAR file and all of the locale-specific resource bundles append the locale to the file name (for example, `MyPortlet_ja.properties` for Japanese).

    d. Declare each supported locale as in the following example:

```
<supported-locale>en</supported-locale>
<supported-locale>de</supported-locale>
```

e. Set the localized values for the portlet description and display name as in the following example.

```
<description xml:lang="EN">
   English description
</description>
<display-name xml:lang="EN">
   English display name
</display>-name>
<description xml:lang="DE">
   German description
</description>
<display-name xml:lang="DE">
   German display name
</display>-name>
```

Note: The display name should be set for compatibility reasons. However, it is not currently used by WebSphere Portal Express.

Converting IBM portlets that use the Struts Portlet Framework

The existing versions of the Struts Portlet Framework supported the IBM Portal container API, or the legacy container. This release uses a newer version of the Struts Portlet Framework that supports the standard portlet container. This release will continue to ship a version to support the legacy container and a new version for the Standard container. The Struts Portlet Framework is still shipped as example war files that can be used to build the Struts application. The war files for each container can be distinguished by the name. The SPFLegacy examples support the legacy container, and the SPFStandard examples support the standard container. The SPFLegacyBlank.war is the starting point for Struts applications for the Legacy container, and the SPFStandardBlank is the starting point for the Struts applications for the Standard container.

The Struts Portlet Framework for the Legacy Container

The SPFLegacyBlank.war includes the files to be included with the Struts application. The directories of interest are the `WEB-INF/lib` and the `WEB-INF/tld` directory. Here is the list of libraries to be used in the application from the `WEB-INF/lib` directory:

1. PortalStruts.jar
2. PortalStrutsCommon.jar
3. PortalStrutsTags.jar
4. StrutsUpdateForPortal.jar
5. wp.struts-commons-logging.jar
6. commons-beanutils.jar
7. commons-collections.jar
8. commons-fileupload.jar
9. commons-lang.jar
10. commons-validator.jar
11. struts-legacy.jar
12. struts.jar

The files from the `TLD` directory are

1. struts-bean.tld
2. struts-chtml.tld

3. struts-html.tld
4. struts-logic.tld
5. struts-nested.tld
6. struts-portal-html.tld
7. struts-portal-wml.tld
8. struts-template.tld
9. struts-tiles.tld
10. struts-wml.tld

Files common to both the standard and IBM portlet containers

The following files are the Jakarta Struts 1.1 binary files, and the same in both the Standard and Legacy versions of the Struts Portlet Framework:

1. commons-beanutils.jar
2. commons-collections.jar
3. commons-fileupload.jar
4. commons-lang.jar
5. commons-validator.jar
6. struts-legacy.jar
7. struts.jar

The following files from the TLD directory are same on for both containers. This can change in future releases, so it is strongly encouraged to use the files from the blank for the required container.

1. struts-bean.tld
2. struts-chtml.tld
3. struts-html.tld
4. struts-logic.tld
5. struts-nested.tld
6. struts-portal-html.tld
7. struts-portal-wml.tld
8. struts-template.tld
9. struts-tiles.tld
10. struts-wml.tld

Conversion to the Standard Version of the Struts Portlet Framework from previous versions of the Struts Portlet Framework

Converting the legacy version of the Struts Portlet Framework to the Standard versions starts with updating the jars, and TLDs cataloged with the SPFStandardBlank.war file.

Here is a list of the files that should be updated in the `WEB-INF/lib` directory of the application:

1. wp.struts.standard.framework.jar
2. PortalStrutsCommon.jar
3. PortalStrutsTags.jar
4. StrutsUpdateForPortal.jar
5. wp.struts-commons-logging.jar
6. commons-beanutils.jar
7. commons-collections.jar

8. commons-fileupload.jar
9. commons-lang.jar
10. commons-validator.jar
11. struts-legacy.jar
12. struts.jar
13. commons-digester.jar
14. commons-logging.jar
15. jakarta-oro.jar

Here is a list of the TLD files that should be updated with the TLDs from the SPFStandardBlank.war file:

1. struts-bean.tld
2. struts-chtml.tld
3. struts-html.tld
4. struts-logic.tld
5. struts-nested.tld
6. struts-portal-html.tld
7. struts-portal-wml.tld
8. struts-template.tld
9. struts-tiles.tld
10. struts-wml.tld

The following JAR file is only required on the IBM container and must be deleted:

1. PortalStruts.jar

Web Deployment Descriptor

The Standard container requires a web deployment descriptor because the application is packaged as a war file. However, most of the initialization parameters are now configured through the portlet deployment descriptor.

1. Remove the servlet class from the web deployment descriptor. The servlet-class is no longer that way to specify the portlet class for the application in the Standard container. The portlet is now specified as the portlet class in the portlet deployment descriptor.
2. Move the init parameters from the web deployment descriptor to the portlet deployment descriptor. Since the portlet class is now defined in the portlet deployment descriptor, the init parameters are also specified in the portlet deployment descriptor. Note, the init-parameter are specified as name and value in the portlet deployment descriptor, not param-name and param-value as they are named in the web deployment descriptor.
3. The taglib elements still remain in the web deployment descriptor, no changes required.
4. The welcome file elements still remain in the web deployment descriptor, no changes required.

Portlet Deployment Descriptor

The definition for the Portlet Deployment Descriptor for the Standard container is different from the legacy container. There are some changes that are required for the converted example to deploy in the Standard

container. The information center contains details for the semantics of the portlet deployment descriptor for the Standard container.

1. The Standard container introduces the portlet-class element for specifying the class of the portlet. The portlet class for the Struts Portlet Framework is `com.ibm.portal.struts.portlet.StrutsPortlet`.
2. The init parameters for the portlet are defined in the portlet deployment descriptor. The init parameters should be converted from the web deployment descriptor.
3. The Standard container does not have the abstract and concrete separation in the portlet deployment descriptor. The portlet element defines the supported modes, and portlet preferences.
4. The Struts Portlet Framework no longer uses a portlet filter. The FilterChain init parameter should not be converted.

Struts Configuration File

The Struts Portlet Framework defines the Request Processor that must be configured in the Struts configuration file. The controller attribute processClass must be converted to the following value to be deployed on the Standard container: `<controller processorClass="com.ibm.portal.struts.portlet.WpRequestProcessor">` If the Struts application is using the Struts Request processor that supports Tiles, then the Struts plug-in must be converted as well: `<plug-in`

```
className="com.ibm.portal.struts.plugins.WpTilesPlugin">
```

Struts Action

The Struts action class is passed a HttpServletRequest object, so the application may not have a dependency on the Portal container. However, many applications use the PortletApiUtils to obtain the portlet request and interface directly with the portlet API. If so, then the application must replace the org.apache.jetspeed interfaces with the equivalent javax.portlet interfaces. The new interfaces are documented in the information center.

Note: The following example illustrates the change in which the PortletApiUtils object is obtained:

- Old: `PortletApiUtils portletUtils =`

  ```
  PortletApiUtils.getInstance();
  ```
- New: `PortletApiUtils portletUtils =`

  ```
  PortletApiUtils.getUtilsInstance();
  ```

StrutsPortlet

The `com.ibm.wps.portlets.struts.WpsStrutsPortlet` class for the legacy container extended the PortletAdapter class. The Struts application using the Struts Portlet Framework may have been customized by extending the WpsStrutsPortlet class. If so, those changes should be applied for the Standard container. The `com.ibm.portal.struts.portlet.StrutsPortlet` class for the Standard container extends the standard container's GenericPortlet.

Request processor

The `com.ibm.wps.portlets.struts.WpsRequestProcessor` class for the legacy container may have been extended to customize the processing. The

Request Processor class for the standard container is `com.ibm.portal.struts.portlet.WpRequestProcessor`. If the legacy interfaces were used for the customizations, these changes should be converted to the Standard interfaces.

"Converting portlet instances and settings from the IBM API to the standard API"
The portal provides a portlet conversion task that allows you to convert the settings and instances of IBM API portlets to the corresponding standard API portlets. This is useful when you intend to replace IBM API portlets by standard API portlets.

Related reference:

"Standard portlet API" on page 2934
The Java Portlet Specification addresses the requirements of aggregation, personalization, presentation, and security for portlets running in a portal environment.

Converting portlet instances and settings from the IBM API to the standard API:

The portal provides a portlet conversion task that allows you to convert the settings and instances of IBM API portlets to the corresponding standard API portlets. This is useful when you intend to replace IBM API portlets by standard API portlets.

About this task

You can use this task as follows: The portlet conversion task converts portlet settings of the IBM API portlet to portlet preferences of the standard API portlet. It also converts instances of the IBM API portlet to instances of the standard API portlet. User customized portlet data that is associated with the portlet instance is converted into standard API portlet preferences.

To convert the instances and settings of an IBM API portlet to a standard API portlet, proceed by the following steps:

Procedure

1. Install the standard API portlet by which you want to replace the IBM API portlet.
2. Create a portlet conversion properties file that identifies both the IBM API portlet and the standard API portlet. To do this, create or update the following file:

   AppServer_root`/ConfigEngine/properties/portletconversion.properties`

   Confirm that the following parameters are set as specified or modify them if necessary:

   For the IBM API portlet:

   ibmwebapp.uid
   The uid of the Web application that contains the IBM API portlet. This property is required.

   Parameters for identifying the IBM API portlet:
   To identify the IBM API portlet, specify one of the following three parameters: the portlet name, the object ID, or the unique name of the IBM API portlet:

ibmportlet.portletname
The portlet name of the IBM API portlet.

ibmportlet.uniquename
The unique name of the IBM API portlet.

ibmportlet.objectid
The object ID of the IBM API portlet.

For the standard portlet:

jsrwebapp.uid
The uid of the web application that contains the standard API portlet. This property is required.

Parameters for identifying the standard API portlet:
To identify the standard API portlet, specify one of the following three parameters: the portlet name, the object ID, or the unique name of the standard API portlet:

jsrportlet.portletname
The portlet name of the standard API portlet.

jsrportlet.uniquename
The unique name of the standard API portlet.

jsrportlet.objectid
The object ID of the standard API portlet.

Additional parameters:

pages.uniquename
This parameter is optional. Specify a comma separated list of unique names of pages. If you specify this parameter, only portlets on these pages and their descendants are converted. If you leave this parameter empty or missing, instances of IBM API portlets on all pages are converted.

converter
The name of a converter class that is invoked by the `porltetconversion` task and that performs the conversion of portlet settings and portlet data. The converter class must implement the interface `com.ibm.portal.portletconversion.Converter`. You can specify the default converter `com.ibm.wps.pe.task.DefaultConverter` here. This converter performs basic conversion by filtering out portlet data items whose type is not `String`.

converter.classpath
Semicolon separated list of files and folders that are added to the classpath in order to load the converter class.

xmlaccess.url
The URL to the portal XML configuration interface servlet. You can use this parameter to run conversions for specific virtual portals. If this parameter is empty or missing, the default portal is used to run the conversion.

3. Change to the directory *AppServer_root*`/PortalServer/ConfigEngine/`.
4. Run the portlet conversion task `ConfigEngine convert-portlets` .
5. Verify the conversion by reviewing the console. The message `Build successful` indicates a successful conversion. If the message `Build failed` is displayed upon completion of the task, review the previous steps.

6. After successful conversion you can uninstall the IBM API portlet.

Example

Examples of `portletconversion.properties` files:

```
ibmportlet.objectid=3_O4C9FI930GPE90IGU02QAR0006
jsrportlet.objectid=3_O4C9FI930GPE90IGU02QAR00G3
ibmwebapp.uid=DCE:472fb1b0-3d22-1211-0000-005da8cf7ayz:2
jsrwebapp.uid=StdPortletDataTestPortlet.war.webmod
converter=com.ibm.wps.pe.task.DefaultConverter
```

```
ibmportlet.portletname=An PortletData test portlet
jsrportlet.portletname=StdPortletDataTestPortlet
ibmwebapp.uid=DCE:472fb1b0-3d22-1211-0000-005da8cf7ayz:2
jsrwebapp.uid=StdPortletDataTestPortlet.war.webmod
converter=com.ibm.wps.pe.task.DefaultConverter
```

Portlet development reference

View important information and concepts related to portlet development.

Related information

- "Standard portlet API" on page 2934
- "IBM Portlet API" on page 3110

"Markup guidelines"
View the guidelines for using HTML, WML, and cHTML markup in your portlet JSPs to provide a consistent, clean, and complete user interface.

"Building `.ear` and `.war` files" on page 3127
You can run a **`configengine`** task to build an `.ear` file or a `.war` file from an expanded directory.

"Accessing the portlet session on the anonymous page" on page 3128
View some pointers on handling portlet sessions in situations where portlets are placed on pages that do not require authentication.

"Deployment descriptors" on page 3129
The deployment descriptors define configuration information for the portlets that a portlet application contains.

"JSP tags for standard portlets" on page 3133
The standard portlet API defines several tags that can be used in portlet JSPs to access the portlet request and response and to generate URLs.

"Handling and visibility of request parameters in portlets" on page 3135
Learn how the standard portlets and IBM portlets set request parameters for requests targeted to portlets. Acquire an understanding of the parameter visibility for standard portlets, IBM portlets, and the included JSPs or servlets.

"Detailed descriptions of the Struts WML tags" on page 3138
Learn about the WML tags used by the portlets within the Struts Application Framework.

"Application extension registry" on page 3148
WebSphere Portal Express provides an application extension registry which is equivalent to the application extension registry provided by IBM WebSphere Application Server. This registry allows any J2EE compliant application to define extension points for other applications to use, or to plug in to other extensible applications.

Markup guidelines

View the guidelines for using HTML, WML, and cHTML markup in your portlet JSPs to provide a consistent, clean, and complete user interface.

The portal server page is displayed with *skins* and *themes* that are defined by the portal designer or administrator. For portlets to integrate with an organization's portal or user's customized portal, they must generate markup that starts the generic style classes for portlets. They must not use tags or attributes to specify colors, fonts, or other visual elements.

Portlets are allowed to render only markup fragments, which are then assembled by the portlet framework for a page. Portlet output must contain complete, well-structured, and valid markup fragments. This output helps to prevent the portlet's HTML code, for example, from corrupting the portal's aggregated HTML code. Use a validation tool for your markup, such as the W3C HTML Validation Service or a tool from a markup editor.

These guidelines are based on the JSP coding guidelines for standard portlets. For more information, read the corresponding guide for the type of portlet you are developing on the WebSphere Portal Zone.

HTML

- Use standard HTML. For the official HTML specification, see W3C.
- Use only HTML fragments. Because portlets contribute to the content of a larger page, they must provide HTML fragments and must not have `<html>`, `<head>`, or `<body>` tags.
- Use only elements that can be rendered properly in an HTML table cell (`<td>...</td>`). Frames, for example, do not open when inserted in a table.
- Make sure that the fragments are well-formed to prevent unbalanced pages. Watch out for unterminated, extraneous, or improperly nested tags. The behavior of pages with improperly stopped tags is unpredictable.
- Avoid lengthy, complex HTML. Portlets share a page with others and the more content each portlet generates, the more the browser must process before it can open anything.
- Use portlet classes in the portal server's stylesheet, `Styles.css`. If you hardcode the fonts and colors, the portlet's appearance looks out of place when the user changes the page style settings. Portal administrators and users can affect the appearance of the portal by changing the portal theme and the portlet skins. Portlets can pick up on style changes by using styles that are defined in the portal theme's cascading stylesheet, `Styles.css`. For example, instead of decorating an `<input>` element with the **`style`** attribute, refer to the theme's input class definition: `<inputclass="portlet-form-button" type="submit">`

  Portlet style classes are marked with the following comment:

  ```
  /* Styles used in portlets
  ```

  When available, use the WSRP style classes, which have a `portlet-` prefix. See *CSS Style Definitions* in the WSRP 1.0 specification.

  The relevant document says, *"One of the goals of an aggregated page is a common look-and-feel across the Portlets contained on that page. This not only affects the decorations around the Portlets, but also their 25 content. Using a common CSS stylesheet for all Portlets, and defining a set of standard styles, provides this common look-and-feel without requiring the Portlets to generate Consumer-specific markup. Portlets SHOULD use the CSS style definitions from this specification in order to participate in a uniform display of their content by various Consumers. For markup types that support CSS stylesheets, Consumers MUST supply a CSS stylesheet to the End-User's agent 30 with definitions for the classes defined in section 10.6 of this specification. This section defines styles for various logical units in the markup."*

- Do not use CSS for absolute positioning. It can defeat the portal features that allow users and administrators to place content on the page.
- Limit the size of the portlet output. Portlets contribute a portion of a larger page. The size of the JSPs (in terms of horizontal and vertical span) can determine how easily the portlet can fit on a page with multiple columns and other portlets. A large portlet forces other portlets off the screen and creates large scrolling regions, resulting in usability issues. When you design a portlet's JSPs, avoid unnecessary layout elements. Focus the portlet's view on the pertinent information and consider whether the portlet is intended to be placed on pages with other portlets. In particular, take notice of image size, pre-formatted text (<pre/> tag), and absolute widths on elements, such as tables.
- Use Java style comments instead of HTML style. HTML comments remain in the rendered content, adding to the document size and the amount of data that passes to the client. If you use Java style comments within your JSPs, instead, they are removed from the rendered source, along with the rest of the Java code. You must embed these comments within scriptlets:

  ```
  <% // this is a comment %>
  ```
- Make pages fully accessible. To allow portal users with disabilities to be able to use your portlet, the JSPs must be fully enabled for keyboard-only control and other assistive technologies. Follow these general guidelines:
  - Use the **alt** attribute with images to define descriptive text of the image content.
  - Use <label> tags to associate labels with form input controls so that page readers are able to associate prompts with inputs.
  - Do not use color alone to denote state or information. For example, red text to emphasize information does not help people who are color blind. Use bold or italics instead, or use color with graphic changes.
  - Do not use voice or other sounds to convey information.

  For more information, see IBM Accessibility Center.
- URIs, HTML element name attributes, and JavaScript resources must be namespace encoded. Use <portlet:namespace/> (standard portlet API) for named elements to prevent name conflicts with other portlets on the portal page.
- Test the portlet output in different browsers. Check portlet behavior when you resize the page to ensure that your portlet works with different browser sizes. Check portlet behavior when the default browser font is changed; your portlet should handle these situations seamlessly.
- Do not draw portlet banners, such as title bars, since they are provided by the portal aggregation.
- Minimize dependencies on JavaScript. Because JavaScript implementations and behavior differ widely among browser types and versions, the more your portlet depends on JavaScript, the more browser-dependent your portlet becomes. Additionally, the more of the page that is rendered using JavaScript, the more difficult it is to maintain and to extend to markups other than HTML. Also, it is more difficult to namespace encode JavaScript resources and nearly impossible to properly encode (response.encodeUrl()) URLs built with JavaScript.
- Do not use pop-up windows. Interactions within the portal are state-based, which means that the portal tracks your trail through the pages and portlets. Using the browser's back button or creating pop-up windows in browser instances that contain portlets can cause the portal to lose track of your current state and cause problems. Other than using JavaScript prompts, there is no safe way to create pop-up windows within the portal, unless the new link takes you to an external page, outside the portal. The alternative is to link to an external

page within a new browser window (with the TARGET attribute on the anchor). The user remains within the portal in the original browser window.

- Use IFRAMEs with caution. IFRAMEs are an easy way to include external content within a portlet, but undermine the whole portlet principle because the portlet API is tunneled or side-stepped. Therefore, IFRAMEs can be used for special cases, such as surfacing existing applications. Consider the following potential issues when you use an IFRAME:
  - The IFRAME fills its content based on a URL. The URL must be addressable by the browser, therefore the server that is the target of the URL must be accessible by the browser.
  - Not all browser levels support IFRAMEs.
  - If the content is larger than the IFRAME region, then enable horizontal and vertical scrolling to allow the user scroll through the embedded content. Content that contains scrolling regions itself can make it difficult for the user to manipulate all scrolling regions to view all embedded content, causing usability problems.
- Use JSTL instead of inventing common tags. The Java Standard Tag Library (JSTL) defines many commonly needed tags for conditions, iterations, URLs, globalization, and formatting. For more information, go to Using JSTL in portlet JSPs.

WML

- Use standard WML. For information, refer to Open Mobile Alliance.
- When you design your portlet, assume that it can be opened alone or with other portlets.
- Watch out for unterminated and extraneous tags; the behavior of pages with improperly ended tags is undefined.
- Use only elements that can be included into a card (probably together with other portlets and elements) and do not create separate cards.
- Do not set the card title.
- Do not use templates.
- Avoid lengthy, complex WML since the buffer length can be restricted by the user agent, and portlets can share a page with others.
- Avoid having too many images. When you use images, define a meaningful **`alt`** name, since it is necessary for the devices that do not support images, or if the image cannot be fetched. If possible, define **`localsrc`**.
- Do not use intrinsic events since these events (such as oneventforward and oneventbackward) can be added only by the aggregation level.
- Avoid timers.
- Use user-triggered events only if necessary, and always define label and name parameters explicitly. Prefix the name with the portlet's unique identifier, otherwise conflict is possible between different portlets on one page. Use meaningful names for labels.
- Do not create fixed-width WML tables or images in portlets.
- Avoid long, unbroken lines of unwrapped text to help prevent unnecessary scrolling.

Compact HTML (cHTML)

- Use standard cHTML. For information on i-mode compatible HTML, refer to Compact HTML for Small Information Appliances.

- Assume when you design your portlet that it can be opened alone or with other portlets.
- Watch out for unterminated, extraneous, or improperly nested tags. The behavior of pages with improperly ended tags is unpredictable.
- Use only elements that can be included into a body; do not use <head> or <body> tags.
- Avoid lengthy, complex cHTML since portlets can share a page with others.
- Avoid having too many images and do not use alignment and positioning parameters with images.
- Do not exceed a length of 24 symbols for the phone numbers that are used as references in the <a> tag.
- Do not use URLs that are longer than 200 bytes.

Building .ear and .war files

You can run a **configengine** task to build an .ear file or a .war file from an expanded directory.

About this task

Run the following tasks to build .ear and .war files:

Procedure

1. Run the following task to build a .war file:
   - Linux: `./ConfigEngine.sh build-war-file -Dsource.war.directory=directory_path.war -Doutput.war=directory_path.war`
   - Windows: `ConfigEngine.bat build-war-file -Dsource.war.directory=directory_path.war -Doutput.war=directory_path.war`
   - IBM i: `ConfigEngine.sh build-war-file -Dsource.war.directory=directory_path.war -Doutput.war=directory_path.war`

   where the following parameters are defined as:
   - **source.war.directory**: The location of the expanded or customized war file.
   - **output.war**: The location of the generated war file.

   The following information is an example of the **build-war-file** task:

```
Extract the bannerad.war file to a directory
 cd /tmp/files
 jar -xvf /opt/IBM/WebSphere/PortalServer/bp/wp.bp.bannerad/installableApps/bannerad.war

Make an update to a file inside /tmp/files/WEB-INF

Build a new war from the expanded directory
 ./ConfigEngine.sh build-war-file -Dsource.war.directory=/tmp/files/ -Doutput.war=/tmp/wars/bannerad.war

You can now install the updated portlet into WebSphere_Portal
 /tmp/wars/bannerad.war
```

2. Run the following task to build an .ear file:
   - Linux: `./ConfigEngine.sh build-ear-file -Dsource.war.directory=directory_path.war -Doutput.ear=directory_path.ear -Dapp.name=app_name -Dwar.name=war_name.war -Ddisp.name="display_name"`
   - Windows: `ConfigEngine.bat build-ear-file -Dsource.war.directory=directory_path.war -Doutput.ear=directory_path.ear -Dapp.name=app_name -Dwar.name=war_name.war -Ddisp.name="display_name"`

- IBM i: `ConfigEngine.sh build-ear-file -Dsource.war.directory=`*`directory_path`*`.war -Doutput.ear=`*`directory_path`*`.ear -Dapp.name=`*`app_name`* `-Dwar.name=`*`war_name`*`.war -Ddisp.name="`*`display_name`*`"`

Where the following parameters are defined as:

- **source.war.directory**: The location of the expanded or customized war file.
- **output.ear**: The location of the generated ear file.
- **app.name**: The application name substituted in the `application.xml` template.
- **war.name**: The war name substituted in the `application.xml` and `ibm-application-runtime.props` files.
- **disp.name**: The display name substituted in the `application.xml` template.

The following information is an example of the **build-ear-file** task:

```
Make a local copy of the expanded PA_ThemesAndSkinsMgr.ear file (which contains an expanded
ThemesAndSkinsMgr.war file inside of it)
 cd /tmp/files
 cp -R /opt/IBM/WebSphere/wp_profile/installedApps/Cell_name/PA_ThemesAndSkinsMgr.ear/* /tmp/files

Validate the parameters to use by looking at the application.xml file
 /tmp/files/META-INF/application.xml

Build a new ear from the expanded directory
 ./ConfigEngine.sh build-ear-file -Dsource.war.directory="/tmp/files/ThemesAndSkinsMgr.war"
  -Doutput.ear="/tmp/ears/PA_ThemesAndSkinsMgr.ear" -Dapp.name="PA_ThemesAndSkinsMgr"
  -Dwar.name="ThemesAndSkinsManager.war" -Ddisp.name="ThemesAndSkinsMgr_war"

You can now install an enterprise application into the WebSphere Integrated Solutions Console.
 /tmp/ears/PA_ThemesAndSkinsMgr.ear
```

Accessing the portlet session on the anonymous page

View some pointers on handling portlet sessions in situations where portlets are placed on pages that do not require authentication.

Note: Authenticated and remembered users must have cookies enabled on their browser. Users can access portal sites without cookies enabled if they are anonymous users. If you turn on session tracking for anonymous users, then anonymous users also require cookies.

Administrators can place your portlet on a page that is presented to anonymous users (similar to the Welcome page provided by WebSphere Portal). By default, when a portlet is placed on a page that does not require authentication and no user is logged in, the portal server does track sessions across subsequent request to the server. Portlets should not create a session using the `request.getSession(true)` call in this case; which results in WebSphere Application Server warning messages similar to:

```
SESN0066E: Response is already commited to client. Session cookie cannot be set.
```

In this case, a temporary session is created and your session information will be lost in the next request. If you need to enable session tracking across requests for non-authenticated users, you can do so by setting the `public.session` parameter in the portal Navigator service configuration or by setting the `com.ibm.portal.public.session` container run time option in a JSR 286 portlet deployment descriptor. Note that this may result in significantly increased memory consumption. For details about the `com.ibm.portal.public.session` option and a code sample refer to the topic about Deployment descriptors, section about Container run time options. Instead of using these options, portlets that need to maintain interaction state even for non-authenticated users should use render parameters to keep this information instead of the portlet session, as recommended by the Java Portlet Specification.

The portlet may need to present the user with an appropriate message if it requires a valid portlet session to operate correctly. For example:

```
This content cannot be displayed until you log in.
Please report this problem to the site administrator.
```

In addition, the administrator will need more helpful information that the portlet can provide in the portlet log:

```
Unable to locate the portlet session.
This portlet requires a session to function.
Move the portlet to an authenticated page or
turn on session tracking for anonymous users.
```

If the portlet does not require a session for critical operation, then perhaps any subfunctions within the portlet require the session can be suppressed to anonymous users. This should be evaluated for each individual portlet.

If `request.getPortletSession()` or `request.getPortletSession(true)` are called when the user is not logged in and WebSphere Portal is not configured to use a session for anonymous users, each request from each client creates an extraneous `PortletSession` object that is lost and consumes JVM memory. This causes more frequent JVM garbage collection and hurts overall WebSphere Portal performance.

In order for a portlet to function without a portlet session, you must add this line to the beginning of all portlet JSPs.

```
<%@ page session="false" %>
```

Without this directive, the JSP page compiler generates code that accesses the session even if you don't use it in your JSP. You also cannot access any beans with scope set to session in any of your JSP pages as shown.

```
<jsp:useBean ... scope="session" />
```

This will create sessions when you do not want them. Instead, determine whether the session exists, for example:

```
<%
   com.ibm.MyClassName theBeanId = null;
   PortletSession session = request.getPortletSession(false);
   if (session != null) {
      theBeanId = (com.ibm.MyClassName)session.getAttribute("theBeanId");
   }

   // later in your code always check to see if the bean exists before using it
   if (theBeanId != null) {
      // use the bean
   }
%>
```

Deployment descriptors

The deployment descriptors define configuration information for the portlets that a portlet application contains.

Each portlet application requires two deployment descriptors:

- The *portlet deployment descriptor* provides the portal server with information about the portlet resources in the application, including configuration, support characteristics, and titles. The portal server uses this information to provide services for the portlet. For example, if a portlet registers its support for help

and edit mode in the portlet deployment descriptor, the portal server renders icons to allow the user to start the portlet's help and edit pages. This file is always named `portlet.xml`.

- The *web application descriptor* provides the application server with information about the web resources in the application. This file is always named `web.xml`. The format of this file is described in the Deployment Descriptor chapter of the *Servlet Specification*.

The following topics describe the deployment descriptors of a portlet application:

- Web application deployment descriptor
- Portlet deployment descriptor
- Reserved parameter names

Web application deployment descriptor

This section describes the required elements of the `web.xml` for standard portlets. For more information about the web application deployment descriptor, see the Java Servlet Specification Version 2.3.

Web application descriptor for standard portlets

As described in the Java Portlet Specification, a portlet application is an extended web application and therefore has a servlet context. The portlet context uses most of its function from the servlet context of the portlet application. Since the `web.xml` is a mandatory item in a Java Platform, Enterprise Edition web ARchive, you must verify that your portlet application includes a `web.xml` in its WAR file that conforms to Version 2.3 or a later version of the Java servlet Specification. This ensures that the portlet application is compatible with other portal server implementations and WebSphere Portal Express. The `web.xml` must contain, at a minimum, the `<web-app/>` and `<display-name/>` elements. You can also include context-wide parameters using the `<init-param>` element. Refer to the *Java Servlet Specification Version* for more information about the structure of the web application deployment descriptor.

IBM web application descriptor

As with other servlets that follow the Java Platform, Enterprise Edition model, portlets are packaged as WAR or EAR files with a web application deployment descriptor, `web.xml`. This descriptor defines each portlet as a servlet within the web application, including unique identifiers for each portlet, the portlet class, and initialization parameters.

The definition of the servlets in the `web.xml` must be in the same order as the definition of portlets in the `portlet.xml`. The servlet identifier must be referenced by the portlet deployment descriptor, using the **`href`** attribute of the <portlet> tag. As shown in the following table, the **`href`** attribute indicates the path of the web application descriptor in the WAR file that is appended by the servlet ID as the anchor.

Table 462. The href attribute in the portlet.xml and web.xml files

portlet.xml	web.xml
`<portlet id="Portlet_1"` `href="WEB-INF/web.xml#Servlet_1">` `<portlet-name>Mail</portlet-name>` `...` `</portlet>`	`<servlet` `id="Servlet_1">` `<servlet-name>MailPortlet</servlet-name>` `...` `</servlet>`

Table 462. The href attribute in the portlet.xml and web.xml files (continued)

portlet.xml	web.xml
<portlet id="Portlet_2" href="WEB-INF/web.xml#Servlet_2"> <portlet-name>Calendar</portlet-name> ... </portlet>	<servlet id="Servlet_2"> <servlet-name>CalendarPortlet</servlet-name> ... </servlet>

Portlet deployment descriptor

This section describes the `portlet.xml` for standard portlets.

- The elements and structure of the IBM `portlet.xml` is provided in this topic.
- For information about the portlet deployment descriptor for the standard portlet API, see the Java Portlet Specification.

Guidelines for portlet application identifiers

The identifiers of portlet applications must identify them unambiguously in the area of their usage, which can be worldwide. The same is true for concrete portlet applications for portlets. To make this possible, follow these guidelines.

- Include the portlet's namespace in the identifier, using the same format that is used for the Java packages
- Add some portlet application-specific description
- Add some arbitrary characters to ensure uniqueness within the namespace, for example:

  ```
  com.ibm.wps.samplet.mail.4969
  ```

- For IBM portlets, add suffixes for the corresponding concrete portlet applications, for example:

  ```
  com.ibm.wps.samplet.mail.4969.1
  ```

Portlet identifiers must be unique within the application.

Standard portlet deployment descriptor

In the Java Portlet Specification, the portlet descriptor is in the format of an XML schema. Since the portlet is not a servlet, the portlet descriptor does not reference a corresponding servlet ID in the `web.xml`. The first-level element is the `<portlet-app/>`, which contains one or more `<portlet/>` definitions. Portlet-specific initialization parameters are stored in the `portlet.xml` by using the `<init-param/>` element and are obtained from the `PortletConfig` object that is provided during initialization.

IBM portlet deployment descriptor

The structure of the IBM portlet deployment descriptor is defined by a DTD, which is located at *wps.war*`/dtd/portlet_1.1.dtd`.

For IBM portlets, each concrete portlet definition indicates its parent portlet using the **href** attribute of the <concrete-portlet> tag. As shown in the following table, the **href** attribute indicates the portlet ID as an anchor.

Table 463. The href attribute in the portlet tag and the concrete portlet tag

Portlet tag	Concrete portlet tag
<portlet id="Portlet_1" href="WEB-INF/web.xml#Servlet_1"> <portlet-name>Mail</portlet-name> ... </portlet>	<concrete-portlet href="#Portlet_1"> <portlet-name>Mail Box</portlet-name> ... </concrete-portlet>
<portlet id="Portlet_2" href="WEB-INF/web.xml#Servlet_2"> <portlet-name>Calendar</portlet-name> ... </portlet>	<concrete-portlet href="#Portlet_2"> <portlet-name>Group calendar</portlet-name> ... </concrete-portlet>

Reserved parameter names

This section describes parameters that can be set for portlets. There are two types of parameters:

Configuration parameters
These are parameters for a portlet that can be changed only by an administrator. They are defined in the file `portlet.xml` using either the **<config-param/>** elements (for IBM portlets) or the **<preferences/>** element marked as read-only (for standard portlets).

Initialization parameters
These parameters are set during deployment. For standard portlets, they are set by the <init-param/> element in the `portlet.xml`. For IBM portlets, they are set by the <init-param/> element in the `web.xml`.

The following parameters can be set as configuration parameters in IBM portlets or initialization parameters in standard portlets:

`redirect.action.without.session = (false)`
Set this parameter to `true` to enforce a redirect after an action on anonymous pages for this portlet.

`wps.enforce.redirect = (false)`
Set this parameter to `true` if your portlet requires a redirect for a portlet action. The advantage of sending a redirect is that the URL changes to hold the complete state in the URL such that the resulting URL is present in the browser and can be bookmarked. The downside is that the additional request generates more load on the server. The portlet does not need the redirect after action if it does not set any public or private render parameters in processAction or processEvent.

`wps.markup`
This is an initialization parameter in standard portlets to define the supported markup types for the portlet. Use it to differentiate between markup types, such as HTML and cHTML, that use the same MIME type.

`com.ibm.portal.automaximize`
Set this parameter to true to automatically maximize the portlet when the user switches from view mode to any of the other supported modes. The default is false.

`wps.multiple.action.execution`
Set this parameter to `true` to enable the user to run the same action URL several times.

The following parameter can be set as a configuration parameter in IBM portlets or read-only preferences in standard portlets:

com.ibm.portal.pagecontext.enable = (false)
When you set this parameter to `true`, you indicate that the portlet can retrieve task properties. This parameter is used by task processing portlets. The default is false.

Container run time options

The portal defines container run time options, which can be used by portlets that conform to the Java Portlet Specification 2.0 (JSR286). You set these parameters by using the `<container-runtime-option>` element in the `portlet.xml`.

com.ibm.portal.public.session
Set this parameter to `true` to allow session creation for anonymous users for this particular portlet. Example:

```
<container-runtime-option>
   <name>com.ibm.portal.public.session</name>
   <value>true</value>
</container-runtime-option>
```

Related reference:

"Standard portlet API" on page 2934
The Java Portlet Specification addresses the requirements of aggregation, personalization, presentation, and security for portlets running in a portal environment.

JSP tags for standard portlets

The standard portlet API defines several tags that can be used in portlet JSPs to access the portlet request and response and to generate URLs.

To make these tags available in a JSP, the following directive is required at the beginning of the JSP:

```
<%@ taglib uri="http://java.sun.com/portlet" prefix="portlet"%>
```

For more information, see Java Portlet Specification.

WebSphere Portal Express provides an extra tag for use in standard portlets. To make these tags available in a JSP, the following directive is required:

```
<%@ taglib
    uri="http://www.ibm.com/xmlns/prod/websphere/portal/v8/portlet/ibm-portlet-ext"
    prefix="portlet-ext" %>
```

The following is a brief description of the extension to the JSR 168 portlet JSP tag library.

<portlet-ext:portalRenderURL ***attribute*****="*****value*****">**
Creates a URL to pages or portlets on pages. Attributes are as follows:

contentNode="***id*** **|** ***name*****"**
Indicates the ID or unique name of the page. The name or ID of the content node is also used to specify the page where the portlet can be found.

portletWindow="***id*** **|** ***name*****"**
Indicates the ID or unique name of the control that holds the portlet. Must be used in combination with the attribute **`contentNode`** to identify the page where the portlet is located.

portletMode="view | help | edit | configure"
For URLs to a portlet indicated by **portletWindow**, this attribute sets the portlet mode. This parameter is ignored if the attribute **portletWindow** is not set.

windowState="maximized | minimized | normal"
In a portlet, this attribute indicates the state of the portlet window when it is displayed. If the portlet state is not specified, the page is shown with the previous state of the portlet. This parameter is ignored if the attribute **portletWindow** is not set.

portalState="solo | normal"
Indicates whether the specified portlet window is rendered normally or solo, that is without a theme. This parameter is ignored if the attribute **portletWindow** is not set.

newWindow="true | false"
This attribute creates a session partition. For portlet URLs, use this attribute if you want to display the portlet either in a new window or in an iFrame. The default value is false. The portlet window state for the addressed portlet in the new window is set to maximized. The portlet mode is set to the value of the current parent window.

locale="*locale*"
Specifies the locale with which subsequent portal page requests are rendered. This attribute is optional.

var ="*name*"
Specifies the name of a scripting variable that is exposed in the body of the tag. This attribute is optional. The variable exposes an object that implements the interface `com.ibm.portal.DisposableURL` that can be used to stream the URL to the output.

<portlet-ext:urlParam name="*parameter_name*" value="*parameter_value*">
Use this tag to add custom parameters of your choice to the parent **portalRenderURL**. Parameters are added to the **portalRenderURL** as render parameters of the specified portlet window. Parameters are ignored if the enclosing **portalRenderURL** does not specify the attribute **portletWindow**. Specify attributes as follows:

name Required. Indicates the name of the parameter.

value Required. Indicates the value of the parameter.

<portlet-ext:bidi dir="rtl | ltr" />
This tag is used to support bidirectional languages. Bidirectional languages contain text that reads in both directions. For example, URLs, code samples, or directory and file names can be read in the opposite direction of the rest of the text.

dir Indicates the normal direction of text in the language.

- For **dir="rtl"**, the tag content is written only if the client's locale belongs to a bidirectional language. This is the default setting if **dir** is not specified.
- For **dir="ltr"**, the tag content is written only if the client's locale does not belong to a bidirectional language.

locale The tag content is written only if the language is not bidirectional.

<portlet-ext:setBundle basename="*value*" var="*value*" scope="*value*" bundle="*value*" provider="*value*"/>

Use this tag to compute the locale that is used for the JSTL format tags that are based on the portal specific locale computation algorithms. To make sure that the locale used by JSTL matches the locale that is used by other dynamic elements on the portal page, prefer this tag over the JSTL tag `<fmt:setBundle>`. This creates a globalization context and stores it in the scoped variable or the `javax.servlet.jsp.jstl.fmt.localizationContext` configuration variable.

basename
The resource bundle base name. This is the fully qualified resource name of the bundle. It has the same form as a fully qualified class name, that is, it uses a period full stop (.) as the package component separator. It does not have a file type suffix, such as `.class` or `.properties`.

var The name of the exported scoped variable that stores the globalization context of type `javax.servlet.jsp.jstl.fmt.LocalizationContext`.

scope The scope of `var` or the globalization context configuration variable.

bundle
The instance of the `java.util.ResourceBundle` to use.

provider
The instance of the `com.ibm.portal.model.ResourceBundleProvider` to use.

The use of `basename`, `bundle`, and `provider` are mutually exclusive.

You can also use JSTL tags as described in *Generating output*.

Related concepts:

Generating output
View how portlets use JSPs to generate markup, create URLs to portlet resources, support multiple devices, markups, and languages in portlets, and make use of JSTL.

"Advanced URL generation for data exchange" on page 3072
For data exchange, IBM WebSphere Portal Express supports cross-portlet links

Related reference:

"Standard portlet API" on page 2934
The Java Portlet Specification addresses the requirements of aggregation, personalization, presentation, and security for portlets running in a portal environment.

Handling and visibility of request parameters in portlets

Learn how the standard portlets and IBM portlets set request parameters for requests targeted to portlets. Acquire an understanding of the parameter visibility for standard portlets, IBM portlets, and the included JSPs or servlets.

Both IBM and standard portlets have different ways of setting request parameters for requests targeted to portlets. This topic is best understood if you have some familiarity with passing parameters using either of these APIs for portlet programming.

1. If the portlets generate URLs using the portlet API, they can set parameters programmatically using the PortletURL.setParameter (standard) or PortletURI.addParameter (IBM) calls.
2. When HTML forms are submitted to the portal, the form values are passed as request parameters. If you use the GET method for form submission in IBM portlets, other parameters that have been programmatically set on the form action URL may not be passed back by the client. For JSR 168 and 286 HTTP GET parameters are transmitted as render parameters.
3. Portlets can trigger operations on other portlets using the Property Broker APIs. In this case, the target portlet can receive parameters that propagate values from the source portlet.
4. Portal theme and skin JSPs can create URLs to portlets with the `<portal-navigation:urlGeneration/>` tag. These portlet URLs can also include parameters that are visible to all IBM portlets (not to standard portlets).

Multiple portlets are normally called within the processing of a single client request, and these different methods of setting parameters can interact in various ways. Therefore you need to consider which parameters are actually visible to a portlet, especially if standard and IBM portlets are placed on the same page and are invoked within a single client request.

Parameter visibility for standard portlets

Standard portlets do not use parameter namespacing. Parameter namespacing is not necessary In an environment consisting of only standard portlets because these portlets receive the parameters from the client request only if they are directly targeted by a URL, so it is implicitly clear to which portlets the parameters belong.

Standard portlets are the target of a URL if the URL has been created with the RenderResponse.createRenderURL() or RenderResponse.createActionURL() methods. In this case, the portlet will receive the parameters that have been programmatically set on the URL. In the case of an action URL, the portlet can read the query or form parameters for the request, so that it can read the form input if the URL was the action of a HTML form. As stated by the Java Portlet Specification, portlets should not use render URLs for HTML form actions because form parameters may not be visible in this case.

For portlets that are not the target of a URL, the Java Portlet Specification defines the notion of render parameters. In any render call that is not triggered by a render URL targeted to the portlet (that is, when the portlet is simply shown as part of a page where the user interacts with another portlet), the portlet sees only its active set of render parameters, independent of any parameters that are set on the client request. The portlet cannot see request parameters set by the `<portal-navigation:urlGeneration/>` tag unless the tag explicitly specifies a target portlet by using the layoutNode attribute.

New render parameters are set either by a render URL or by a portlet action and the portal will keep providing those render parameters to the portlet as long as the portlet is not the explicit target of a URL or a property broker event. That means, the render parameters can (and should) be used to store the navigational state of the portlet.

Finally, an action of a standard portlet can be invoked through the property broker if property values are propagated from another portlet. In this case, the portlet will see its current set of render parameters, so that it is able to keep (parts of) its

current state across the action call. In addition, those parameters that are defined by the property passing mechanism of property broker will be visible, and they will override render parameter values with the same name. Again, the visible parameters are independent of any parameters set on the client request.

Note that the render parameters are only passed to the portlet action, if the action is invoked by property broker. Actions that are invoked by a URL will not automatically see them. You can, however, pass all render parameters by setting them explicitly on the URL:

```
PortletURL url = renderResponse.createActionURL;
url.setParameters(renderRequest.getParameterMap());
```

Parameter visibility for IBM portlets

IBM portlets normally use parameter namespacing, which means that a parameter name clearly identifies to which portlet a parameter belongs. Namespaced parameters are visible only to the portlet to which they belong. In addition, IBM portlets may also have access to additional, non-namespaced parameters that have been set on the request.

- If you use the PortletURI.addParameter() method to set parameters, the namespacing is done automatically.
- When the portlet outputs an HTML form, the programmer should explicitly namespace the input fields, including parameter names, by using the <portletAPI:encodeNamespace/> tag. However, if IBM portlets use forms without namespacing, it is guaranteed that they will receive the form parameters.
- It is guaranteed that a portlet can read request parameters that have been set by the `<portal-navigation:urlGeneration/>` tag.

Because of these guarantees, the portal is not always able to filter out parameters that are not intended for the portlet but are nevertheless visible. Therefore, an IBM portlet must be prepared to handle the fact that a PortletRequest may contain additional (non-namespaced) parameters. In the case of parameter name conflicts, a namespaced parameter overrides a non-namespaced parameter and the non-namespaced parameter is not visible.

These same considerations apply to all IBM portlet methods that take a PortletRequest or an event from which a PortletRequest can be retrieved, whether they are triggered directly by a URL or indirectly through the property broker.

Parameter visibility in included JSPs or servlets

When portlets include servlets or JSPs, the included resource will see the same parameters as the including portlet. The servlet specification allows the portlet to add a query string to the include path, as in the following example.

```
context.getRequestDispatcher("/jsps/View.jsp?mode=detailed");
```

Parameters that are set this way will always be visible in the included servlet or JSP, for both IBM and standard portlets. The value will take precedence over an old value for the same parameter name that has been specified by another mechanism. That means, the getParameter() call will return the new value and getParameterValues() will return the new value and any old values.

If the portlet being rendered is an IBM portlet, namespaced parameters still override non-namespaced parameters.

Detailed descriptions of the Struts WML tags

Learn about the WML tags used by the portlets within the Struts Application Framework.

The following tags are used by portlets within the Struts Application Framework. For a brief description of WML support in the Struts Application Framework, see Using the WML tags.

<wml:cancel/>

Renders a WML <postfield> element with a value of `cancel`. This tag is only valid when nested inside a form tag body. Posting this element causes the `Action` servlet to bypass calling the associated form bean validate() method.

Table 464. cancel attributes

Attribute name	Description
property	The parameter name with the specified value that is set in the request object. **Note:** If the property attribute is set, then the application will have to handle cancel detection. [Runtime expression]
value	The value of the request parameter. [Runtime expression]

<wml:card/>

This tag renders a card element. This element is not rendered when the tag is executed in WebSphere Portal Express. This allows writing JSPs that can be used in both the servlet and portlet environments.

Table 465. card attributes

Attribute name	Description
id	This attribute is the card unique identifier. [Runtime expression]
newcontext	This is a flag to indicate that context should be reinitialized when loaded. [Runtime expression]
onenterbackward	Load URL when accessed through `<prev>` task. [Runtime expression]
onenterforward	Load URL when accessed through `<go>` task. [Runtime expression]
ontimer	Load URL when timer expires. [Runtime expression]
ordered	This is the flag to indicate that content is ordered. [Runtime expression]
title	This tag indicates the title of the card. [Runtime expression]

Table 465. card attributes (continued)

Attribute name	Description
titleKey	Key to look up title in resource bundle, titleKey is only used if title attribute is null. [Runtime expression]

<wml:errors/>

Retrieves the set of error messages from the request object with the default key of `Action.ERROR_KEY` or the value specified by attribute name. If `ActionErrors` are found then the errors are displayed. This tag also requires the following two message keys in the application scope MessageResources.

- `errors.header` - header that is displayed before the error messages list.
- `errors.footer` - header that is displayed after the error messages list.

Table 466. error attributes

Attribute name	Description
bundle	This is the servlet context attribute key for the MessageResources instance to use. If not specified, defaults to the application resources configured for the Strut `Action` servlet. [Runtime expression]
locale	The session attribute key for the Locale used to select messages to be displayed. If not specified, defaults to the Struts standard value. [Runtime expression]
name	Name of the request scope bean under which our error messages have been stored. If not present, the name specified by the `Action.ERROR_KEY` constant string will be used. [Runtime expression]
property	Name of the property for which error messages should be displayed. If not specified, all error messages, regardless of property, are displayed. [Runtime expression]

<wml:form/>

This tag does not render any markup, but it is used to scope beans and transactions. The tags used with the body of the form tag can use the form bean to populate the input fields.

Table 467. form attributes

Attribute name	Description
action	The action is the URL that is used for the form submission. The action will be picked up by a nested tag, and postfield tags are used to submit the data. For additional information on specifying the action, see the Struts `html:form` documentation. [Required] [Runtime expression]

Table 467. form attributes (continued)

Attribute name	Description
name	This attribute is the name of the bean. The scope attribute is used to determine where the bean can be located. The tags that are contained in the body of the form tag can use the form bean for populating the input field. For additional information on specifying the name, see the Struts `html:form` documentation. [Runtime expression]
scope	Specifies the scope of the form bean associated with this form. For additional information on the scope attribute, see the Struts `html:form` documentation. [Runtime expression]
type	The fully qualified class name of the form bean. For additional information on the type attribute, see the Struts `html:form` documentation. [Runtime expression]

<wml:go/>

This tag renders a WML <go> element. If the go tag is used in the body of a form tag, the form's action is used as a hyperlink. If the form's action attribute is set, then the `forward`, `href`, and `page` attributes are ignored.

Table 468. go attributes

Attribute name	Description
accept-charset	This attribute allows specifying the character encodings that the application can handle. [Runtime expression]
action	The action is the URL that is used for the form submission. The action will be picked up by a nested tag, and postfield tags are used to submit the data. [Runtime expression]
forward	The name of the global `ActionForward` to be used to create the URL for the go element. **Note:** This attribute is ignored if this tag is specified in the body of a form tag that specifies the action attribute. For addition information on the forward attribute, see the Struts `html:link` documentation. [Runtime expression]
href	This attribute specifies the URL for the go element. **Note:** This attribute is ignored if this tag is specified in the body of a form tag that specifies the action attribute. For addition information on the href attribute, see the Struts `html:link` documentation. [Runtime expression]

Table 468. go attributes (continued)

Attribute name	Description
method	Allows specifying the HTTP submission method, get or post [Runtime expression]
page	The context relative path to the URL that will be used as the href for the go element. **Note:** This attribute is ignored if this tag is specified in the body of a form tag that specifies the action attribute. For addition information on the page attribute, see the Struts `html:link` documentation. [Runtime expression]
sendreferer	This attribute allows specifying that the deck URL should be included the request [Runtime expression]
transaction	A postfield element will be created so that the current transaction control token can be sent if this attribute is `true`. For addition information on the transaction attribute, see the Struts `html:link` documentation. [Runtime expression]
urlType	The type of portlet URL to create. If not specified URL type is standard. Supported values: return, standard [Runtime expression]

<wml:head/>

Renders a WML <head> element with language attributes extracted from the user's current Locale object, if there is one. This element is not rendered when the tag is executed in WebSphere Portal Express. This allows writing JSPs that can be used in both the servlet and portlet environments.

<wml:link/>

Renders a WML <a> element as a hyperlink to the specified URL. URL rewriting will be applied automatically to maintain session state in the absence of cookies. The tag's body is displayed as the name of the link. The base URL for this hyperlink is calculated based on which of the following attributes you specify:

- forward
- href
- page

Note: One and only one of the `forward`, `href`, or `page` attributes can be specified.

Table 469. link attributes

Attribute name	Description
accesskey	A number 0 through 9 that is displayed that indicates to a user which keypad number is required to select this element. [Runtime expression]

Table 469. link attributes (continued)

Attribute name	Description
forward	The name of the global `ActionForward` to be used to create the URL. For addition information on the forward attribute, see the Struts `html:link` documentation. [Runtime expression]
href	This attribute specifies the hyperlink to be unchanged as the URL. For addition information on the href attribute, see the Struts `html:link` documentation. [Runtime expression]
indexed	For information on this attribute, see the Struts `html:link` documentation. [Runtime expression]
indexId	For information on this attribute, see the Struts `html:link` documentation. [Runtime expression]
name	For information on this attribute, see the Struts `html:link` documentation. [Runtime expression]
page	The context relative path to the URL that will be used as the href. For addition information on the page attribute, see the Struts `html:link` documentation. [Runtime expression]
paramId	For information on this attribute, see the Struts `html:link` documentation. [Runtime expression]
paramName	For information on this attribute, see the Struts `html:link` documentation. [Runtime expression]
paramProperty	For information on this attribute, see the Struts `html:link` documentation. [Runtime expression]
paramScope	For information on this attribute, see the Struts `html:link` documentation. [Runtime expression]
property	For information on this attribute, see the Struts `html:link` documentation. [Runtime expression]
scope	For information on this attribute, see the Struts `html:link` documentation. [Runtime expression]
rel	This attribute allows specifying the relationship [Runtime expression]

Table 469. link attributes (continued)

Attribute name	Description
sendreferer	This attribute allows specifying that the deck URL should be included in the request [Runtime expression]
transaction	A postfield element will be created so that the current transaction control token can be sent if this attribute is true. For addition information on the transaction attribute, see the Struts `html:link` documentation. [Runtime expression]
urlType	The type of portlet URL to create. If not specified URL type is standard. Supported values: return, standard [Runtime expression]

<wml:option/>

Renders a WML <option> element, representing one of the choices for an enclosing <select> element. The text displayed to the user comes from either the body of this tag, or from a message string looked up based on the bundle, locale, and key attributes. If the value of the corresponding bean property matches the specified value, this option will be marked selected. This tag is only valid when nested inside a <wml:select> tag body.

Table 470. option attributes

Attribute name	Description
bundle	This attribute allows specifying the key for the MessageResources stored in the servlet context. [Runtime expression]
key	This attribute specifies the key to the text that is contained in the bundled, determined from the bundle attribute. If this attribute is not specified, then the text from the tag's body is used. [Runtime expression]
locale	The locale to use for looking up messages in the resource bundle. [Runtime expression]
onpick	The URL to navigate when a selection is made. [Runtime expression]
title	Brief field title [Runtime expression]
titleKey	Key to look up title in resource bundle, titleKey is only used if title attribute is null. [Runtime expression]
value	If the user selects this option, then this is the value that is submitted. [Required] [Runtime expression]

<wml:options/>

Renders a set of WML <option> elements, representing possible choices for a <select> element. This tag can be used multiple times within a single <wml:select> element, either in conjunction with or instead of one or more <wml:option> elements. The use of the collection attribute is documented in the HTML version of the `options` tag.

Table 471. options attributes

Attribute name	Description
collection	Name of the bean used to build the selection options. For addition information on the collection attribute, see the Struts `html:link` documentation. [Runtime expression]
labelName	For addition information on the labelName attribute, see the Struts `html:link` documentation. [Runtime expression]
labelProperty	For addition information on the labelProperty attribute, see the Struts `html:link` documentation. [Runtime expression]
name	For addition information on the href attribute, see the Struts `html:link` documentation. [Runtime expression]
property	For addition information on the href attribute, see the Struts `html:link` documentation. [Runtime expression]

<wml:password/>

Renders a WML <input> element of type password, populated from the specified value or the specified property of the bean associated with our current form. This tag is only valid when nested inside a form tag body.

Table 472. password attributes

Attribute name	Description
accesskey	A number 0 through 9 that is displayed that indicates to a user which keypad number is required to select this element. [Runtime expression]
emptyok	Flag to indicate that this field can remain blank. [Runtime expression]
format	Format mask for the input field [Runtime expression]
maxlength	Maximum number of input characters to accept. [No limit] [Runtime expression]

Table 472. password attributes (continued)

Attribute name	Description
name	The attribute name of the bean whose properties are consulted when rendering the current value of this input field. If not specified, the bean associated with the form tag we are nested within is utilized. [Runtime expression]
property	Name of the request parameter that will be included with this submission, set to the specified value. [Required] [Runtime expression]
size	Number of character positions to allocate. [Runtime expression]
tabindex	The tab order in a card. [Runtime expression]
title	Brief field title [Runtime expression]
titleKey	Key to look up title in resource bundle, titleKey is only used if title attribute is null. [Runtime expression]
value	Value of the label to be placed on this button. This value will also be submitted as the value of the specified request parameter. [Runtime expression]

<wml:postfield/>

Renders a WML <postfield> element. This tag is only valid when nested inside a form tag body.

Table 473. postfield attributes

Attribute name	Description
name	The attribute name of the bean whose properties are consulted when rendering the current value of this input field. If not specified, the bean associated with the form tag we are nested within is utilized. [Runtime expression]
property	The corresponding bean property. [Required] [Runtime expression]
value	The value of the postfield tag. [Runtime expression]

<wml:rewrite/>

Renders a request URI based on exactly the same rules as the link tag does, but without creating the <a> hyperlink. The base URI for this hyperlink is calculated based on which of the following attributes you specify:

- forward
- href
- page

Note: One and only one of the `forward`, `href`, or `page` attributes can be specified.

Table 474. rewrite attributes

Attribute name	Description
forward	The name of the global `ActionForward` to be used to create the URL. For addition information on the forward attribute, see the Struts `html:link` documentation. [Runtime expression]
href	This attribute specifies the hyperlink to be unchanged as the URL. For addition information on the href attribute, see the Struts `html:link` documentation. [Runtime expression]
indexed	For information on this attribute, see the Struts `html:rewrite` documentation. [Runtime expression]
indexId	For information on this attribute, see the Struts `html:rewrite` documentation. [Runtime expression]
name	For information on this attribute, see the Struts `html:rewrite` documentation. [Runtime expression]
page	The context relative path to the URL that will be used as the href. For addition information on the page attribute, see the Struts `html:rewrite` documentation. [Runtime expression]
paramId	For information on this attribute, see the Struts `html:rewrite` documentation. [Runtime expression]
paramName	For information on this attribute, see the Struts `html:rewrite` documentation. [Runtime expression]
paramProperty	For information on this attribute, see the Struts `html:rewrite` documentation. [Runtime expression]
paramScope	For information on this attribute, see the Struts `html:rewrite` documentation. [Runtime expression]
property	For information on this attribute, see the Struts `html:rewrite` documentation. [Runtime expression]

Table 474. rewrite attributes (continued)

Attribute name	Description
scope	For information on this attribute, see the Struts `html:rewrite` documentation. [Runtime expression]
transaction	A postfield element will be created so that the current transaction control token can be sent if this attribute is `true`. For addition information on the transaction attribute, see the Struts `html:rewrite` documentation. [Runtime expression]

<wml:select/>

Renders a WML <select> element, associated with a bean property specified by our attributes. This tag is only valid when nested inside a form tag body. See the `html:select` for additional information on usage.

Table 475. select attributes

Attribute name	Description
iname	Index number of default option(base 1) [Runtime expression]
ivalue	Default value [Runtime expression]
multiple	Set to support multiple selections. [Runtime expression]
name	The name of the bean used to determine the pre-selected options. [Runtime expression]
property	The attribute sets the name of the request parameter used for the value submission. [Required] [Runtime expression]
tabindex	The tab order in a card. [Runtime expression]
title	Brief field title. [Runtime expression]
titleKey	Key to look up title in resource bundle. `titleKey` is only used if the title is `null`. [Runtime expression]
value	The value for determining if an option has been selected. [Runtime expression]

<wml:text/>

Renders a WML <input> element of type text, populated from the

specified value or the specified property of the bean associated with our current form. This tag is only valid when specified in the body of a form tag.

Table 476. text attributes

Attribute name	Description
accesskey	A number 0 through 9 that is displayed that indicates to a user which keypad number is required to select input. [Runtime expression]
emptyok	Flag to indicate that this field can remain blank.
format	Format mask for input field. [Runtime expression]
maxlength	Maximum number of input characters to accept. [No limit] [Runtime expression]
name	The attribute name of the bean whose properties are consulted when rendering the current value of this input field. If not specified, the bean associated with the form tag we are nested within is utilized. [Runtime expression]
property	Name of this input field, and the name of the corresponding bean property if value is not specified. The corresponding bean property (if any) must be of type String. [Required] [Runtime expression]
size	Number of character positions to allocate. [Runtime expression]
tabindex	The tab order in a card. [Runtime expression]
title	Brief field title. [Runtime expression]
titleKey	Key to look up title in resource bundle. Title must be null. [Runtime expression]
value	Value to which this field should be initialized. [Use the corresponding bean property value] [Runtime expression]

<wml:wml/>

Renders a <wml/> element. In the WebSphere Portal Express environment, the element is not rendered.

Application extension registry

WebSphere Portal Express provides an application extension registry which is equivalent to the application extension registry provided by IBM WebSphere

Application Server. This registry allows any J2EE compliant application to define extension points for other applications to use, or to plug in to other extensible applications.

The application extension registry uses the Eclipse plug-in descriptor format and APIs as the standard extensibility mechanism for WebSphere applications. For more information about the application extension registry, see *Application extension registry* in the *IBM WebSphere Application Server information center* and *Notes on the Eclipse Plug-in Architecture*.

Related tasks:

Application extension registry in WebSphere Application Server

Notes on the Eclipse Plug-in Architecture

Predefined public render parameters

WebSphere Portal defines a set of portal-specific public render parameters, which can be used to work with portal-specific state information within portlets.

Public render parameters is a concept that is defined by the Java Portlet Specification 2.0. As opposed to private render parameters, public render parameters can be shared between portlets to implement coordination use cases.

The main benefit of these predefined parameters is that the portlet developer can rely on the Portlet API to work with portal_specific state information without using any portal-specific APIs or SPIs. For example, you can easily create a portlet render URI that addresses a portal page by using the public render parameter that holds the portal page selection information.

"Pre-defined public render parameters representing portal state" on page 3150
Pre-defined public render parameters that represent portal-specific state information are available in all lifecycle methods of the portlet: processAction, processEvent, render, and serveResource. During rendering, you can create portlet URLs that address these parameters.

"To register pre-defined public render parameters in portlet.xml" on page 3152
To use these pre-defined public render parameters in your portlet, declare each of them in your portlet deployment descriptor (`portlet.xml`) as specified in the Java Portlet specification.

"To use pre-defined public render parameters in your portlet" on page 3153
Creating portlet URLs based on these pre-defined public render parameters is not different from creating any other portlet URL. The typical use case is to create a portlet render URL that addresses one or multiple pre-defined public render parameters.

Related concepts:

"Public render parameters" on page 3070
Public render parameters allow JSR 286 portlets to share navigational state information. They are specially useful for coordinating the multiple navigation or viewer portlets that display different information items that are all related to the same parameter name. The portal stores all portlet render parameters, including public render parameters, as an encoded part of the current portal URL. Therefore, public render parameters are correctly preserved by typical browser navigation actions such as the Back button and bookmarking.

Pre-defined public render parameters representing portal state

Pre-defined public render parameters that represent portal-specific state information are available in all lifecycle methods of the portlet: processAction, processEvent, render, and serveResource. During rendering, you can create portlet URLs that address these parameters.

The following table lists all predefined public render parameters and explains their semantics. All of them represent portal-specific state information and are part of the following namespace:

`http://www.ibm.com/xmlns/prod/websphere/portal/publicparams`

Table 477.

Local name	Value (String array)	Description
selection	Serialized *ObjectID/unique* name of page; for example, `[Z6_0000000AGKJ0G3RCD00]`	Parameter that provides read/write access to the portal page selection information in the navigational state. This parameter can be used to create portlet URLs to portal pages, read the page selection, or modify it during action/event processing.
URI	Serialized *URI*; for example, `[pm:oid:Z3_SASD8732DSG3RCD00]`	Parameter that allows for addressing portal resources that use POC URIs. This parameter can be used to create portlet URLs that address an arbitrary portal resource through its POC URI.
locale	Serialized *locale*; for example, `[en]`	Parameter that provides read/write access to the locale information in the navigational state. This parameter can be used to address a certain locale through a portlet URL, read the locale information from the state, or modify it during action/event processing.

Table 477. (continued)

Local name	Value (String array)	Description
themeTemplate	*Template name*; for example, [Plain]	Parameter that provides read/write access to the theme template that is used during rendering. This parameter can be used to create portlet URLs that enforce a certain theme template that is used for rendering the addressed portal page. The parameter can also be used to read the theme template information from the state or modify it during action / event processing.
editMode	Serialized *boolean value*; for example,[true]	Parameter that provides read/write access to the state of the page edit mode. A value of true means that page edit mode is active; a value of false indicates that it is inactive. This parameter can be used to create portlet URLs that address page edit mode, read the state of the page edit mode or modify it during action/event processing.
infoMode	Serialized *boolean value*; for example, [false]	Parameter that provides read/write access to the state of the page information mode. A value of true means that page information mode is active; a value of false indicates that it is inactive. This parameter can be used to create portlet URLs that address page information mode, read the state of the page edit information mode or modify it during action/event processing.

Table 477. (continued)

Local name	Value (String array)	Description
showTools	Serialized *boolean value*; for example, [true]	Parameter that provides read/write access to the state of the site toolbar; a value of true means that the toolbar is opened, a value of false indicates that it is closed. This parameter can be used to create portlet URLs to open / close the toolbar, read the toolbar state, or modify it during action/event processing.

To register pre-defined public render parameters in portlet.xml

To use these pre-defined public render parameters in your portlet, declare each of them in your portlet deployment descriptor (portlet.xml) as specified in the Java Portlet specification.

The following example shows how to register the parameters that hold the portal page selection and locale information. In this case, the prefix mapping for the portal namespace is defined on the root *portlet-app* element of the deployment descriptor.

Note: The portlet itself does not need to deal with the qualified names of the respective render parameters in its implementation. It can use local identifiers instead, which are defined as part of the respective public-render-parameter element at the end of the descriptor. In this specific case, the page selection parameter is mapped to the local identifier *pageID* whereas the locale parameter is mapped to the local identifier *locale*.

Code Sample:

```
<?xml version="1.0" encoding="UTF-8"?>
<portlet-app id="my.sample.portlet"
     xmlns="http://java.sun.com/xml/ns/portlet/portlet-app_2_0.xsd"
     xmlns:portal="http://www.ibm.com/xmlns/prod/websphere/portal/publicparams"
     version="2.0">
     <portlet>
          <portlet-name>My sample portlet</portlet-name>
          ...
          <supported-public-render-parameter>pageID</supported-public-render-parameter>
          <supported-public-render-parameter>locale</supported-public-render-parameter>
     </portlet>
     <public-render-parameter>
          <identifier>pageID</identifier>
          <qname>portal:selection</qname>
     </public-render-parameter>
     <public-render-parameter>
          <identifier>locale</identifier>
          <qname>portal:locale</qname>
     </public-render-parameter>
</portlet-app>
```

To use pre-defined public render parameters in your portlet

Creating portlet URLs based on these pre-defined public render parameters is not different from creating any other portlet URL. The typical use case is to create a portlet render URL that addresses one or multiple pre-defined public render parameters.

The following code snippet shows how to create a portlet render URL that takes the user to a specific portal page. And to create a portlet render URL that requests the page to be displayed in a certain locale. The URL is created within the render phase of the portlet lifecycle. The sample code assumes that the page selection and locale parameter are registered in the portlet deployment descriptor and mapped to the local identifiers *pageID* and *locale*. For more information, see the sample in *To register pre-defined public render parameters in portlet.xml*.

The logic that determines the values of the parameters is not a part of the code example as it strongly depends on the specific use case. Typically you would use the Model SPI to search for a specific page in the content model and then address the page by using its *ObjectID*. If you want to create deep links to specific pages, you can also address the page through its unique name. In this case, you can set the unique name as the value for the selection parameter.

Code Sample:

```
import com.ibm.portal.ObjectID;
import com.ibm.portal.identification.Identification;
import com.ibm.portal.serialize.SerializationException;

public class MyPortlet extends GenericPortlet {

    /** The identification service. */
    private Identification identification = null;;

    @Override
    public void init() throws PortletException {
        try {
            // default handling
            // ...
            // lookup identification service
            final Context ctx = new InitialContext();
            identification = (Identification) ctx.lookup(Identification.JNDI_NAME);
        } catch (NamingException e) {
           throw new PortletException(e);
        }
   }

   @Override
   protected void doView(final RenderRequest request, final RenderResponse response)
           throws PortletException, IOException {
       try {
           // content type handling, markup generation etc.
           // ...
           // determine the values that should be set
           final ObjectID pageID = getPageID(request);
           final Locale locale = getLocale(request);
           // create a new render URL
           final PortletURL renderURL = response.createRenderURL();
           // set the page selection parameter
           renderURL.setParameter("pageID", identification.serialize(pageID));
           // set the locale parameter
           renderURL.setParameter("locale", locale.toString());
           // write the URL to the markup
           // ...
     } catch (SerializationException e) {
```

```
            throw new PortletException(e);
        }
    }
    ....
```

The IBM Web Content Manager API

You can use the Web Content Manager API to extend functions of Web Content Manager.

Use the Web Content Manager API to:

- create items
- delete items
- move items
- copy items
- retrieve items
- approve items in a workflow
- search for items

See the Javadoc documentation for further information. The Javadoc files for Web Content Manager are in the *PortalServer_root*/doc/Javadoc/spi_docs/com/ibm/workplace/wcm directory.

Getting started

To use the Web Content Manager API, you have 2 options:

1. Write your code as JSPs that you deploy on your server in your Java enterprise application. These JSPs can then be used in your Web Content Manager content by using JSP components or custom JSPs in elements.
2. Write you code as a compiled java enterprise application. In this case, you need to create a project in your development environment that has the two JAR files *ilwwcm-api.jar* and *wp.base.jar* in the build path. These JAR file files can be copied from your portal server from here:
   - /opt/WebSphere/PortalServer/wcm/prereq.wcm/wcm/shared/app/ilwwcm-api.jar
   - /opt/WebSphere/PortalServer/base/wp.base/shared/app/wp.base.jar

   This code can then be used in your Web Content Manager content by using custom plug-ins. See “How to create custom plug-ins” on page 3205 for further information.

   “How to use the Web Content Manager API” on page 3155
   The workspace is the heart of the IBM Web Content Manager API. Items are created, saved, deleted and searched for in the workspace item. A workspace is basically an interface to Web Content Manager that is associated with a user. Using a workspace item, the user can perform operations as that user.

   “The Query API” on page 3156
   The query API provides querying capabilities that are much more rich than the "findBy" methods on the workspace class.

   “Web Content Manager JSP tags” on page 3157
   You use IBM Web Content Manager JSP tags with the Web Content Manager API to pull Web Content Manager content and components into external JSP applications.

"Web content library management APIs" on page 3161
You can perform various web content library functions by using the Web content API.

"Syndication APIs" on page 3163
You can run various syndication functions by using the web content API.

How to use the Web Content Manager API

The workspace is the heart of the IBM Web Content Manager API. Items are created, saved, deleted and searched for in the workspace item. A workspace is basically an interface to Web Content Manager that is associated with a user. Using a workspace item, the user can perform operations as that user.

To get a workspace item, you must first retrieve the `WebContentService`:

```
try
 {
    // Construct and initial Context
    InitialContext ctx = new InitialContext();

    // Retrieve WebContentService using JNDI name
    WebContentService webContentService = (WebContentService)
    ctx.lookup("portal:service/wcm/WebContentService");
 }
 catch (NamingException ne)
 {
    System.out.print("Naming Exception: " + ne);
 }
```

You then request one from the repository singleton with the following call:

```
webContentService.getRepository().getWorkspace("my username", "my password");
```

To get a workspace item without specifying a user name and password, use one of the following calls:

- When used in a portlet: `Workspace workspace = webContentService.getRepository().getWorkspace( (Principal) portletRequest.getUser() );`
- When not used in a portlet: `Workspace workspace = webContentService.getRepository().getWorkspace((Principal) request.getUserPrincipal() );`

If the user is not recognized as a Web Content Manager user, or for some other reason could not be authenticated, an `OperationFailedException` will be thrown.

Note: Only Web Content Manager users (including external LDAP users if enabled) are recognized. For example, A workspace cannot be retrieved using an LTPA token.

Operations available on the workspace include:

- Searching for items with the provided "findBy" methods.
- Creating new items of available editable types.
- Saving and deleting editable items.

You must call `endWorkspace()` when finished with the workspace item.

```
webContentService.getRepository().endWorkspace();
```

Note: You don't need to call `endWorkspace()` when using a JSP component as rendering and session management is handled by Web Content Manager.

Note: You use the `setCurrentDocumentLibrary` method to make calls library-specific. If not specified, the default library that has been configured in the WCM WCMConfigService service is used.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

The Query API

The query API provides querying capabilities that are much more rich than the "findBy" methods on the workspace class.

Here is an example of how to retrieve and use the QueryService to find the first content directly under the SiteArea parentSiteArea that has an element that is called "myElement":

```
Content contentWithMyElement;
QueryService queryService = workspace.getQueryService();
Query query = queryService.createQuery(Content.class);
query.addParentId(parentSiteArea.getId(), QueryDepth.CHILDREN);
try
{
   ResultIterator resultIterator = queryService.execute(query);
   if (resultIterator.hasNext())
   {
      Content childContent = (Content) resultIterator.next();
      while (childContent.hasComponent("myElement"))
      {
         contentWithMyElement = childContent;
         break;
      }
   }
}
catch (QueryServiceException e)
{
   // Handle exception
}
```

Here is an example of how to use create a query with 'AND' and 'OR' conditions by using Conjunction and Disjunction. The example processes all content whose name starts with 'news', or whose name starts with 'article' and has a keyword 'news':

```
QueryService queryService = workspace.getQueryService();
Query query = queryService.createQuery(Content.class);
Disjunction or = new Disjunction();
or.add(Selectors.nameLike("news%"));
Conjunction and = new Conjunction();
and.add(Selectors.nameLike("article%"));
and.add(ProfileSelectors.keywordsContain("news"));
or.add(and);
query.addSelector(or);
try
{
   ResultIterator resultIterator = queryService.execute(query);
   while (resultIterator.hasNext())
   {
```

```
            Content content = (Content) resultIterator.next();

            // Process Content result
        }
    }
    catch (QueryServiceException e)
    {
        // Handle exception
    }
```

Web Content Manager JSP tags

You use IBM Web Content Manager JSP tags with the Web Content Manager API to pull Web Content Manager content and components into external JSP applications.

Note: A JSP referenced within a JSP component must not include a reference, directly or indirectly, to the same JSP component. This includes references within Web Content Manager tags or the API. If it does, a loop is created and your server crashes.

Note: To use the Web Content Manager JSP tags, the following directive must be provided in the JSP:

```
<%@ taglib uri="/WEB-INF/tld/wcm.tld" prefix="wcm" %>
```

Storing JSP files: JSP files are stored within a web application that runs on the portal. To reference a JSP file in another web application, use the following path: `contextPath;jspPath`. For example: `/wps/customapplication;/jsp/jspFilename.jsp`.

A dynamic context path value can be defined by adding a token to the context path that corresponds to a key and value pair to the Web Content Manager configuration service environment provider. When this key is used as the token in the jsp value field, it is replaced dynamically at render time. For example: `my.custom.key;myfile` where *my.custom.key* is a constant within the Web Content Manager configuration service.

Writing JSP to be referenced within a JSP component:

The setExplicitContext and setContext tags are not required when you render a JSP file with a JSP Component. They are only required when directly accessing a JSP file.

Reloading JSP files:

JSP files that are referenced by Web Content Manager are reloaded once every 10 seconds. If you update a JSP file, you might need to wait for it to be reloaded before your changes are displayed.

InitWorkspace tag

This is used to set the initial workspace. This tag:

- Sets the WCM workspace as a local variable called wcmWorkspace.
- Sets the WCM workspace on the pageContext as a parameter with key com.ibm.workplace.wcm.api.Workspace.WCM_WORKSPACE_KEY.
- Sets the WCM RenderingContext on the request as a parameter with key Workspace.WCM_RENDERINGCONTEXT_KEY

```
<wcm:initworkspace username=" " password=" " user=" " >
[Error Message]
</wcm:initworkspace>
```

Table 478. Parameters

Parameter	Details
username	The user name of a valid Web Content Manager user.
user	This parameter is used to specify a java.security.Principal object instead of the user name.
password	The password for the valid Web Content Manager user name or user.

Note: User name, user, and password are optional. If not specified, the current user is used instead, including the anonymous user.

This is an example use of this tag:

```
<%@ taglib uri="/WEB-INF/tld/wcm.tld" prefix="wcm"%>
<%@ page import="com.ibm.workplace.wcm.api.*" %>

<p><wcm:initworkspace>login fail</wcm:initworkspace>
<%
  // Get the workspace for use
  Workspace workspace = (Workspace) pageContext.getAttribute(Workspace.WCM_WORKSPACE_KEY);

  // Get the WCM rendering context for use
  RenderingContext renderingContext = (RenderingContext) request.getAttribute(Workspace.WCM_RENDERIN
%>
```

Explicit context tag

This sets the path to your Web Content Manager server. This is not required in JSP that is displayed using a JSP component.

```
<wcm:setExplicitContext wcmWebAppPath=" " wcmServletPath=" " path=" "
 requestParameters=" " prefix=" " project=" " >
[Error Message]
</wcm:setExplicitContext>
```

Table 479. Parameters

Parameter	Details
wcmWebAppPath	The URL up to the web application. For example: `http://localhost:10040/wps/wcm`
wcmServletPath	The servlet path to the Web Content Manager servlet. For example: `/connect`
path	The path to the content and site areas. For example: `/Site Area A/Site Area B/Content C`
requestParameters	You specify java Map request parameters to set in the context. These parameters can be used by menu components that are rendered via the JSP that use a query string. See "Writing links to web content" on page 1836 for details of request parameters that can be used when referencing web content items.
project	The name of the project to set in the context. If the corresponding project cannot be found, it is ignored and an error is logged. An empty string is used to clear any project that is previously set in the context.

Note: The **project**, **wcmWebAppPath**, and **wcmServletPath** parameters are optional. However, if **wcmWebAppPath** is specified, **wcmServletPath** must also be specified.

Developers can add insert context tags at any place in the page and it changes the context for the rest of the page execution, but the tags cannot be nested.

Context retrieval tag

Sets the context given the location of a path string. This is not required in JSP that is displayed using a JSP component.

```
<wcm:setContext location=" "  wcmWebAppPath=" " wcmServletPath=" " param=" " project=" " defaultPa
[Error Message]
</wcm:setContext>
```

Table 480. Parameters

Parameter	Details
location	This sets the context of the location of a path string. Either: **location="query"** The context is obtained from the query parameter. **location="request"** The context is obtained from the value of the request. **location="session"** The context is obtained from the value of the current session. **location="portalContext"** This is used to define the path of a site area or content item that is used as the current context of a page. For example: `/library1/sitearea3/content4` **location="portalMapping"** This is used to define the path of a site area or content item that is used as the default site area of a page. For example: `/library1/sitearea3`
wcmWebAppPath	The URL up to the web application. For example: `http://localhost:10040/wps/wcm`
wcmServletPath	The servlet path to the Web Content Manager servlet. For example: `/connect`
param	This is the name of the parameter that the path string is in.
project	The name of the project to set in the context. If the corresponding project cannot be found, it is ignored and an error is logged. An empty string is used to clear any project that is previously set in the context.
defaultPath	If the location parameter does not resolve to a valid location, then the value of the **defaultPath** is used. For example: `/library2/sitearea1`

Note: The **project**, **wcmWebAppPath**, **wcmServletPath**, and **defaultPath** parameters are optional. However, if **wcmWebAppPath** is specified, **wcmServletPath** must also be specified.

Developers can add context tags at any place in the page and it changes the context for the rest of the page execution, but the tags cannot be nested.

Rendering tags

These are equivalent to element and component tags.

Rendering an element from the current site area, or content item

```
<wcm:contentComponent type=" " key=" " >
[Error Message]
</wcm:contentComponent>
```

Table 481. Parameters

Parameter	Details
type	This determines where the element is being referenced from. Either *content* or *sitearea*.
key	This is the name of the element that is being referenced.

```
<wcm:libraryComponent name=" " library=" " >
[Error Message]
</wcm:libraryComponent >
```

Rendering a component from the Component Library

Table 482. Parameters

Parameter	Details
name	This is the name of the component that is being referenced.
library	This is the name of the library where the component is stored.

For example:

```
<wcm:libraryComponent name="SC Menu Events" library="Showcase" />
You do not have access to this item.
</wcm:libraryComponent >
```

Rendering Content based on the current context of a page

```
<wcm:content pageDesign=" " >
[Error Message]
</wcm:content >
```

Table 483. Parameters

Parameter	Details
pageDesign	This name of the Presentation Template used to determine context. This parameter is optional.

Error handling

The following tag can be added to error messages to enable error handling:

```
<%=error%>
```

Plug-in tag

Rendering plug-ins can be referenced within JSP code by using a plug-in tag:

```
<wcm:plugin name=" " param1="value" param2="value2" >

// Your text.

</wcm:plugin>
```

See Creating a plug-in tag for further information.

Web content library management APIs

You can perform various web content library functions by using the Web content API.

- Create a library
- Delete a library
- Copy a library
- Import a library
- Export a library

Note: Drafts are not copied or exported when using the API to copy or export libraries.

See the Javadoc documentation for further information. The Javadoc files for Web Content Manager are in the *PortalServer_root*/doc/Javadoc/spi_docs/com/ibm/workplace/wcm directory.

Invoking web content library API methods asynchronously

Although Web content library API functions can be invoked synchronously, if you run these against web content libraries containing large amounts of data, they may take extremely long to complete execution. For example, if these methods are invoked from a JSP page, this may result in the JSP page being invalidated due to a session timeout.

WebSphere Application Server uses a mechanism known as asynchronous beans. An asynchronous bean is a Java object that can be run asynchronously. The "Work object" asynchronous bean is used to invoke web content library API methods asynchronously.

A Work object (which is represented by the interface com.ibm.websphere.asynchbeans.Work) extends java.lang.Runnable It is used to run a block of code as an independent thread. WebSphere Application Server maintains a pool of independent threads that can be assigned to run code encapsulated in Work instances. This pool of threads is managed by the WorkManager. This is used to spawn threads to run Work objects and to monitor them. WebSphere Application Server maintains default Work Managers for each of the application servers that are contained on a particular node. The sample in this topic makes use of the default Work Manager (wm/wpsWorkManager) for the WebSphere Portal application server. This maintains a pool of 300 threads. It is possible to create new Work manager instances with customized thread pools. This is done using the WebSphere Integrated Solutions Console for the WebSphere Portal server.

The example uses the DeleteWork class to implement the Work interface:

```
package deletesample;

import com.ibm.workplace.wcm.api.*;
import javax.naming.*;

public class DeleteWork implements com.ibm.websphere.asynchbeans.Work
{
   private String m_username = null;

   private String m_password = null;

   private String m_libraryToDelete = null;
```

```
   public DeleteWork(String username, String password, String library)
   {
      m_username = username;
      m_password = password;
      m_libraryToDelete = library;
   }

   public void release()
   {
   }

   public void run()
   {
       try
       {
  // Construct and inital Context
      InitialContext ctx = new InitialContext();

  // Retrieve WebContentService and WebContentLibraryService using JNDI name
  WebContentService webContentService = (WebContentService)
     ctx.lookup("portal:service/wcm/WebContentService");
  WebContentLibraryService webContentLibraryService = (WebContentLibraryService)
    ctx.lookup("portal:service/wcm/WebContentLibraryService");

  Workspace ws = webContentService.getRepository().getWorkspace(m_username, m_password);
  DocumentLibrary docLib = ws.getDocumentLibrary(m_libraryToDelete);
  LibraryTaskResult res = webContentLibraryService.deleteLibrary(ws, docLib);

  // Once you get the result object back, print status to StandardOut
  if (res.getResultType() == ResultTypes.OPERATION_SUCCESS)
  {
   System.out.println("Successfully Deleted Library " + m_libraryToDelete);
  }
  else
  {
   System.out.println("Failed To Delete Library " + m_libraryToDelete);
  }
       }
       catch (Exception e)
       {
                 e.printStackTrace();
       }
   }
}
```

The run method is what is required in order to implement this interface. This is where you wrap the library method that you want to run in a thread separate from the calling thread. In the previous example, `DeleteWork` is instantiated passing in credentials as well as the library to be deleted. In `run()`, the repository is logged into and a Workspace instance is obtained as is a `DocumentLibrary` corresponding to the library that is to be deleted. `deleteLibrary()` is then called to perform the actual deletion. Once this method is completed, the Result object can be queried to determine the status of the deletion. This can then be logged or processed as required.

The following JSP file is used to invoke the `DeleteWork` object:

```
<%@ page import="java.util.*" %>
<%@ page import="java.io.*" %>
<%@ page import="java.lang.*" %>
<%@ page import="com.ibm.workplace.wcm.api.*" %>
<%@ page import="com.aptrix.identity.*" %>
<%@ page import="com.ibm.workplace.wcm.services.library.*" %>
<%@ page import="com.ibm.workplace.wcm.services.*" %>
```

```
<%@ page import="com.ibm.workplace.wcm.services.repository.*" %>
<%@ page import="com.ibm.websphere.asynchbeans.*" %>
<%@ page import="javax.naming.*" %>
<%@ page import="deletesample.DeleteWork" %>



<%
/*
 *  JSP Sample to demonstrate how to delete a WCM Library
 *  making use of the WebSphere Application Server
 *  Work asynchronous bean.
 */

try
{
   //obtain a work manager instance - the work manager manages a pool of threads
   //which can be used to invoke the functionality encapsulated
   // within a work instance
   InitialContext ctx  = new InitialContext();
   com.ibm.websphere.asynchbeans.WorkManager wm  =
   (com.ibm.websphere.asynchbeans.WorkManager)
        ctx.lookup("wm/wpsWorkManager");

   //create a new work instance
   DeleteWork workItem = new DeleteWork(request.getParameter("username"),
request.getParameter("password"),
   request.getParameter("library"));
   //spawn a thread to run the create work instance
   wm.startWork(workItem);



}
catch (Exception e)
{
%>  <%= e.toString() %><%
}%>
```

In order to run a Work object, it is necessary to do a JNDI lookup to obtain the default WebSphere Portal Server Work Manager instance. Once this is done, the `DeleteWork` class can be instantiated. To run `DeleteWork` on a separate thread, call `startWork()` on the `WorkManager` passing in the `DeleteWork` instance. For example, `wm.startWork(workItem);` The `System.out` log can be checked to see when `DeleteWork` finishes.

Syndication APIs

You can run various syndication functions by using the web content API.

- Retrieve, enable, or disable syndicators and subscribers.
- Start full and partial syndication updates.
- Dynamically review syndication progress.
- View syndicator and subscriber details such as the name of the libraries that are being syndicated.
- Query syndication details such as which items were updated, saved, modified, or removed.

See the Javadoc documentation for further information. The Javadoc files for Web Content Manager are in the *PortalServer_root*`/doc/Javadoc/spi_docs/com/ibm/workplace/wcm` directory.

Search REST API specification

The following document describes the API call to search IBM WebSphere Portal Express. You can search WebSphere Portal Express to find content that contains a specific text string in its title or content, or is tagged with a specific tag.

Context Paths

There are a number of different context paths available for this API to allow for different authentication mechanisms:

Context Path	Authentication
`/PORTAL_CONTEXT/contenthandler/searchfeed/search`	None
`/PORTAL_CONTEXT/mycontenthandler/searchfeed/search`	Basic

Parameters

The order of the parameters in the requests does not matter. The parameter names are case-sensitive; they must be entered in the format documented. Any unknown or unsupported parameters that are submitted as part of a request is ignored.

The request must be a standard **`HTTP GET`** or **`POST`** command.

- When the request is **`GET`**, the URL is formed by combining the search server's host name, port, and path; and a collection of name-value pairs (input parameters) separated by *&* characters. Any parameter value must be URL-escaped if in **`GET`** request.
- When the request is **`POST`**, the URL is formed by combining the search server's host name, port, and path; and a collection of name-value pairs (input parameters) is passed on the request as parameters.

Name	Description	Comments
`locale`	• Locale of request client • Any message that is returned is in the client locale • Available scopes and collections; titles and description are in the client locale • Sorting method might be effected by the locale • Any additional information in the search results that are locale aware, like description	Specifies the language to use to parse the search request. See *ISO-639* and *ISO-3166* for valid values, for example. *en_US*. This parameter is optional. When specified, the appropriate dictionary for the specified language is used. **Note:** The dictionary for the language that is specified must be enabled for this parameter to work.

Name	Description	Comments
query		• Text to search; returns a list of results with the specified text in the title, description, or content. • Encode the strings. • By default, spaces are treated as an **OR** operator. The following operators are supported: **AND or &&** Searches for items that contain both words. For example: **query=red%20AND%20test** returns items that contain both the word *red* and *test*. **AND** is the default operator. **NOT or !** Excludes the word that follows the operator from the search. For example: **query=test%20NOT%20red** returns items that contain the word *test*, but not the word *red*. **OR** Searches for items that contain either of the words. For example: **query=test%20OR%20red**. To search for a phrase, enclose the phrase in quotation marks (" "). **+** The plus sign indicates that the word must be present in the result. For example: **query=+test%20red** returns only items that contain the word *test* and many that also contain *red*, but none that contain only the word *red*. **?** Use a question mark to match individual characters. For example: **query=te%3Ft** returns items that contain the words *test*, *text*, *tent*, and

Name	Description	Comments
queryLang	Language of the query string	Specifies the language to use to parse the query parameter. See *ISO-639* and*ISO-3166* for valid values, for example, *en_US*. This parameter is optional. When specified, the appropriate dictionary for the specified language is used. **Note:** The dictionary for the language that is specified must be enabled for this parameter to work.
start	Offset to first result to return in results	Defines an offset from the first result in the set. This parameter is ignored if a page parameter is provided. The value starts from *0*. The default is *0*. If specified value is negative, the value is defaulted to *0*; if the specified value is greater than the number of results, no results are returned.
page	Page number	Specifies the page to be returned. The default value is *1*, which returns the first page.
pageSize	Number of results that are wanted for a single request	Specifies the number of entries to return per page. The minimum value is *0* (negative values default to *0*). The default value is *10*. The maximum value that you can specify is *150*.
scope	Identifier of which scope to search; the list of valid scopes is available in the *Scopes API*.	Default is to search all scopes.
sortKey	The key, which controls the sorting order of the search results.	The following values are supported: *date* and *relevance*. A valid value for this parameter is one of these values, or the name of any other sortable field.
sortOrder	Determines the order by which the results are sorted: ascending or descending.	The only valid values are *asc* or *desc*.
constraint	Allows constraining the search results according to the provided criteria.	The provided criteria. For more information, see *Constraints API*.
facet	Specifies which facets are returned for the query, in addition to search results.	Addition to search results. For more information, see *Facets API*.

Name	Description	Comments
index	Specifies which index (collection) to use for the search	For more information, see *Indexes API*.

Examples

/searchfeed/search?**queryLang**=en&locale=en&resultLang=en&query=development &**scope**=1345374377545&start=0&results=10 Search query with query text = *development*.

Response Format

The response is Atom-compliant. The following table describes the significance of the elements that are returned in the response:

Section	Remarks
/feed	The container element for metadata and data that is associated with the search results feed.
/feed/title	Descriptive title of the feed.
/feed/link[@href]	Reference from the feed to a web resource. For more information, see *Feed Paging and Archiving*.
/feed/link[@rel= "next"|"previous"|"first"|"last"]	*first*, *last*, *next*, and *previous* links are included, for supporting.
/feed/author/name	Description of the feed generator.
/feed/id	Permanent, universally unique identifier for the feed.
/feed/updated	Date and time the query was issued. The value conforms to the date-time production in RFC3339.
/feed/openSearch:totalResults	Total number of results for submitted query.
/feed/openSearch:Query	Contains information about the query that was submitted by the user.
/feed/openSearch:Query[@role]	The role attribute value is *request*.
/feed/openSearch:Query[@searchTerms]	Represents the user submitted query terms.
/feed/openSearch:startIndex	Initial result number for the search results returned in this feed.
/feed/openSearch:itemsPerPage	Number of search results that are returned in this feed.
/feed/entry	Encompasses the information for a single search result.
/feed/entry/category	Conveys information about a category (often corresponding to a facet) associated with an entry.
/feed/entry/category@term	A string that identifies the category to which the entry belongs.
/feed/entry/category@scheme	An IRI that identifies a categorization scheme.

Section	Remarks
/feed/entry/title	Text construct that conveys a human-readable title for an entry.
/feed/entry/title[@type]	Indicates whether the text construct is *text*, *html*, or *xhtml*. Text construct is *text* if not otherwise specified.
/feed/entry/link	Defines a reference to the search result resource.
/feed/entry/link[@rel]	Indicates the link relation type. If not present, the link relation type is `alternate`.
/feed/entry/link[@href]	URI link to document.
/feed/entry/link[@type]	Content type of the URI document link is an advisory media type.
/feed/entry/relevance:score	Indicates a relative assessment of relevance for a particular search result with regards to the search query.
/feed/entry/updated	Last modified date for the document. The value conforms to the date-time production in RFC3339.
/feed/entry/id	Unique identifier of the document.
/feed/entry/summary	Text construct that conveys a short summary, abstract, or excerpt of an entry.
/feed/entry/summary[@type]	Indicates whether the text construct is *text*, *html*, or *xhtml*. Text construct is *text* if not otherwise specified.
/feed/entry/author	A person construct that indicates the author of the entry or feed.
/feed/entry/author/name	Human-readable name for the person.
/feed/entry/author/uri	Identifier that is associated with the person.
/feed/entry/author/email	The person email address. Depending on the IBM Connections configuration settings, this value might not be returned as part of the feed.
/feed/entry/wplc:field	This element is used to represent the name and value of a field of a document. The **id** attribute represents the name of the field. The body of the element represents the value of the field. More fields are included in the search result response if specified through the **includeField** parameter.
/feed/ibmsc:facets	For more information, see *Facets*.

- The *namespace* in the table is `http://www.w3.org/2005/Atom` unless otherwise specified.
- The *openSearch* identifier is used to refer to the namespace `http://a9.com/-/spec/opensearch/1.1`.
- The *relevance* identifier is used to refer to the namespace `http://a9.com/-/opensearch/extensions/relevance/1.0/`.
- The *ibmsc* identifier is used to refer to the namespace `http://www.ibm.com/search/content/2010`.

- The *spelling* identifier is used to refer to the namespace `http://a9.com/-/opensearch/extensions/spelling/1.0/`.

Example

To search for all content across IBM WebSphere Portal Express that contains the text *Development*, send the following HTTP request:

```
> GET /searchfeed/search?query=development&scope=com.ibm.lotus.search.ALL_SOURCES HTTP/1.1
```

The following content is returned by the server:

```
<?xml version="1.0" encoding="UTF-8"?>
   <atom:feed xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/"
       xmlns:xhtml="http://www.w3.org/1999/xhtml"
       xmlns:wplc="http://www.ibm.com/wplc/atom/1.0"
       xmlns:atom="http://www.w3.org/2005/Atom">
       <atom:title>Search results for query "development" on scope "com.ibm.lotus.search.ALL_S
       <atom:link href="searchfeed:search" rel="self" type="application/atom+xml"/>
       <atom:author>
           <atom:name>Enterprise Search API Web Service.</atom:name>
       </atom:author>
       <atom:id>searchfeed:search</atom:id>
       <atom:category term="com.ibm.lotus.search.ALL_SOURCES" label="com.ibm.lotus.search.ALL_
       <atom:updated>2013-01-14T08:35:27.482Z</atom:updated>
       <opensearch:totalResults exact="true">412</opensearch:totalResults>
       <opensearch:Query role="request" searchTerms="development"/>
       <opensearch:startIndex>0</opensearch:startIndex>
       <opensearch:itemsPerPage>10</opensearch:itemsPerPage>

       <atom:entry>
          <atom:id>ResourceinjectionusingRationalApplicationDeveloperv7.5</atom:id>
          <atom:title type="text/html">Resource injection using Rational Application Developer
          <atom:author>
             <atom:uri>Dan_Haim</atom:uri>
             <atom:name>Dan Haim</atom:name>
          </atom:author>
          <atom:author>
             <atom:uri>James_Chung</atom:uri>
             <atom:name>James Chung</atom:name>
          </atom:author>
          <atom:link href="http://www.ibm.com/developerworks/rational/library/10/resourceinjec
          <atom:category term="ContentSourceType/default" scheme="com.ibm.wplc.taxonomy://feat
          <opensearch:relevance>100.0</opensearch:relevance>
          <atom:updated>2010-06-07T06:49:09.000Z</atom:updated>
          <atom:summary type="html"><![CDATA[<Strong>Summary:</Strong> Java&#8482; platf....]]
          <atom:link href="/wps/images/icons/Document.gif" rel="icon"/>
          <wplc:field id="name">95c189804d4268bf8d49ede9170f1e3d</wplc:field>
    <wplc:field id="contentSourceType">Seedlist</wplc:field>
   <wplc:field id="defaultcontext">/poc</wplc:field>
          <wplc:field id="effectivedate">1236246335000</wplc:field>
     <wplc:field id="modifier">Replicator</wplc:field>
     <wplc:field id="securecontext">/mypoc</wplc:field>
     <wplc:field id="search_controllable_uuid">2c1e7b59-b465-49da-bc99-5aee3c00932b</wplc:fie
     <wplc:field id="locale">en</wplc:field>
     <wplc:field id="RatingAverage">4</wplc:field>
     <wplc:field id="author_info">Dan_Haim<![CDATA[<Dan Haim<]]></wplc:field>
     <wplc:field id="author_info">James_Chung<![CDATA[<James Chung<]]></wplc:field>
     <wplc:field id="acls">public</wplc:field>
     <wplc:field id="authoringtemplate">Blog Home</wplc:field>
     <wplc:field id="popularity">7811</wplc:field>
     <wplc:field id="security_ids">Z6QReDeIPO2JIT62BDIJM8CKHDAJMG6P1P2MM8C3BEIJMK61BPAMPCCG1C
     <wplc:field id="difficulty">Advanced</wplc:field>
     <wplc:field id="contentPath">/Blog Solo Template v70/Blog/Home/95c189804d4268bf8d49ede91
```

```
        <wplc:field id="category">Rational</wplc:field>
    </atom:entry>
     ...
</atom:feed>
```

"Search scopes REST API specification"
The Scopes API returns the set of supported values that can be passed to the **scope** parameter of the WebSphere Portal Express Search API.

"Search indexes REST API specification" on page 3171
The Indexes API returns the set of supported values that can be passed to the **index** parameter of the WebSphere Portal Express Search API.

"Search constraints REST API specification" on page 3171
Constraints are part of the Search API. They provide a structured method for advanced search over document metadata. They limit search results to the ones that match specific metadata values. It is an alternative for client-side applications, which include filters and advanced search options.

"Search facets REST API specification" on page 3173
The following document describes the **facet** parameter of the Search API and the corresponding response elements. The **facet** parameter allows obtaining the facets, which are relevant for the search query. The facets that are supported in Portal Search include **Tag**, **Person**, **Date**, **Source**, and for status updates only **Trend**.

Related concepts:

"Search" on page 608
You use Portal Search to search for text that is displayed in websites that are created by IBM Web Content Manager.

Related information:

Feed Paging and Archiving

Search scopes REST API specification

The Scopes API returns the set of supported values that can be passed to the **scope** parameter of the WebSphere Portal Express Search API.

Request Format

Context Path	Authentication
/PORTAL_CONTEXT/contenthandler/searchfeed/scopes	Scopes API

Response Format

Response is Atom formatted list of possible values that can be passed to the Search API by using the **scope** parameter of the Search API.

Response Example

```
<atom:feed xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/"
           xmlns:xhtml="http://www.w3.org/1999/xhtml"
           xmlns:atom="http://www.w3.org/2005/Atom">
    <atom:title>Available Scopes</atom:title>
    <atom:link href="searchfeed:scopes" rel="self" type="application/atom+xml"/>
    <atom:author>
       <atom:name>Enterprise Search API Web Service.</atom:name>
    </atom:author>
    <atom:id>[searchfeed:scopes]</atom:id>
    <atom:updated>2013-01-14T14:01:01.837Z</atom:updated>
```

```
    <atom:entry>
        <atom:title>All Sources</atom:title>
        <atom:summary>All Sources accessible by the user</atom:summary>
        <atom:id>com.ibm.lotus.search.ALL_SOURCES</atom:id>
        <opensearch:image>/wps/images/icons/scope_search_all.gif</opensearch:image>
    </atom:entry>
    <atom:entry>
       <atom:title>Managed Web Content</atom:title>
       <atom:summary>Search in WCM</atom:summary>
       <atom:id>com.ibm.lotus.search.MANAGEDWEB</atom:id>
       <opensearch:image>/wps/images/icons/scope_search_wcm.gif</opensearch:image>
    </atom:entry><atom:entry>
    ...

 </atom:feed>
```

Search indexes REST API specification

The Indexes API returns the set of supported values that can be passed to the **index** parameter of the WebSphere Portal Express Search API.

Request Format

Context Path	Authentication
/PORTAL_CONTEXT/contenthandler/searchfeed/indexes	Indexes API

Response Format

Response is Atom formatted list of possible values that can be passed to the Search API by using the **index** parameter of the Search API.

Response Example

```
<atom:feed xmlns:xhtml="http://www.w3.org/1999/xhtml"
            xmlns:atom="http://www.w3.org/2005/Atom">
    <atom:title>Available Indexes</atom:title>
    <atom:author>
       <atom:name>Enterprise Search API Web Service.</atom:name>
    </atom:author>
    <atom:id>[searchfeed:indexes]</atom:id>
    <atom:updated>2013-01-15T13:03:02.295Z</atom:updated>
    <atom:entry>
       <atom:title>Default Search Collection</atom:title>
       <atom:id>D...8a</atom:id>
    </atom:entry>
    <atom:entry>
       <atom:title>ImportTest</atom:title>
       <atom:id>D...st</atom:id>
    </atom:entry>
    ...
 </atom:feed>
```

Search constraints REST API specification

Constraints are part of the Search API. They provide a structured method for advanced search over document metadata. They limit search results to the ones that match specific metadata values. It is an alternative for client-side applications, which include filters and advanced search options.

Instead of concatenating values into the query parameter with specific query syntax, an application can pass the values as separate parameters, regardless of

query syntax. The constraint parameter can be passed multiple times in a Search API request to indicate multiple constraints. The constraint parameter is optional in the Search API. Constraints are mandatory that is each search result must match all the constraints that are provided on the request. If a query parameter is provided, all search results must match the query and the constraints. Constraints are differentiated between three basic types: **field**, **category**, and **range** constraints. Each constraint can apply to single or multiple values. The value of the constraint parameter is a JSON string. The following section outlines the details of the syntax for the various types of constraints.

Field Constraint

A field constraint allows only results that match specific field values.

Syntax: constraint=**type**: *field*, **id**: *field_id*, **values**: *[fieldValue1,fieldValue2]*

Parameters:

Context Path	Authentication
type	Always equals *field* for this type of constraint.
id	The identifier of the indexed field.
values	An array of field values. Each search result must match at least one of the values.

Example: constraint=**type**:*field*, **id**:*title*, **values**:*[test]*

Category Constraint

A category constraint refers to a constraint on a specific category in a *Facet*. A category constraint is similar to a field constraint in its syntax. However, since categories are indexed differently than regular fields, it is declared as a special type of constraint to allow better handling of the request. Category constraints are always an exact match.

Syntax: constraint=**type**: *category*, **values**: [*root/a/x*, *root/a/y]*

Parameters:

Context Path	Authentication
type	Always equals *category* for this type of constraint
values	An array of category IDs. Each search result must match at least one of the categories.

Examples:

- Constraint=**type**: *category*, **values**:*[Tag/tag1]*
- Constraint=**type**: *category*, **values**:*[Tag/tag1,Tag/tag2]*
- Constraint=**type**: *category*, **values**:*[Source/forums,Source/profiles,Source/wikis,Source/status_updates]*
- Constraint=**type**: *category*, **values**:*[Tag/tag1]}*
- Constraint=**type**: *category*, **values**:*[Tag/tag2]*

Range Constraint

A range constraint allows only results in a specific range of field values. Values can be strings or numbers.

Numeric Syntax: constraint=**type**: *range*, **id**: *field_id*, **values**: *[{ge: 0.1, le: 0.5}, {g: 3.6}, {l: -5}]*

String Syntax: constraint=**type**: *range*, **id**: *field_id*, **values**: *[{ge: cat, le: dog}, {g: horse}, {l: animal}]*

Parameters:

Context Path	Authentication
type	Always equals *range* for this type of constraint.
id	The identifier of the indexed field.
values	An array of range values. Each value consists of lower and upper boundaries. Each boundary can be inclusive or exclusive. One or more boundaries can be specified for each value. The allowed attributes are • *ge* for lower inclusive boundary. • *g* for lower exclusive boundary. • *le* for upper inclusive boundary. • *l* for upper exclusive boundary.

Related reference:

"Search facets REST API specification"
The following document describes the **facet** parameter of the Search API and the corresponding response elements. The **facet** parameter allows obtaining the facets, which are relevant for the search query. The facets that are supported in Portal Search include **Tag**, **Person**, **Date**, **Source**, and for status updates only **Trend**.

Search facets REST API specification

The following document describes the **facet** parameter of the Search API and the corresponding response elements. The **facet** parameter allows obtaining the facets, which are relevant for the search query. The facets that are supported in Portal Search include **Tag**, **Person**, **Date**, **Source**, and for status updates only **Trend**.

Request Format

The **facet** parameter of the Search API is optional. When absent, facets are not returned with the search results. The **facet** parameter can repeat multiple times. Each instance of the parameter defines a category for which facets must be computed. For example: **Person**, **Date**, **Tag**, and so on. The value of the **facet** parameter is a JSON string. A number of attributes for the JSON string are

supported. All of the attributes, which have a default value are optional. Any unknown attributes are ignored. The following are the attributes that are supported for the JSON string:

Name	Description	Default Value	Comments
id	Defines the ID of the category from which facets must be computed. This attribute is mandatory.		This value can be obtained from a response of a previous search request.
depth	Defines how to collect facet values in the category. If the depth is *0*, only the category is returned. If the depth is *1*, its immediate children are also returned. If the depth is the constant value *ALL*, then there's no limitation on how deep to collect facet values.	*1*	Negative values are not allowed.
count	Defines the maximal number of values to return for this category. If the count is the constant value *ALL*, then all the category's descendants in the provided depth are returned, up to the maximum.	*10*	Negative values are not allowed. The maximum is *1000*.
sortOrder	Defines how results must be sorted, either in ascending or descending order. Facets results are sorted by weight.	*DESC*	This attribute has only 2 allowed values: *ASC*, *DESC*.

The full structure of the parameter in JSON format is

```
facet={"id": "<facet_id>", "depth": <depth>, "count": <count>, "sortOrder": "<order>"}
```

Examples

Some examples:

```
facet={"id": "Tag", "count": 50}
facet={"id": "Person", "count": 250}
facet={"id": "Date", "count": 250, "depth": 2}
facet={"id": "Source", "count": 50}
facet={"id": "Trend", "count": 50}
```

Response Format

The response format is designed to allow a faceted search client, which is as generic as possible. The format is XML-based, and augments the existing Atom search feed with elements. All additional elements have the *ibmsc* namespace, where *ibmsc* is short for `http://www.ibm.com/search/content/2010`. The root *atom:feed* element is augmented with *ibmsc:facet* elements, as defined in the following schema:

```
atomFeed = element atom:feed {
 ibmscFacets ~//in addition to existing elements
 }

  ibmscFacets = element ibmsc:facets {
 attribute taxonomyId { text},
 ibmscFacet*
 }

  ibmscFacet = element ibmsc:facet {
 attribute id { text},
 attribute type { text },
 ibmscFacetValue*
 }

 ibmscFacetValue = element ibmsc:facetValue{
 attribute id { text},
 attribute label { text },
 attribute weight { float },
 ibmscFacetValue*
 }
```

Description of response elements:

Element	Description
ibmsc:facets	Root element for all facets information.
ibmsc:facets/@taxonomyId	Identifies the facets taxonomy to be used in a category Constraint.
ibmsc:facet	Defines the category for which facets were computed.
ibmsc:facet/@id	The identifier of the category.
ibmsc:facet/@type	Defines the type of facet. Can be used as a hint to the client side on how to render the facet. For example, if **type** is *Person*, it can be rendered as a *live* person link. If **type** is *Tag*, it can be rendered as cloud. These values are predefined: *Date*, *Tag*, *Person*, *String*, *Number*, *Integer*.
ibmsc:facetValue	Defines a category for which search results were found. A **facetValue** might contain sub elements, in a tree-like structure.
ibmsc:facet:facetValue/@id	The identifier of the category for which search results were found.
ibmsc:facetValue/@label	A human-readable label for this facet value.
ibmsc:facetValue/@weight	A number that indicates the relative weight of this facet value within the search results. This number might correlate to the actual number of search results that match this query, but that is not mandatory.

Note: When a facet value is selected by the user, the API provides the following ways to reflect that selection in the faceted search request:

1. To limit search results to the ones that match this facet value, use the **constraint** parameter. Combine the *id* of the **facetValue** element with the **taxonomyId** attribute of the facets element on the response to build the new service URL. This option can also be used to select two-facet values or more simultaneously. The category value must match the *Id* of the selected facet value.
2. To explore facets in a subcategory of the selected facet value if it exists, use the *id* of the **facetValue** element as the *id* attribute of the facet request parameter.

Example

```
<ibmsc:facets taxonomyId="facets">
<ibmsc:facet id="Date" type="Date">
<ibmsc:facetValue id="Date/2012" label="2012" weight="1">
<ibmsc:facetValue id="Date/2012/07" label="07" weight="1"/>
</ibmsc:facetValue>
</ibmsc:facet>
</ibmsc:facets>
```

Related information:

Introducing JSON

Extending tagging and rating by using service APIs

Developers can enhance and extend the tagging and rating features of the portal. For this purpose the portal tagging and rating feature provides service APIs that you can use to enhance tagging and rating by your requirements.

The portal provides a Java Model API and a RESTful API that relates to tagging and rating. You can also extend the included Dojo widget user interface. For example, you can enhance portal tagging and rating in the following ways:

- You can create additional queries about different aspects of the tagging and rating behavior of your portal users.
- You can build additional user interfaces to show the statistics about the tagging and rating behavior of your portal users.
- You can implement additional functionality by exploiting the tags via the access to shared render parameters. Thereby portlets on a page can react to what is currently happening in the tag widget.

"The Java API"
The Java API that the portal provides for tagging and rating follows the pattern of the portal controller SPI. For more details refer to the section about the portal controller API.

"The REST API" on page 3177
The REST API provides various possibilities to work with tagging and rating.

The Java API

The Java API that the portal provides for tagging and rating follows the pattern of the portal controller SPI. For more details refer to the section about the portal controller API.

The following controllers are available for working with resources, tags and ratings:

RatingModelController
Use this controller to control ratings, that is to create or delete ratings.

TagModelController
Use this controller to control tags, that is to create or delete tags.

ResourceModelController
Use this controller to control resources that are tagged or rated or both. For example, specify a category to which a resource belongs.

Related concepts:

"Controller SPI " on page 2883
You can use the Controller SPI for portal administration. It allows you to modify portal resources. It enhances the read-only portal Model SPI by adding writable aspects.

"Model SPI overview" on page 2858
Models provide information that is needed by WebSphere Portal Express to perform tasks such as content aggregation or building navigation to browse the aggregated content. The information that is aggregated is represented through models that can be accessed programmatically by using the Model SPI (read-only). The information of a model is usually persistent (stored in a database) but can also be transient (computed and stored only in memory). Models can be represented by using a tree structure (nodes have a parent-child relationship), a list structure, or a selection structure (a selected element in a tree structure).

Related reference:

"The tagging and rating user interface" on page 1309
Learn about the user interface that IBM WebSphere Portal Express provides for users to work with tags and rating.

The REST API

The REST API provides various possibilities to work with tagging and rating.

"Basic addressing"
Portal users can address resources that have been registered with the tagging and rating engine.

"Adding query parameters" on page 3181
You can add query parameters as required by using the following instructions.

"Querying models in correlation to each other" on page 3185
You can query one model in correlation to another model.

"Other queries" on page 3187
You can add other query parameters, for example for related tags.

"Using the Rest API to add, update, and delete tags and ratings" on page 3190
You can create, update, and delete tags and ratings by using the REST API.

Related reference:

"The tagging and rating user interface" on page 1309
Learn about the user interface that IBM WebSphere Portal Express provides for users to work with tags and rating.

Basic addressing

Portal users can address resources that have been registered with the tagging and rating engine.

Users can address resources, tags, tag spaces, ratings, and rating spaces. They can either address all entities part of these five models or start with an empty set of these models and add single entities by adding additional query parameters. For details refer to the topic about Adding query parameters.

The URL that is to be used for addressing resources is as follows:
`http://host:port/wps/mycontenthandler?uri=uri`

Valid URIs are given in the following sections.

Addressing resources

To address resources, use the following URIs:

rm:all This returns a feed that contains all resources that are registered with the tagging and rating engine. A resource is registered with the tagging and rating engine when a user tags or rates it at least once. Note that using this operation can be rather expensive, depending on the amount of registered resources. Sample feed:

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:feed xmlns:thr="http://purl.org/syndication/thread/1.0"
   xmlns:xhtml="http://www.w3.org/1999/xhtml"
   xmlns:atom="http://www.w3.org/2005/Atom"
   xmlns:model="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model-elements"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:portal="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model"
   xmlns:cp="http://www.ibm.com/xmlns/prod/websphere/portal/v7.0/portal-contextual-portal"
   xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/"
   xmlns:base="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0/ibm-portal-composite-base">
      <atom:author>
         <atom:name>IBM WebSphere Portal/8.0</atom:name>
      </atom:author>
      <atom:id>rm:all</atom:id>
      <atom:link href="/wps/mypoc/!ut/p/digest!YlBr3X0WY5DZfagM7Jz3_A/rm/all?mode=download"
         rel="self" type="application/atom+xml"/>
      <atom:title>WebSphere Portal Server Resource Feed</atom:title>
      <atom:updated>2009-12-21T23:20:05.503Z</atom:updated>
      <atom:entry>
         <atom:title xml:lang="en">A sample page</atom:title>
         <atom:id cp:oid="7_2QC68B1A086070IK5UQSRK20O4">
            rm:pm:oid:7_2QC68B1A086070IK5UQSRK20O4</atom:id>
         <atom:published>2009-12-21T22:51:16.469Z</atom:published>
         <atom:updated>2009-12-21T22:51:16.548Z</atom:updated>
         <atom:link portal:uri="pm:oid:7_2QC68B1A086070IK5UQSRK20O4"
             href="/wps/mypoc/!ut/p/digest!YlBr3X0WY5DZfagM7Jz3_A/pm/oid:
             7_2QC68B1A086070IK5UQSRK20O4?mode=view"
             type="application/atom+xml"/>
</atom:entry>
```

rm:empty
This returns a feed that represents the empty set of resources. You can add individual resources to this feed by adding additional query parameters.

rm:*type_schema*:*ssp*
This returns a feed that represents a single unique resource. Samples:

- For a portal resource: `rm:nm:oid:oid_of_a_page` . This URI addresses a portal content node.
- For a custom resource: `rm:dvd:rambo` . This addresses a custom resource of some kind.

Addressing tags

To address tags, use the following URIs:

tm:all This returns a feed that contains all available tags. Note that using this operation can be rather expensive depending on the amount of registered tags. Sample feed:

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:feed xmlns:thr="http://purl.org/syndication/thread/1.0"
   xmlns:xhtml="http://www.w3.org/1999/xhtml"
   xmlns:atom="http://www.w3.org/2005/Atom"
   xmlns:model="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model-elements"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:portal="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model"
   xmlns:cp="http://www.ibm.com/xmlns/prod/websphere/portal/v7.0/portal-contextual-portal"
   xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/"
   xmlns:base="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0/ibm-portal-composite-base">
```

```
      <atom:author>
         <atom:name>IBM WebSphere Portal/8.0</atom:name>
      </atom:author>
      <atom:id>tm:all</atom:id>
      <atom:link href="/wps/mypoc/!ut/p/digest!YlBr3X0WY5DZfagM7Jz3_A/tm/all?mode=download"
         rel="self" type="application/atom+xml"/>
      <atom:title>WebSphere Portal Server Tag Feed</atom:title>
      <atom:updated>2009-12-21T23:36:52.805Z</atom:updated>
      <atom:entry>
         <atom:title xml:lang="en">sampleTag</atom:title>
         <atom:author>
            <atom:name>wpsadmin</atom:name>
            <atom:uri>um:oid:9eAeK9O86O07M1E6JMG6GHC2JMG61JOIJM4CM1C0JM8CL9E8JO07NHCAJQ86K1</atom:uri>
         </atom:author>
         <atom:id cp:scope="COLLABORATIVE">tm:oid:CI_2QC68B1A086070IK5UQSRK2040</atom:id>
         <atom:published>2009-12-21T22:50:36.509Z</atom:published>
         <atom:updated>2009-12-21T22:50:36.509Z</atom:updated>
         <atom:link portal:uri="tm:oid:CI_2QC68B1A086070IK5UQSRK2O07"
            portal:rel="rm" href="/wps/mypoc/!ut/p/digest!YlBr3X0WY5DZfagM7Jz3_A/rm/oid:
            7_2QC68B1A086070IK5UQSRK2084?mode=download" rel="replies" type="application/atom+xml"/>
         <atom:link portal:uri="tm:oid:CI_2QC68B1A086070IK5UQSRK2040"
            href="/wps/mypoc/!ut/p/digest!YlBr3X0WY5DZfagM7Jz3_A/tm/oid:
            CI_2QC68B1A086070IK5UQSRK2040?mode=download" type="application/atom+xml"/>
</atom:entry>
```

tm:empty
: This returns a feed that represents the empty set of tags. You can add individual tags to this feed by adding additional query parameters.

tm:name:*tag_name*
: This returns a feed that contains all tags that match the name *tagname*.

tm:oid:*oid_of_a_tag*
: This returns a feed that represents a single unique tag with the ID `oid_of_a_tag`.

Addressing tag spaces

Unlike the tag model, the tag space model does not contain information about which user has assigned a certain tag or to which concrete resource a tag has been assigned. A tag space represents only the association between a tag name and its count. Therefore tag spaces are useful to aggregate tag clouds. Use the following URIs:

tm:ts:all
: This returns a feed that contains all available tag spaces, that is all available tags and their names and counts. Sample feed:

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:feed xmlns:thr="http://purl.org/syndication/thread/1.0"
  xmlns:xhtml="http://www.w3.org/1999/xhtml"
  xmlns:atom="http://www.w3.org/2005/Atom"
  xmlns:model="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model-elements"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:portal="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model"
  xmlns:cp="http://www.ibm.com/xmlns/prod/websphere/portal/v7.0/portal-contextual-portal"
  xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/"
  xmlns:base="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0/ibm-portal-composite-base">
    <atom:author>
      <atom:name>IBM WebSphere Portal/8.0</atom:name>
    </atom:author>
    <atom:id>tm:ts:all</atom:id>
    <atom:link href="/wps/mypoc/!ut/p/digest!vLQAhj2WCgHcg0_gV7N7XQ/tm/ts:all?mode=download"
      rel="self" type="application/atom+xml"/>
    <atom:title>WebSphere Portal Server TagSpace Feed</atom:title>
    <atom:updated>2009-12-22T00:00:12.868Z</atom:updated>
    <atom:entry>
      <atom:title>sampleTag</atom:title>
      <atom:id>tm:name:bread</atom:id>
      <atom:published>2009-12-21T22:50:51.698Z</atom:published>
      <atom:updated>2009-12-21T22:50:51.698Z</atom:updated>
      <atom:content type="application/xml">
        <cp:tagspace>
          <cp:supportedLocale>en</cp:supportedLocale>
        </cp:tagspace>
      </atom:content>
      <atom:link thr:count="1" thr:isMine="true" portal:uri="rm:empty"
        portal:rel="rm"
        href="/wps/mypoc/!ut/p/digest!vLQAhj2WCgHcg0_gV7N7XQ/rm/empty?mode=download&tmparam=tm%3aname%3abread"
        rel="replies" type="application/atom+xml"/>
      <atom:link thr:count="1" thr:isMine="true" portal:uri="rm:empty"
        portal:rel="rm"
        href="/wps/mypoc/!ut/p/digest!vLQAhj2WCgHcg0_gV7N7XQ/rm/empty?mode=view&tmparam=tm%3aname%3abread"
        rel="replies" type="text/html"/>
```

```
      <atom:link portal:uri="tm:name:bread"
        href="/wps/mypoc/!ut/p/digest!vLQAhj2WCgHcg0_gV7N7XQ/tm/name:bread?mode=download"
        type="application/atom+xml"/>
</atom:entry>
```

tm:ts:empty
This returns a feed that represents the empty set of tag spaces. You can add individual tag spaces to this feed by adding additional query parameters.

Addressing ratings

To address ratings, use the following URIs:

rtm:all This returns a feed that contains all available ratings. Note that using this operation can be rather expensive depending on the amount of registered ratings. Sample feed:

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:feed xmlns:thr="http://purl.org/syndication/thread/1.0"
  xmlns:xhtml="http://www.w3.org/1999/xhtml"
  xmlns:atom="http://www.w3.org/2005/Atom"
  xmlns:model="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model-elements"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:portal="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model"
  xmlns:cp="http://www.ibm.com/xmlns/prod/websphere/portal/v7.0/portal-contextual-portal"
  xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/"
  xmlns:base="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0/ibm-portal-composite-base">
    <atom:author>
      <atom:name>IBM WebSphere Portal/8.0</atom:name>
    </atom:author>
    <atom:id>rtm:all</atom:id>
    <atom:link href="/wps/mypoc/!ut/p/digest!vLQAhj2WCgHcg0_gV7N7XQ/rtm/all?mode=download"
      rel="self" type="application/atom+xml"/>
    <atom:title>WebSphere Portal Server Rating Feed</atom:title>
    <atom:updated>2009-12-21T23:53:53.912Z</atom:updated>
    <atom:entry>
      <atom:title>4</atom:title>
      <atom:author>
        <atom:name>wpsadmin</atom:name>
        <atom:uri>um:oid:9eAeK9O86O07M1E6JMG6GHC2JMG61JOIJM4CM1C0JM8CL9E8JO07NHCAJQ86K1</atom:uri>
      </atom:author>
      <atom:id cp:scope="COLLABORATIVE">rtm:oid:CJ_2QC68B1A086070IK5UQSRK20C4</atom:id>
      <atom:published>2009-12-21T23:53:18.830Z</atom:published>
      <atom:updated>2009-12-21T23:53:18.830Z</atom:updated>
      <atom:link portal:uri="rtm:oid:CJ_2QC68B1A086070IK5UQSRK20C4"
        ortal:rel="rm" href="/wps/mypoc/!ut/p/digest!vLQAhj2WCgHcg0_gV7N7XQ/rm/oid:
        6_CGAH47LO003H80IKPQVPKO0GS3?mode=download" rel="replies" type="application/atom+xml"/>
      <atom:link portal:uri="rtm:oid:CJ_2QC68B1A086070IK5UQSRK20C4"
        href="/wps/mypoc/!ut/p/digest!vLQAhj2WCgHcg0_gV7N7XQ/rtm/oid:
        CJ_2QC68B1A086070IK5UQSRK20C4?mode=download" type="application/atom+xml"/>
</atom:entry>
```

rtm:empty
This returns a feed that represents the empty set of ratings. You can add individual ratings to this feed by adding additional query parameters.

rtm:oid:*oid_of_a_rating*
This returns a feed that represents a single unique rating with the ID `oid_of_a_rating`.

Addressing rating spaces

Unlike the rating model, the rating space model does not contain information about which user has assigned a certain rating or to which concrete resource a rating has been assigned. A rating space represents only the association between a rating value and its count. Use the following URIs:

rtm:rs:all
This returns a feed that contains all available rating spaces, that is all available ratings and their values and counts. Sample feed:

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:feed xmlns:thr="http://purl.org/syndication/thread/1.0"
   xmlns:xhtml="http://www.w3.org/1999/xhtml"
```

```
    xmlns:atom="http://www.w3.org/2005/Atom"
    xmlns:model="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model-elements"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:portal="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model"
    xmlns:cp="http://www.ibm.com/xmlns/prod/websphere/portal/v7.0/portal-contextual-portal"
    xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/"
    xmlns:base="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0/ibm-portal-composite-base">
       <atom:author>
          <atom:name>IBM WebSphere Portal/8.0</atom:name>
       </atom:author>
       <atom:id>rtm:rs:all</atom:id>
       <atom:link href="/wps/mypoc/!ut/p/digest!vLQAhj2WCgHcg0_gV7N7XQ/rtm/rs:all?mode=download"
          rel="self" type="application/atom+xml"/>
       <atom:title>WebSphere Portal Server RatingSpace Feed</atom:title>
       <atom:updated>2009-12-22T00:13:40.386Z</atom:updated>
       <atom:entry>
          <atom:title>4</atom:title>
          <atom:id>rtm:value:4</atom:id>
          <atom:published>2009-12-21T23:53:18.830Z</atom:published>
          <atom:updated>2009-12-21T23:53:18.830Z</atom:updated>
          <atom:link thr:count="1" thr:isMine="true" portal:uri="rm:empty"
             portal:rel="rm"
             href="/wps/mypoc/!ut/p/digest!vLQAhj2WCgHcg0_gV7N7XQ/rm/empty?mode=
                 download&rtmparam=rtm%3avalue%3a4"
             rel="replies" type="application/atom+xml"/>
          <atom:link
             portal:uri="rtm:value:4"
             href="/wps/mypoc/!ut/p/digest!vLQAhj2WCgHcg0_gV7N7XQ/rtm/value:4?mode=download"
             type="application/atom+xml"/>
</atom:entry>
```

rtm:rs:empty

This returns a feed that represents the empty set of rating spaces. You can add individual rating spaces to this feed by adding additional query parameters.

Related reference:

"Adding query parameters"
You can add query parameters as required by using the following instructions.

"Querying models in correlation to each other" on page 3185
You can query one model in correlation to another model.

"Other queries" on page 3187
You can add other query parameters, for example for related tags.

Adding query parameters

You can add query parameters as required by using the following instructions.

Sorting the results

Use the query parameters `orderMetric` and `order` to sort entries of a result set.

orderMetric

Use this parameter to specify by what metric you want to sort the result list. Valid values depend on the model that you are using:

For the resource model:

- RESOURCE_ID
- RESOURCE_CREATION_DATE
- RESOURCE_LAST_MODIFIED_DATE
- RESOURCE_TYPE_SCHEMA
- RESOURCE_SCHEME_SPECIFIC_PART
- RESOURCE_URI
- RESOURCE_TITLE

For the tag model:

- TAG_ID
- TAG_CREATION_DATE

- TAG_LAST_MODIFIED_DATE
- TAG_LOCALE
- TAG_NAME
- TAG_RESOURCE_ID
- TAG_OWNER_ID
- TAG_SCOPE

For the tag space model:

- TAG_SPACE_COUNT_REVERSE_NAME. This is the default.
- TAG_SPACE_NAME
- TAG_SPACE_COUNT
- TAG_SPACE_CREATION_DATE
- TAG_SPACE_LAST_MODIFIED_DATE
- TAG_SPACE_COUNT_NAME

For the rating model:

- RATING_ID
- RATING_CREATION_DATE
- RATING_LAST_MODIFIED_DATE
- RATING_VALUE
- RATING_RESOURCE_ID
- RATING_OWNER_ID
- RATING_SCOPE

For the rating space model:

- RATING_SPACE_VALUE
- RATING_SPACE_COUNT
- RATING_SPACE_CREATION_DATE
- RATING_SPACE_LAST_MODIFIED_DATE

order=ASC|DESC
Use this parameter to specify whether you want to sort in ascending or descending order. Valid values are `ASC` and `DESC`.

Example:

```
tm:ts:all&orderMetric=TAG_SPACE_COUNT_NAME&order=ASC
```

This returns a feed that contains all available tag spaces, that is all available tags and their names and counts. The results are sorted in ascending order, first by tag count, and, if counts are equal, by tags name.

Limiting the results

You can use the query parameters `start-index` and `max-results` to limit the query results in the feed to a subset.

start-index
Use this parameter to specify the first item from the overall result set that you want to have returned.

max-results
Use this parameter to specify how many additional items after the start item you want to have returned.

Examples:

1. `uri=tos:typeahead&term=A&max-results=10`

   This example returns a feed that contains at the most the first 10 elements of the result.
2. `uri=tos:typeahead&term=A&start-index=20&max-results=10`

   This example returns a feed that contains at the most 10 elements, starting with the 20th element from the result.
3. `tm:name:`*`tag_name`*`&start-index=5&max-results=5`

   This example returns a feed that contains 5 tag entries of the overall result set that match the name `tag_name`, leaving out the first four results, and returning the following 5 tag entries.

Using scopes

Users can apply both tags and ratings as community tags or personal tags, either public or private. For details refer to the topic about how tagging and rating works in the portal. To control whether you query only community or personal tags or ratings, or both types of tags or ratings, use the parameter `scope`. Valid values are as follows:

For tagging:

COMMUNITY
Use this value to return only community tags.

PERSONAL_PRIVATE
Use this value to return only personal private tags.

PERSONAL_PUBLIC
Use this value to return only personal public tags.

PERSONAL
Use this value to return all personal tags, both public and private. This has the same effect as using the scope parameter with both values `PERSONAL_PRIVATE` and `PERSONAL_PUBLIC` .

ALL Use this value to return all tags, community tags, and personal public and private tags. This has the same effect as using the scope parameter with multiple values `PERSONAL_PRIVATE` , `PERSONAL_PUBLIC`, and `COMMUNITY` .

Valid combinations of values are as follows:

- `COMMUNITY` and `PERSONAL`
- `COMMUNITY` and `PERSONAL_PRIVATE`
- `COMMUNITY` and `PERSONAL_PUBLIC`
- `COMMUNITY` and `PERSONAL_PUBLIC` and `PERSONAL_PRIVATE`
- `PERSONAL_PUBLIC` and `PERSONAL_PRIVATE`

For rating:

COMMUNITY
Use this value to return only community ratings.

PERSONAL_PRIVATE
Use this value to return only personal private ratings.

PERSONAL_PUBLIC
Use this value to return only personal public ratings.

PERSONAL
Use this value to return all personal ratings, both public and private. This has the same effect as using the scope parameter with both values `PERSONAL_PRIVATE` and `PERSONAL_PUBLIC` .

ALL Use this value to return all ratings, community ratings, and personal public and private ratings. This has the same effect as using the scope parameter with multiple values `PERSONAL_PRIVATE` , `PERSONAL_PUBLIC`, and `COMMUNITY` .

Valid combinations of values are as follows:

- `COMMUNITY` and `PERSONAL`
- `COMMUNITY` and `PERSONAL_PRIVATE`
- `COMMUNITY` and `PERSONAL_PUBLIC`
- `COMMUNITY` and `PERSONAL_PUBLIC` and `PERSONAL_PRIVATE`
- `PERSONAL_PUBLIC` and `PERSONAL_PRIVATE`

Example:

```
tm:ts:all&scope=COMMUNITY&scope=PERSONAL_PUBLIC
```

This returns a feed that contains all available tag spaces with community tags and the public tags of the current user.

Locale sensitive queries

You can specify that you want to search only a specific locale or set of locales. This can be useful when you work with URIs that address tags by their name locales. To do this, use the parameter `locale`. Examples:

```
tm:name:tag_name&locale=de
```

This returns a feed that contains all tags that match the name *`tag_name`* in the locale `de`.

```
tm:name:tag_name&locale=de&locale=en
```

This returns a feed that contains all tags that match the name *`tag_name`* in the locale `de` or `en`.

Related concepts:

"Introduction to tagging and rating" on page 1299
Tagging and rating are features that support collaboration and interaction between users when using Web content.

Related reference:

"Normalizing tags" on page 1305
The portal provides several options for normalizing tags. Normalization is a process of transforming a text fragment, such as a tag, into another, more generic representation. This bundles different spellings or grammatical versions of the same lexical word that users might use as tags, for example `color`, `Color`, `COLOR`, `colour`, `colors`, `colored`.

"How tagging and rating works in the portal" on page 1303
Use these topics for general administrative information about tagging and rating in the portal.

"Grouping tags and ratings via resource categorization" on page 1304
Users apply tags and ratings to resources. Users can tag and rate all resources that can be uniquely identified.

"How public and private tags and ratings work in the portal" on page 1303
Users can choose between applying a tag or rating as a private or public tag or rating.

Querying models in correlation to each other

You can query one model in correlation to another model.

To do this, you start with an empty set of one of the available models, and add single entries by using additional query parameters. For details about empty sets and additional query parameters refer to the topic about Adding query parameters.

Examples:

- To correlate resources with tags, you start with the feed that represents an empty set of resources and add single resources based on how these have been tagged.
- To correlate resources with ratings, you start with the feed that represents an empty set of resources and add single resources based on how these have been rated.
- To correlate tags with resources you start with the feed that represents an empty set of tags and add single tags based on the resources to which these tags have been assigned.
- To correlate ratings with resources, you start with the feed that represents an empty set of ratings and add single ratings based on the resources to which these ratings have been assigned.

Note: You cannot correlate a model with itself. For example, the tag model query parameter `tm_param` is not supported for the tag model; equally, the rating model query parameter `rtm_param` is not supported the rating model, and so on. The only exception from this rule is to use the URI `tm:name:related`, which is listed under the topic about *Other queries*.

Correlating the resource model with other models

To correlate the resource model with other models, use the following parameters:

rm:empty&tmparam=tm:name:*tag_name*
: This returns a feed that contains all resources that have been tagged with the tag `tagname`.

rm:empty&tmparam=tm:name:*tag_name*&rmparam=rm:type:*type*
: This returns a feed that contains all resources that are registered with type schema `type` and that have been tagged with the tag `tagname`.

rm:empty&tmparam=tm:name:*tag_name*&rmparam=rm:category:*category_name*
: This returns a feed that contains all resources that are registered under the category `category_name` and that have been tagged with the tag `tag_name`.

Correlating the tag model with other models

To correlate the tag model with other models, use the following parameters:

tm:empty&rmparam=rm:*resource_uri*
: This returns a feed that contains all tags for a resource with the URI `resource_uri`.

tm:empty&rmparam=rm:*resource_uri_1*&rmparam=rm:*resource_uri_2*
This returns a feed that contains all tags for resources with the URIs `resource_uri_1` or `resource_uri_2` .

Correlating the tag space model with other models

To correlate the tag space model with other models, use the following parameters:

tm:ts:empty&rmparam=rm:*resource_uri*
This returns a feed that contains all tag spaces for a resource with the URI `resource_uri`.

tm:ts:empty&rmparam=rm:*resource_uri_1*&rmparam=rm:*resource_uri_2*
This returns a feed that contains all tag spaces for resources with the URIs `resource_uri_1` or `resource_uri_2` .

tm:ts:empty&rmparam=rm:type:*type*
This returns a feed that contains all tag spaces for resources registered with the type schema `type`.

tm:ts:empty&rmparam=rm:category:*category*
This returns a feed that contains all tag spaces for resources registered in category `category`.

tm:ts:empty&rmparam=rm:type:*type_1*&rmparam=rm:type:*type_2*
This returns a feed that contains all tag spaces for resources registered with the type schema `type_1` or `type_2` .

tm:ts:empty&rmparam=rm:category:*category_1*&rmparam=rm:category:*category_2*
This returns a feed that contains all tag spaces for resources registered in category `category_1` or `category_2` .

Correlating the rating model with other models

rtm:empty&rmparam=rm:*resource_uri*
This returns a feed that contains all ratings for a given resource with the URI `resource_uri` .

rtm:empty&rmparam=rm:*resource_uri1*&rmparam=rm:resource_uri2
This returns a feed that contains all ratings for a given resource with the URIs `resource_uri_1` or `resource_uri_2` .

Correlating the rating space model with other models

rtm:rs:empty&rmparam=rm:*resource_uri*
This returns a feed that contains all rating spaces for a given resource with the URI `resource_uri` .

rtm:rs:empty&rmparam=rm:*resource_uri1*&rmparam=rm:resource_uri2
This returns a feed that contains all ratings for a given resource with the URIs `resource_uri_1` or `resource_uri_2` .

rtm:rs:empty&rmparam=rm:type:*type*
This returns a feed that contains all rating spaces for resources registered with the type schema `type` .

rtm:rs:empty&rmparam=rm:category:*category*
This returns a feed that contains all rating spaces for resources registered in the category `category` .

rtm:rs:empty&rmparam=rm:type:*type_1*&rmparam=rm:type:*type2*
This returns a feed that contains all rating spaces for resources registered with type schema `type_1` or `type_2` .

rtm:rs:empty&rmparam=rm:category: *category_1*
&rmparam=rm:category:*category_2*
This returns a feed that contains all rating spaces for resources registered in category `category_1` or `category_2` .

Related reference:

"Adding query parameters" on page 3181
You can add query parameters as required by using the following instructions.

Other queries

You can add other query parameters, for example for related tags.

The query for related tags can be useful to search for related content, that is for resources that are or can be similar to a given resource. Resources are considered to be similar if they share at least one tag with a given resource.

Example: To search for resources `r2 . . . rn` that are related to a resource `r1`, you determine the tags of resource `r1` and hand them over to the following query:
`tm:ts:related&tmparam=tm:name:`*`tagname`*

This query returns a feed that contains all tags related to the tag names or set of tags that you specified. The query results in a list of all tags assigned to all resources that have been assigned at least one of the tags that resource `r1` has also been assigned. In other words, This query lists all tags of all similar or related resources `r2 . . . rn`. You can now use this list to query for all these listed resources `r2...rn`.

"Type-ahead with the deprecated tag widget"
The tag widget from earlier IBM WebSphere Portal Express versions provided a query for the type-ahead feature. With portal V 8.5, that tag widget is deprecated. The type-ahead feature makes it easier for users to find suitable tags. Type-ahead supports users when they work with tags. For example, when users apply tags using the tag widget, or when they search for tags, for example by using the open search functionality, type-ahead provides users suggestions for tags that other users have applied already before. Type-ahead can also help reduce the number of variants of tags.

"Search suggestions for tag names" on page 3189
To make search for tag names by users easier, search suggestions provide suggestions for tag names as they appear in the tag cloud and in the tag result portlet.

"Querying for the OpenSearch description document" on page 3190
You can query for the OpenSearch description document.

Type-ahead with the deprecated tag widget:

The tag widget from earlier IBM WebSphere Portal Express versions provided a query for the type-ahead feature. With portal V 8.5, that tag widget is deprecated. The type-ahead feature makes it easier for users to find suitable tags. Type-ahead supports users when they work with tags. For example, when users apply tags using the tag widget, or when they search for tags, for example by using the open search functionality, type-ahead provides users suggestions for tags that other users have applied already before. Type-ahead can also help reduce the number of variants of tags.

Note: The type-ahead feature works only with the dialog tag widget of the default tagging user interface of portal versions earlier than V 8.5. With WebSphere Portal Express V 8.5, the tag and rating widgets of earlier portal versions are deprecated. To query for the type-ahead feature, use the following query:

`tos:typeahead&term=`*`term`*

where `term` represents the beginning of a term that a user is typing for which you want the portal to provide typeahead suggestions.

The returned feed is by default sorted by relevance. This means that tags that have been applied more often are displayed before tags that have been applied less often.

This query allows you to modify all of the following:

- The parameters `order`, `orderMetric`, `locales` . The parameters control how the type-ahead result looks. For more details see the topic about *Adding query parameters* under the sections about *Sorting the results* and *Locale sensitive queries*.
- The pagination of the type-ahead results, that is with which item the result list starts and how many results are displayed. For more details see the topic about *Adding query parameters* under the section about *Limiting the results*.

You can configure how the tag names are displayed in terms of normalization. To do this, use the setting `com.ibm.wps.cp.tagging.normalization.typeAhead` in the CP Configuration Service. The response of the suggestions is in the JSON (JavaScript Object Notation) format and consists of an array that contains the searched partial term and suggestions.

Example: If the search term is `web`, then the response to `tos:typeahead&term=web` contains the suggestions `WebSphere`, `Website`, and `WebBrowser`. The notation in JSON is as follows in the mime type application/json: `["Web",["WebSphere", "Website", "WebBrowser"]]` . This is the format proposed by OpenSearch Suggestions extension Version 1.1 draft 1.

Related reference:

"Adding query parameters" on page 3181
You can add query parameters as required by using the following instructions.

"Type-ahead feature for the deprecated tag widget" on page 1306
The tag widget from earlier IBM WebSphere Portal Express versions provided a type-ahead feature. With portal V 8.5, that tag widget is deprecated. The type-ahead feature makes it easier for users to find suitable tags. Type-ahead supports users when they work with tags. For example, when users apply tags using the tag widget, or when they search for tags, for example by using the open search functionality, type-ahead provides users suggestions for tags that other users have applied already before. Type-ahead can also help reduce the number of variants of tags.

"Normalizing tags" on page 1305
The portal provides several options for normalizing tags. Normalization is a process of transforming a text fragment, such as a tag, into another, more generic representation. This bundles different spellings or grammatical versions of the same lexical word that users might use as tags, for example `color`, `Color`, `COLOR`, `colour`, `colors`, `colored`.

Related information:

OpenSearch Suggestions extension Version 1.1 draft 1: http://www.opensearch.org/Specifications/OpenSearch/Extensions/Suggestions/1.1

Search suggestions for tag names:

To make search for tag names by users easier, search suggestions provide suggestions for tag names as they appear in the tag cloud and in the tag result portlet.

This is important with respect to normalization as you can configure the display mode of the tags in a tag cloud independently from the type-ahead suggestions. Compared to the query for the type-ahead feature, the search suggestions behave in terms of normalization like the display mechanism of the Tag Cloud.

To query for search suggestions, use the following query:

`tos:searchsuggestions&term=`*`term`*

where `term` represents the beginning of a term that a user is typing for which you want the portal to provide search suggestions.

The returned feed is by default sorted by relevance. This means that tags that have been applied more often are displayed before tags that have been applied less often.

This query allows you to modify all of the following:

- The parameters `order`, `orderMetric`, `locales` . The parameters control how the type-ahead result looks. For more details see the topic about *Adding query parameters* under the sections about *Sorting the results* and *Locale sensitive queries*.
- The pagination of the type-ahead results, that is with which item the result list starts and how many results are displayed. For more details see the topic about *Adding query parameters* under the section about *Limiting the results*.

Note: The type-ahead feature works only with the dialog tag widget of the default tagging user interface of portal versions earlier than V 8.5. With WebSphere Portal Express V 8.5, the tag and rating widgets of earlier portal versions are deprecated.

You can configure how the tag names are displayed in terms of normalization. To do this, use the setting `com.ibm.wps.cp.tagging.normalization.displayNormalizedName` in the CP Configuration Service. The response of the suggestions is in the JSON (JavaScript Object Notation) format and consists of an array that contains the searched partial term and suggestions.

Example: If the search term is `web`, then the response to `tos:searchsuggestion &term=web` contains the search suggestions `WebSphere`, `Website`, and `WebBrowser`. The notation in JSON is as follows in the mime type application/json: `["Web",["WebSphere", "Website", "WebBrowser"]]` . This is the format proposed by OpenSearch Suggestions extension Version 1.1 draft 1.

Related reference:

"Adding query parameters" on page 3181
You can add query parameters as required by using the following instructions.

"Normalizing tags" on page 1305
The portal provides several options for normalizing tags. Normalization is a process of transforming a text fragment, such as a tag, into another, more generic representation. This bundles different spellings or grammatical versions of the same lexical word that users might use as tags, for example `color`, `Color`, `COLOR`, `colour`, `colors`, `colored`.

Related information:

OpenSearch Suggestions extension Version 1.1 draft 1: http://www.opensearch.org/Specifications/OpenSearch/Extensions/Suggestions/1.1

Querying for the OpenSearch description document:

You can query for the OpenSearch description document.

To query for the OpenSearch description document, use the following query:
`tos:search&osbu=`*`bundle-name`*`&osde=`*`key-description`*`&ossn=`*`key-short-name`*`&osic16=`*`url-icon-16`*

The following query parameters are mandatory with this query:

osbu Use this query parameter to specify the base name of a resource bundle. Specify a fully qualified Java class name.

osde Use this query parameter to specify the key within the previous given resource bundle the value of which you want to be used as the description of the OpenSearch description document.

ossn Use this query parameter to specify the key within the previous given resource bundle the value of which you want to be used as the short name of the OpenSearch description document.

Optionally, you can specify the following query parameter:

osic16
Use this query parameter to pass a URL for an icon that browsers can render to visualize the OpenSearch description document. For details about the characteristics of the icon refer to the JavaDoc of the interface `com.ibm.portal.resolver.opensearch.DefaultOpenSearchContentHandler.startImage(int, int, String)` . Note that the icon must be 16 pixels by 16 pixels.

Related concepts:

"Searching for tagged content" on page 610
Users can search the tag space by using the browser search box.

Related information:

Open Search - http://www.opensearch.org/Home

Using the Rest API to add, update, and delete tags and ratings

You can create, update, and delete tags and ratings by using the REST API.

Creating and deleting tags by using the REST API

To create a tag, send an HTTP POST against the URI `tm:ts:all` with the following payload:

```
<?xml version="1.0" encoding="UTF-8"?>',
<atom:feed xmlns:atom="http://www.w3.org/2005/Atom"
   xmlns:xhtml="http://www.w3.org/1999/xhtml"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:creation-context=
      "http://www.ibm.com/xmlns/prod/websphere/portal/v6.1.0/portal-creation-context"
   xmlns:trc="http://www.ibm.com/xmlns/prod/websphere/portal/v7.0/portal-tag-rate-comment"
   xmlns:cp="http://www.ibm.com/xmlns/prod/websphere/portal/v7.0/portal-contextual-portal"
   xmlns:portal="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model"
   xmlns:thr="http://purl.org/syndication/thread/1.0"
   xmlns:model="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model-elements"
   xmlns:base="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0/ibm-portal-composite-base"
   xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/"
   xml:base="http://localhost/wps/poc">
```

```
      <atom:author>
         <atom:name>IBM WebSphere Portal/8.5</atom:name>
      </atom:author>
      <atom:title>IBM WebSphere Portal Model Feed</atom:title>
      <atom:link href="." rel="self" type="application/atom+xml"/>
      <atom:entry creation-context:private="${isPrivate}">
         <atom:title xml:lang="${locale}">${tagname}</atom:title>
         <atom:id>${id}</atom:id>
         <atom:link portal:uri="${resourceURI}" rel="replies" href="example.org" />
         <atom:category term="${category}" />
      </atom:entry>
</atom:feed>
```

Replace the variables `isPrivate`, `tagname`, `resourceURI`, and `category` (optional) by the appropriate values.

To delete a tag, send an HTTP DELETE against the URI `uri tm:oid:`*`oid_of_a_tag`*.

Creating and deleting ratings by using the REST API

To create a rating, send an HTTP POST against the URI `rtm:rs:all` with the following payload:

```
<?xml version="1.0" encoding="UTF-8"?>',
<atom:feed xmlns:atom="http://www.w3.org/2005/Atom"
   xmlns:xhtml="http://www.w3.org/1999/xhtml"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:creation-context=
      "http://www.ibm.com/xmlns/prod/websphere/portal/v6.1.0/portal-creation-context"
   xmlns:trc="http://www.ibm.com/xmlns/prod/websphere/portal/v7.0/portal-tag-rate-comment"
   xmlns:cp="http://www.ibm.com/xmlns/prod/websphere/portal/v7.0/portal-contextual-portal"
   xmlns:portal="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model"
   xmlns:thr="http://purl.org/syndication/thread/1.0"
   xmlns:model="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0.1/portal-model-elements"
   xmlns:base="http://www.ibm.com/xmlns/prod/websphere/portal/v6.0/ibm-portal-composite-base"
   xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/"
   xml:base="http://localhost/wps/poc">
      <atom:author>
         <atom:name>IBM WebSphere Portal/8.5</atom:name>
      </atom:author>
      <atom:title>IBM WebSphere Portal Model Feed</atom:title>
      <atom:link href="." rel="self" type="application/atom+xml"/>
         <atom:entry creation-context:private="${isPrivate}">
            <atom:title>${rating}</atom:title>
            <atom:id>${id}</atom:id>
            <atom:link portal:uri="${resourceURI}" rel="replies" href="example.org" />
            <atom:category term="${category}" />
         </atom:entry>
</atom:feed>
```

Replace the variables `isPrivate`, `rating`, `resourceURI`, and `category` (optional) by the appropriate values.

To update a rating, send an HTTP PUT that contains the same payload against the same URI.

To delete a rating, send an HTTP DELETE against the URI `uri rtm:oid:`*`oid_of_a_rating`*.

Related reference:

"The tagging and rating user interface" on page 1309
Learn about the user interface that IBM WebSphere Portal Express provides for users to work with tags and rating.

How to create a custom launch page

You can configure an authoring portlet to use a launch page of your own design instead of the default user interface.

A custom launch page can either be a JSP or HTML file. You use remote actions to call different views and functions from with the authoring portlet's user interface. You can also use the web content API to add other functions to your launch page. When you create a custom launch page, you configure your authoring portlet to use the custom launch page instead of the default authoring portlet user interface.

Storing JSP files: JSP files are stored within a web application that runs on the portal. To reference a JSP file in another web application, use the following path: `contextPath;jspPath`. For example: `/wps/customapplication;/jsp/jspFilename.jsp`.

A dynamic context path value can be defined by adding a token to the context path that corresponds to a key and value pair to the Web Content Manager configuration service environment provider. When this key is used as the token in the jsp value field, it is replaced dynamically at render time. For example: `[my.custom.key];myfile` where *my.custom.key* is a constant within the Web Content Manager configuration service.

A custom launch page example

This is a simple example of some code you can add to a JSP or HTML file, to allow users to create and view content items by using remote actions.

```
<%--
/* Sample Launch Page */
--%>

<%@ taglib uri="/WEB-INF/tld/wcm.tld" prefix="wcm" %>

<%--
/* Add your username and password here */
--%>
<wcm:initworkspace username="wpsadmin" password="wpsadmin" >
An error occurred initializing the WCM workspace:
<%=error%>
</wcm:initworkspace>

<%--
/* Setup your context here */
--%>
<wcm:setExplicitContext wcmWebAppPath="http://localhost:10039/wps/wcm"
wcmServletPath="/myconnect" path="Web Content/Articles/Sample Article" >
An error occurred setting the WCM context:
<%=error%>
</wcm:setExplicitContext>

<div>
   <div>
      <a href='<wcm:plugin name="RemoteAction" action="new"
type="com.ibm.workplace.wcm.api.WCM_Content"/>'>New Content</a>
   </div>
   <div>
      <a href='<wcm:plugin name="RemoteAction" view="contentbytitle"
action="openmainview"/>'>View Content by Title</a>
   </div>
   <div>
```

```
 <a href='<wcm:plugin name="RemoteAction" view="explorer"
action="openmainview"/>'>Open the Library Explorer view</a>
        </div>
</div>
```

Reloading JSP files:

JSP files that are stored in the PA_WCM_Authoring_UI application are not reloaded, even if they are updated. You must restart the server, or put the JSP in a separate web application that is configured to reload JSP files, in order for changes you make to the JSP to be displayed.

How to create a custom HTML editor integration

You can use custom HTML editors in all HTML fields of the authoring interface or specific HTML elements that are defined in an authoring template. Custom HTML fields are used to integrate third-party editors into the authoring interface.

A custom HTML field is JSP that is rendered in place of the standard HTML field of the authoring interface. The custom field is composed of a number of pieces that together work to identify the field that is being modified, register the customization with the authoring interface, access standard HTML field dialogs, and update values when the time comes to commit changes. The `com.ibm.workplace.wcm.api.authoring.HTMLEditorBean`, that is available in the JSP, provides basic field metadata such as the name and whether the field is editable. The `ibm.wcm.ui.html.HTMLEditor` JavaScript library defines the client-side framework that integrates the custom field on the authoring interface. Each integration must implement its own HTMLEditor.

Storing JSP files: JSP files are stored within a web application that runs on the portal. To reference a JSP file in another web application, use the following path: `contextPath;jspPath`. For example: `/wps/customapplication;/jsp/jspFilename.jsp`.

A dynamic context path value can be defined by adding a token to the context path that corresponds to a key and value pair to the Web Content Manager configuration service environment provider. When this key is used as the token in the jsp value field, it is replaced dynamically at render time. For example: `[my.custom.key];myfile` where *my.custom.key* is a constant within the Web Content Manager configuration service.

Reloading JSP files:

JSP files that are stored in the PA_WCM_Authoring_UI application are not reloaded, even if they are updated. You must restart the server, or put the JSP in a separate web application that is configured to reload JSP files, in order for changes you make to the JSP to be displayed.

An example integration

This is a simple example of a basic integration that shows all necessary steps to render and register a custom field with the authoring interface. In this example, the text area that is used to edit HTML is extended by using the Dojo Text Area dijit.

The JSP is broken up into three distinct parts. The first is a simple code snippet to extract the name of the editor field. The next part of the JSP defines the implementation of the custom `ibm.wcm.ui.html.HTMLEditor` object. The `ibm.sample.Editor` defines a constructor to capture the dijit to manage and implement the necessary methods to integrate with the Authoring interface.

Note: It also chooses to show the HTML field toolbar by calling `this.showToolbar(true)` that starts the method on `ibm.wcm.ui.html.HTMLEditor` that controls the visibility of the toolbar.
Finally the dijit is defined, and an instance of the `ibm.sample.Editor` created.

Note: While this sample defines `ibm.sample.Editor` within the JSP, it would be more efficient to load this by using the static JavaScript library. This is important because if you have more than one editor on the page, the JavaScript definition of `ibm.sample.Editor` is loaded each time, but only the last one is loaded and used.

```
<%--
/* Sample HTML Editor */
--%>

<%@ taglib uri="/WEB-INF/tld/portlet.tld" prefix="portletAPI" %>
<%@ page import="com.ibm.workplace.wcm.app.ui.portlet.widget.HTMLEditorBean" %><portletAPI:init /><%
    HTMLEditorBean editor = (HTMLEditorBean) request.getAttribute("EditorBean");

    // The name of the editor corresponds to the ID of the text area field that will be rendered wit
    String docId = editor.getName();
%>

<script>
   dojo.declare("ibm.sample.Editor", [ibm.wcm.ui.html.HTMLEditor], {

      /** Hold a reference to the text area we enriched. */
      theTextArea: null,

      /** Simple overloaded constructor that will dynamically enrich the target text area with a dojo
      constructor: function(editorId, textArea) {
         this.inherited(arguments);
         this.theTextArea = textArea;

         textArea.startup();

         this.showToolbar(true);
      },

      /** This method is called by the HTML/RTF field dialogs. For example when the user has selecte
      insertMarkupAtCursor: function(markup){
         // Do cross browser text insertion
         if (document.selection && (!this.theTextArea.textbox.selectionStart || this.theTextArea.text
         {
            // Handle IE, which defines document.selection
            this.theTextArea.focus();
            var textRange = document.selection.createRange();
            textRange.text = markup;
         }
         else if (this.theTextArea.textbox.selectionStart || this.theTextArea.textbox.selectionStart
         {
            // Handle Mozilla, Opera, which define document.selectionStart
            // Create the new text for the text field. createRange() isn't supported by these browse
            var startPos = this.theTextArea.textbox.selectionStart;
            var endPos = this.theTextArea.textbox.selectionEnd;
            var scrollTop = this.theTextArea.textbox.scrollTop;
            this.setMarkup(this.theTextArea.value.substring(0, startPos) + markup + this.theTextArea

            // Restore the cursor and scroll position which were lost by replacing the text field's
            this.theTextArea.focus();
```

```
            }
        },

        /** This method is called when a user triggers the import markup action of the field. */
        setMarkup: function(markup){
            this.theTextArea.set("value", markup);
        },

        /** This method is called whenever the authoring interface requires the current value of the
        getMarkup: function(){
            return this.theTextArea.value;
        },

        /** This method will be called when the user submits the form to the server. This gives the
        notifySubmit: function() {
            // As we are enriching the text area being submitted, the value
            // is already up to date and we do not need to do anything
        }
    });

    require(["dijit/form/Textarea", "dojo/domReady!"], function(Textarea){
        var textArea = new Textarea({name: '<%= docId %>', style: "width:100%;max-width:910px;min-he
        new ibm.sample.Editor('<%= docId %>', textArea);
    });

</script>
```

Helper methods

The `ibm.wcm.ui.html.HTMLEditor` includes a number of helper methods to make it possible to start the standard set of HTML field dialogs programmatically. Meaning it would be possible to keep the HTML editor toolbar hidden and still provide all the functionality that it exposes.

The available methods are:

- `launchInsertImageDialog()`
- `launchInsertLinkDialog(/* String */ linkText, /* String */ linkHref, /* String */ linkTarget, /* String */ linkDescription, /* String */ linkAttributes)`
- `launchInsertTagDialog(/* boolean */ includeConsumableTags){`

How to use remote actions

Remote actions are used to trigger actions from the IBM Web Content Manager application.

You can reference remote actions by using plug-in tags by using the following format:

```
[plugin:RemoteAction action=" " docid=" "
dialog=" " dialogSize=" " dialogTitle=" " useCurrentContext=" " showInfoMsg=" " ]
```

action This is the remote action to perform.

docid This is the document ID of the item to run the remote action against.

useCurrentContext
If set to true, then the document ID is obtained form the rendering context instead of the docid attribute.

dialog If set to true, when rendered within a Web Content Viewer portlet, the

remote action is rendered as a URL that redirects the user to a hidden portal page that is used by the Web Content Viewer portlet for inline editing.

dialogSize
This optional setting defines the size of the dialog executing the remote action. The value must be in the format "width,height". For example, `dialogSize="200,300"` for a dialog of width 200 pixel and a height of 300 pixel. If omitted, the dialog size is calculated from the content that is displayed in the dialog. This setting is only used if dialog="true".

dialogTitle
This optional setting sets the title of the dialog executing the remote action. If omitted, the action name is used instead. This setting is only used if dialog="true".

showInfoMsg
Set this to true to display success status and other information messages after the remote action has finished. If omitted, this parameter is set to false and only warning and error status messages are displayed. This setting can only be used if dialog="true".

Remote actions can also be appended to the URL of an authoring portlet. For example:

```
http://[host]/wps/myportal/wcmAuthoring?wcmAuthoringAction=action
```

You can also append remote actions to the URL of a local web Content Viewer portlet. This can be useful in sites that feature inline editing of content items.

Custom authoring interfaces: Remote actions are not intended to be used to create a custom authoring interface. There are limitations to the functions delivered using remote actions. For example, remote actions only support plain text. You cannot use remote actions to add markup into elements such as HTML elements. You instead use the Web Content Manager API to create custom authoring interfaces.

Note: Each web content item can be identified by a DocumentId. The "docid" can be retrieved by using the web content API. In the following examples, the value of the "docid" parameter should be the DocumentId as retrieved by using the `DocumentID.getID()` API method. A document ID consists of a document type and a unique ID. The "docid" values that are provided in the examples are placeholders for real document IDs. For example, `com.ibm.workplace.wcm.api.WCM_Content/ID1`

Remote action types

new This is used to open a new item form. You must also specify a "type" parameter.

For example:

- `[plugin:RemoteAction action="new" type="com.ibm.workplace.wcm.api.WCM_Content" ]`

The following type parameters can be used:

- `com.ibm.workplace.wcm.api.WCM_AuthoringTemplate`
- `com.ibm.workplace.wcm.api.WCM_Category`
- `com.ibm.workplace.wcm.api.WCM_Content`
- `com.ibm.workplace.wcm.api.WCM_DateComponent`
- `com.ibm.workplace.wcm.api.WCM_FileComponent`

- com.ibm.workplace.wcm.api.WCM_HTMLComponent
- com.ibm.workplace.wcm.api.WCM_ImageComponent
- com.ibm.workplace.wcm.api.WCM_NumericComponent
- com.ibm.workplace.wcm.api.WCM_PresentationTemplate
- com.ibm.workplace.wcm.api.WCM_RichTextComponent
- com.ibm.workplace.wcm.api.WCM_ShortTextComponent
- com.ibm.workplace.wcm.api.WCM_SiteArea
- com.ibm.workplace.wcm.api.WCM_Taxonomy
- com.ibm.workplace.wcm.api.WCM_TextComponent
- com.ibm.workplace.wcm.api.WCM_Workflow
- com.ibm.workplace.wcm.api.WCM_WorkflowStage

When creating a new content item, you can specify a default authoring template by providing the document ID of the authoring template in the **atid** parameter:

- [plugin:RemoteAction action="new" type="com.ibm.workplace.wcm.api.WCM_Content" atid="com.ibm.workplace.wcm.api.WCM_AuthoringTemplate/ID1"]

When creating site areas, content items and categories, you can specify the document ID of the parent item to save the new item under. Specify this ID in the **pid** parameter:

- [plugin:RemoteAction action="new" type="com.ibm.workplace.wcm.api.WCM_Content" pid="com.ibm.workplace.wcm.api.WCM_SiteArea/ID"]
- [plugin:RemoteAction action="new" type="com.ibm.workplace.wcm.api.WCM_SiteArea" pid="com.ibm.workplace.wcm.api.WCM_SiteArea/ID"]
- [plugin:RemoteAction action="new" type="com.ibm.workplace.wcm.api.WCM_Category" pid="com.ibm.workplace.wcm.api.WCM_Taxonomy/ID"]

When creating site areas, you can specify the position of the new site area by using a **position** parameter. You can specify to save the new site area at the start or end relative to any existing site areas. If not specified, the new site area is saved at the start relative to any existing site areas:

- [plugin:RemoteAction action="new" type="com.ibm.workplace.wcm.api.WCM_Content" position="start"]
- [plugin:RemoteAction action="new" type="com.ibm.workplace.wcm.api.WCM_SiteArea" position="end"]

delete This is used to delete an item. You must also specify the **docid** of the item.

For example:

- [plugin:RemoteAction action="delete" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1"]

edit This is used to open an item form in edit mode. You must also specify the **docid** of the item.

For example:

- [plugin:RemoteAction action="edit" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1"]

read This is used to open an item form in read-only mode. You must also specify the **docid** of the item.

For example:

- `[plugin:RemoteAction action="read" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1"]`

openmainview
This is used to open a view within an authoring portlet. You must also specify a **view** parameter.

For example:

- `[plugin:RemoteAction action="openmainview" view="contentbysitearea"]`

The following view parameters can be used:

- `contentbysitearea`
- `contentbytitle`
- `myrecent`
- `mydraft`
- `mypendingapproval`
- `mypublished`
- `myexpired`
- `mydeleted`
- `alldraftitems`
- `allexpireditems`
- `allpublisheditems`
- `alldeleteditems`
- `componentsbytype`

move This is used to move a site area or content item.

For example, to open the move dialog for a content item or site area:

- `[plugin:RemoteAction action="move" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1"]`

A move direction is specified as "1" for up and "-1" for down. For example, to move a content item up one position:

- `[plugin:RemoteAction action="move" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1" pid="com.ibm.workplace.wcm.api.WCM_SiteArea/ID1" moveDirection="1"]`

link This will link a content item to a site area.

For example:

- `[plugin:RemoteAction action="link" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1" pid="com.ibm.workplace.wcm.api.WCM_SiteArea/ID2"]`

When linking items, you can specify the path to the parent item by using the **ppath** parameter instead of the **pid** parameter:

- `[plugin:RemoteAction action="link" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1" ppath="library1/sitearea1/sitearea2"]`

When linking items, you can create a new parent item by using the **autoCreateParent** parameter. You must also specify the library where the item being linked is located by using the **slibrary** parameter. The **ppath** parameter is used to specify the existing parent that the new parent item is created under:

- `[plugin:RemoteAction action="link" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1" autoCreateParent="true" slibrary="libraryname" ppath="library1/sitearea1/sitearea2"]`

copy This is used to make a copy of an item.

For example, to copy a content item to a new site area:

- `[plugin:RemoteAction action="copy" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1" pid="com.ibm.workplace.wcm.api.WCM_SiteArea/ID2"]`

You can use the following extra parameters when copying:

- `copyAsDraft="true"`

  This will restart the workflow of the copy being creating. In most cases this would result in the copy being created with a status of draft.
- `wid="com.ibm.workplace.wcm.api.WCM_Workflow/ID1"`

  Use this to specify a different workflow to use when creating the copy. This will also restart the workflow of the copy being creating. In most cases this would result in the copy being created with a status of draft.
- `position="start"`

  This will create the copy as the first item under the specified parent item. If not specified the item is copied as the last child of the specified parent item.

When copying items you can specify the path to the parent item by using the **ppath** parameter instead of the **pid** parameter:

- `[plugin:RemoteAction action="copy" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1" ppath="library1/sitearea1/sitearea2"]`

When copying items you can create a new parent item by using the **autoCreateParent** parameter. You must also specify the library where the item being copied is located by using the **slibrary** parameter. The **ppath** parameter is used to specify the existing parent that the new parent item is created under:

- `[plugin:RemoteAction action="copy" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1" autoCreateParent="true" slibrary="libraryname" ppath="library1/sitearea1/sitearea2"]`

approve
This is used to approve an item in a workflow. You must also specify the **docid** of the item.

For example:

- `[plugin:RemoteAction action="approve" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1"]`

decline
This is used to decline an item in a workflow. You must also specify the **docid** of the item.

For example:

- `[plugin:RemoteAction action="decline" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1"]`

saveandapprove
This is used to approve an item in a workflow where that item is open in edit mode within the same session. You must also specify the **docid** of the item.

For example:

- `[plugin:RemoteAction action="saveandapprove" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1"]`
- `[plugin:RemoteAction action="saveandapprove" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1" isdraft="true"]`

previousstage
This is used to move an item to the previous stage in a workflow. You must also specify the **docid** of the item.

For example:

- `[plugin:RemoteAction action="previousstage" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1"]`

viewversions
This is used to open the versions dialog for an item. You must also specify the **docid** of the item.

For example:

- `[plugin:RemoteAction action="viewversions" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1"]`

viewhistory
This is used to open the history dialog for an item. You must also specify the **docid** of the item.

For example:

- `[plugin:RemoteAction action="viewhistory" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1"]`

Populating fields when creating or editing content

When using the "new" or "edit" parameters with content items, you can also add data to different fields in the content item using a URL.

For example, to add "newcontent" as the name of the content item, you would use this URL:

- `[plugin:RemoteAction action="new" type="com.ibm.workplace.wcm.api.WCM_Content" wcmfield.content.name="newcontent"]`

The following parameters can be used to populate fields when creating or editing content:

- `wcmfield.content.name=`
- `wcmfield.content.title=`
- `wcmfield.content.description=`
- `wcmfield.content.authors=`
- `wcmfield.content.owners=`

- wcmfield.content.publishDate=
- wcmfield.content.expiryDate=
- wcmfield.content.generalDateOne=
- wcmfield.content.generalDateTwo=
- wcmfield.content.workflow= (This can only be used when creating content.)
- wcmfield.content.categories=
- wcmfield.content.keywords=
- wcmfield.element.*elementname*=

  Note: You replace *elementname* with the name of the element you are populating. The element parameter can only be used with the following element types:
  - Text
  - Html
  - Rich text
  - Option Selection
  - User Selection
  - Date and time
  - Number
  - JSP
  - Link
  - Component Reference

When populating fields with user IDs, you must use this format:

- [plugin:RemoteAction action="new"
  type="com.ibm.workplace.wcm.api.WCM_Content"
  wcmfield.content.authors="uid=*usera*,cn=*cn-name*,dc=*dc-name*"]

When populating workflow and category fields, you must use the document IDs as their values:

- [plugin:RemoteAction action="new"
  type="com.ibm.workplace.wcm.api.WCM_Content"
  wcmfield.content.workflow="ID1"]
- [plugin:RemoteAction action="new"
  type="com.ibm.workplace.wcm.api.WCM_Content"
  wcmfield.content.categories="ID1"]

When populating date fields, the date format must be US English. Either a date and time, or just a date can be specified. If only a date is specified, the time used will be 12:00:00 AM. For example:

- [plugin:RemoteAction action="new"
  type="com.ibm.workplace.wcm.api.WCM_Content"
  wcmfield.content.generalDateOne="Feb 14, 2008 12:53:03 PM"]
- [plugin:RemoteAction action="new"
  type="com.ibm.workplace.wcm.api.WCM_Content"
  wcmfield.content.generalDateOne="Feb 14, 2008"]

The date and time set here are based on the server's timezone, not the timezone of the user's computer.

When populating a JSP element, you need to specify the path to the JSP file:

- [plugin:RemoteAction action="new"
  type="com.ibm.workplace.wcm.api.WCM_Content"
  wcmfield.element.jspelementname="/wps/wcm/jsp/html/example.jsp"]

When populating a component reference element, you specify the component to reference. For example:

- [plugin:RemoteAction action="new"
  type="com.ibm.workplace.wcm.api.WCM_Content"
  wcmfield.element.mycompref.type="com.aptrix.pluto.cmpnt.NavigatorCmpnt"
  wcmfield.element.mycompref.id="e4bdf10042d0769698ccbeb0e25cc973"]

When populating an option selection element, you specify each selection option. For example:

- [plugin:RemoteAction action="new"
  type="com.ibm.workplace.wcm.api.WCM_Content"
  wcmfield.element.elementname="AA" wcmfield.element.elementname="BB"]

When populating a user selection element, you specify each user. For example:

- [plugin:RemoteAction action="new"
  type="com.ibm.workplace.wcm.api.WCM_Content"
  wcmfield.element.elementname="uid=wpsadmin,o=defaultWimFileBasedRealm"
  wcmfield.element.elementname="uid=wpsadmin2,o=defaultWimFileBasedRealm"]

When populating a Link element, you can specify the following parameters:

Adding a link to a content item:

```
[plugin:RemoteAction action="new"
type="com.ibm.workplace.wcm.api.WCM_Content"
wcmfield.element.elementname.type="content"
wcmfield.element.elementname.id="contentID"]
```

Adding a link to a link component:

```
[plugin:RemoteAction action="new"
type="com.ibm.workplace.wcm.api.WCM_Content"
wcmfield.element.elementname.type="link"
wcmfield.element.elementname.id="linkcomponentID"]
```

Adding a link to an image or file resource component:

```
[plugin:RemoteAction action="new"
type="com.ibm.workplace.wcm.api.WCM_Content"
wcmfield.element.elementname.type="resource"
wcmfield.element.elementname.id="componentID"]
```

Adding a link to a URL:

```
[plugin:RemoteAction action="new"
type="com.ibm.workplace.wcm.api.WCM_Content"
wcmfield.element.elementname.type="external"
wcmfield.element.elementname.externalReference="myurl"]
```

To specify whether to use the name of the item you are linking to as the link text, add this to the tag:

```
wcmfield.element.elementname.useReferenceLinkText="true"
```

When specifying an image to display as the link, add this to the tag:

```
wcmfield.element.elementname.linkImage="imagecomponentID"
```

When specifying the text of the link, add this to the URL:

```
wcmfield.element.elementname.linkText="text"
```

When specifying the description of the link, add this to the URL:

`wcmfield.element.elementname.linkDescription="text"`

When specifying a link target, add this to the URL:

`wcmfield.element.elementname.linkTarget=`

- `_blank`
- `_parent`
- `_self`
- `_top`
- `targetname`

Save parameters

You can add the following "save" parameters to a remote action tag.

`autoSave`

This is used to save a controllable. This happens in the background and is not displayed to users.

For example:

- `wcmfield.autosave="true"`

`saveValidate`

This parameter determines if warning and error messages resulting from the autosave is displayed to the user. If set to "true", warning and error messages are displayed to the user. If set to false, messages are suppressed. The default is true.

For example:

- `&wcmfield.saveValidate="false"`

Adding comments to the item history

When creating items that use a workflow with "Enter comment on approval" set to true, you can add a comment to the item history by adding `comment="comment text"` to the URL.

For example:

```
[plugin:RemoteAction action="edit" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1"
createDraft="true" comment="comment text"]
```

Examples

Open the versions view for an item:

- Tag: `[plugin:RemoteAction action="viewversions" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1"]`
- Url: `http://<host>/wps/myportal/wcmAuthoring?wcmAuthoringAction=viewversions &docid=com.ibm.workplace.wcm.api.WCM_Content/ID1`

Open the history view for an item:

- Tag: `[plugin:RemoteAction action="viewhistory" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1"]`
- Url: `http://<host>/wps/myportal/wcmAuthoring?wcmAuthoringAction=viewhistory &docid=com.ibm.workplace.wcm.api.WCM_Content/ID1`

Open a content item in read mode:

- Tag: `[plugin:RemoteAction action="read" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1"]`
- Url: `http://<host>/wps/myportal/wcmAuthoring?wcmAuthoringAction=read &docid=com.ibm.workplace.wcm.api.WCM_Content/ID1`

Open a content item in edit mode:

- Tag: `[plugin:RemoteAction action="edit" &docid=com.ibm.workplace.wcm.api.WCM_Content/ID1]`
- Url: `http://<host>/wps/myportal/wcmAuthoring?wcmAuthoringAction=edit &docid=com.ibm.workplace.wcm.api.WCM_Content/ID1`

Move a content item up:

- Tag: `[plugin:RemoteAction action="move" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1" moveDirection="1" pid="com.ibm.workplace.wcm.api.WCM_SiteArea/ID1"]`
- Url: `http://<host>/wps/myportal/wcmAuthoring?wcmAuthoringAction=move &docid=com.ibm.workplace.wcm.api.WCM_Content/ID1&moveDirection=1 &pid=com.ibm.workplace.wcm.api.WCM_SiteArea/ID1`

Move a site area down:

- Tag: `[plugin:RemoteAction action="move" docid="com.ibm.workplace.wcm.api.WCM_SiteArea/ID1" "moveDirection="-1" pid="com.ibm.workplace.wcm.api.WCM_SiteArea/ID1"]`
- Url: `http://<host>/wps/myportal/wcmAuthoring?wcmAuthoringAction=move &docid=com.ibm.workplace.wcm.api.WCM_SiteArea/ID1&moveDirection=-1 &pid=com.ibm.workplace.wcm.api.WCM_SiteArea/ID1`

Create a new content item with title of 'newcontent':

- Tag: `[plugin:RemoteAction action="new" type="com.ibm.workplace.wcm.api.WCM_Content" wcmfield.content.title="newcontent"]`
- Url: `http://<host>/wps/myportal/wcmAuthoring?wcmAuthoringAction=new &type=com.ibm.workplace.wcm.api.WCM_Content &wcmfield.content.title=newcontent`

To open a content item in edit mode and automatically change keywords:

- Tag: `[plugin:RemoteAction action="edit" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1" wcmfield.content.keywords="keyword1" wcmfield.content.keywords="keyword2"]`
- Url: `http://<host>/wps/myportal/wcmAuthoring?wcmAuthoringAction=edit &docid=com.ibm.workplace.wcm.api.WCM_Content/ID1 &wcmfield.content.keywords=keyword1&wcmfield.content.keywords=keyword2`

To edit a content item, automatically change the keywords and use autosave to automatically save the content (no dialog opens):

- Tag: `[plugin:RemoteAction action="edit" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1" wcmfield.content.keywords="keyword1" wcmfield.content.keywords="keyword2" wcmfield.autosave="true"]`

- Url: `http://<host>/wps/myportal/wcmAuthoring?wcmAuthoringAction=edit &docid=com.ibm.workplace.wcm.api.WCM_Content/ID1 &wcmfield.content.keywords=keyword1&wcmfield.content.keywords=keyword2 &wcmfield.autosave=true`

To edit a content item, automatically save the item and prevent any validation exception from being displayed, use autosave with saveValidate=false:

- Tag: `[plugin:RemoteAction action="edit" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1" wcmfield.content.keywords="keyword1" wcmfield.autosave="true" wcmfield.saveValidate="false"]`
- Url: `http://<host>/wps/myportal/wcmAuthoring?wcmAuthoringAction=edit &docid=com.ibm.workplace.wcm.api.WCM_Content/ID1 &wcmfield.content.keywords=keyword1&wcmfield.autosave=true &wcmfield.saveValidate=false`

To create a content item, set the name and use autosave to automatically save the content (no dialog opens). The authoring template that is used by the content item must have a workflow pre-selected:

- Tag: `[plugin:RemoteAction action="new" type="com.ibm.workplace.wcm.api.WCM_Content" atid="com.ibm.workplace.wcm.api.WCM_AuthoringTemplate/ID1" pid="com.ibm.workplace.wcm.api.WCM_SiteArea/ID2" wcmfield.content.name="newcontent" wcmfield.autosave="true" wcmfield.saveValidate="true"]`
- Url: `http://<host>/wps/myportal/wcmAuthoring?wcmAuthoringAction=new &type=com.ibm.workplace.wcm.api.WCM_Content &atid=com.ibm.workplace.wcm.api.WCM_AuthoringTemplate/ID1 &pid=com.ibm.workplace.wcm.api.WCM_SiteArea/ID2 &wcmfield.content.name=newcontent&wcmfield.autosave=true &wcmfield.saveValidate=true`

To edit a content item and create a draft on the edit and set the history log comment:

- Tag: `[plugin:RemoteAction action="edit" docid="com.ibm.workplace.wcm.api.WCM_Content/ID1" createDraft="true" comment="comment"]`
- Url: `http://<host>/wps/myportal/wcmAuthoring?wcmAuthoringAction=edit &docid=com.ibm.workplace.wcm.api.WCM_Content/ID1&createDraft=true &comment=comment`

How to create custom plug-ins

A custom plug-in is a reusable Java class that you create to run a task. You can create custom plug-ins such as custom workflow actions, plug-ins to run when a page is rendered, plug-ins to store multi-locale text strings and plug-ins to run when a file is uploaded.

"Creating a custom button class" on page 3207
You can create custom button classes that dynamically add custom actions to the authoring interface. Custom buttons are used to integrate third-party tools into the authoring interface without using a custom element. For example, you can use a custom button that adds automatic profiling of a content item or that adds or changes elements on the item forms.

"Deploying custom plug-in applications" on page 3254
You must deploy your custom plug-in applications on your server before they can be used in your web content system.

Creating a custom button class

You can create custom button classes that dynamically add custom actions to the authoring interface. Custom buttons are used to integrate third-party tools into the authoring interface without using a custom element. For example, you can use a custom button that adds automatic profiling of a content item or that adds or changes elements on the item forms.

About this task

Each extension adds an optional single button to the Read or Edit mode of an item form. When started, the extension can run any data modification on the item. Even in read mode, the extension can change the item state and run save operations. To create a custom button plug-in, you must create a custom button class and then register the class by deploying it on the server.

Procedure

1. Create a java class that implements the interface `com.ibm.workplace.wcm.api.extensions.authoring.AuthoringAction`. This class must implement the following methods:

   `public boolean isValidForForm(final FormContext formContext) {}`
   This method flags whether the button should be included on the button bar or ignored for the open item. Any uncommitted changes that are made to the document contained within the form context is discarded when this method runs. The method is started each time that the page is rendered to limit performance impacts. Use the target authoring form for *formContext*.

   `public int ordinal() {}`
   This method returns a number that determines where the button is placed relative to any other buttons created by other AuthoringAction classes. The button with the lowest number is listed first.

   See the Javadoc documentation for further information. The Javadoc files for Web Content Manager are in the *`PortalServer_root`*`/doc/Javadoc/spi_docs/com/ibm/workplace/wcm` directory.

2. Implement the `ActionResult execute(final FormContext formContext)` method, which is started when the user clicks the button.

   Any uncommitted changes that are made to the document contained within the form context is discarded when this method runs. Use the target authoring form for *formContext*.

3. A plugin.xml file is needed whether the deployment is done by using a WAR or EAR, or by using a loose jar. If deploying by using an application in a WAR or EAR, include the plugin.xml file in the application's "WEB-INF" folder. When using a jar, include the plugin.xml in the root of the jar.

```
<?xml version="1.0" encoding="UTF-8"?>
<plugin id="com.ibm.workplace.wcm.sample.authoringaction"
        name="Sample Authoring Action Extension"
        version="1.0.0"
        provider-name="IBM">
  <extension
     point="com.ibm.workplace.wcm.api.AuthoringAction
```

```
      id="SampleAuthoringAction" >
      <provider class="com.ibm.workplace.wcm.sample.MyAuthoringAction"/>
  </extension>
</plugin>
```

What to do next

- The ID of each plug-in must be unique.
- You must replace the plug-in ID specified in this example, `com.ibm.workplace.wcm.sample.authoringaction`, with a different ID for each extension you create.
- Each AuthoringAction extension is represented by a single `<extension></extension>` tag.
- The value of the point attribute must be `com.ibm.workplace.wcm.api.AuthoringAction`.
- Provide an id value of your choice.
- Specify the provider class for your AuthoringAction extension.

Naming conventions:

If you create a new plug-in application with the same names and IDs as an existing plug-in, the new plug-in can override the first. When creating plug-in applications ensure that the following are unique across your system:

- The plug-in ID, plug-in name and extension ID of the plugin.xml file.
- The fully qualified class name plus path of all classes within the application.
- The file path of any files within the application.

How to create a rendering plug-in class

A rendering plug-in is a reusable class that you create to run a task at render time. It can be referenced within web content by using a plug-in tag. For example, you might write a plug-in that uses attributes from the current user's profile to determine whether the body of the plug-in tag is rendered or not. A rendering plug-in class requires you to reference a set of web content API methods.

Creating a plug-in class

1. Create a Java class that implements the interface `com.ibm.workplace.wcm.api.plugin.rendering.RenderingPlugin` or `RenderingPluginDefinition`. This class must implement the following methods:
   - `public String getName()` .

     Note: This name is used as the "name" parameter of the plug-in tag. See Creating a plug-in tag for further information.
   - `public Boolean render(RenderingPluginModel p_model)` throws `RenderingPluginException`.
2. Implement `render()method`. This method contains the code that is run when the plug-in is started during rendering of a layout that contains a "plug-in" tag that references the custom plug-in. Returning true renders the body markup that is defined in the plug-in tag. If false is returned, the body of the plug-in tag is skipped. If the plug-in tag has no body markup, then the return value is ignored.
3. Methods that are inherited from *com.ibm.portal.Localized* must also be implemented.

public String getTitle(Locale displayLocale) {}
This method returns the title for the rendering plug-in that is used to allow selection of the rendering plug-in.

public ListModel<Locale> getLocales()
This method returns a list of locales that are supported by this rendering plug-in.

public String getDescription(Locale p_arg0)
This method returns a description of the rendering plug-in.

See the Javadoc documentation for further information. The Javadoc files for Web Content Manager are in the *PortalServer_root*/doc/Javadoc/spi_docs/com/ibm/workplace/wcm directory.

4. The rendering plug-in can optionally implement `RenderingPluginDefinition` instead of `RenderingPlugin` to define its type and parameters. This is recommended, because it allows the rendering plug-ins to be used more easily.

   To do this, implement the methods:

   - `public RenderingPluginType getType()`.
   - `public List<RenderingPluginParameter>getParameters()`.

   Note: When creating a custom rendering plug-in:

   - Do not use "name" as a parameter in the rendering plug-in, because the name of the plug-in is stored as the "name" parameter.
   - If you use "id" as a rendering parameter, it will not be displayed in read mode.

For example:

```
package test;

import java.io.IOException;
import java.io.Writer;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
import java.util.Locale;
import java.util.Map;
import java.util.Set;

import com.ibm.portal.ListModel;
import com.ibm.portal.ModelException;
import com.ibm.workplace.wcm.api.plugin.rendering.RenderingPluginDefinition;
import com.ibm.workplace.wcm.api.plugin.rendering.RenderingPluginException;
import com.ibm.workplace.wcm.api.plugin.rendering.RenderingPluginModel;
import com.ibm.workplace.wcm.api.plugin.rendering.RenderingPluginParameter;
import com.ibm.workplace.wcm.api.plugin.rendering.RenderingPluginParameterAdapter;
import com.ibm.workplace.wcm.api.plugin.rendering.RenderingPluginParameterImpl;
import com.ibm.workplace.wcm.api.plugin.rendering.RenderingPluginType;
import com.ibm.workplace.wcm.api.plugin.rendering.RenderingPluginTypes;
import com.ibm.workplace.wcm.api.plugin.rendering.RenderingPluginParameter.Required;
import com.ibm.workplace.wcm.api.plugin.rendering.ValueOptionImpl;

/**
 * A simple rendering plugin to demonstrate the use of the <code>RenderingPlugin</code> API.
 */
public class SimpleRenderingPlugin implements RenderingPluginDefinition
{
   /** The 'render body' parameter */
   private static final String RENDER_BODY_PARAM = "renderbody";

   /**
    * A simple list model holding locales.
```

```
 */
protected static class SimpleLocaleListModel<K> implements ListModel<Locale>
{
   /** the list of locales of this list model */
   final List<Locale> m_localeList = new ArrayList<Locale>();

   /**
    * Constructs this simple list model holding the given locales.
    *
    * @param p_locales
    *           the locales of this list model. May be <code>null</code>.
    */
   public SimpleLocaleListModel(final Locale[] p_locales)
   {
      if (p_locales != null)
      {
         for (int i = 0; i < p_locales.length; ++i)
         {
            m_localeList.add(p_locales[i]);
         }
      }
   }

   @Override
   public Iterator<Locale> iterator() throws ModelException
   {
      return m_localeList.iterator();
   }
}

/** a list model that only contains the English language locale */
private static final ListModel<Locale> ENGLISH_ONLY = new SimpleLocaleListModel<Locale>(new Locale

@Override
public String getDescription(final Locale p_locale)
{
   return "This is a simple rendering plugin.";
}

@Override
public ListModel<Locale> getLocales()
{
   return ENGLISH_ONLY;
}

@Override
public String getName()
{
   return "SimpleRenderingPlugin";
}

@Override
public String getTitle(final Locale p_locale)
{
   return "SimpleRenderingPlugin";
}

@Override
public boolean isShownInAuthoringUI()
{
   return false;
}

@Override
public boolean render(final RenderingPluginModel p_model) throws RenderingPluginException
{
   final Map<String, List<String>> params = p_model.getPluginParameters();
```

```
      // determine whether the inner contents of the plugin should actually be rendered
      final boolean renderBody;
      final List<String> renderBodyList = params.get(RENDER_BODY_PARAM);
      if (renderBodyList != null && renderBodyList.get(0).equals("false"))
      {
         renderBody = false;
      }
      else
      {
         renderBody = true;
      }

      // render the output of the plugin to the writer provided by the RenderingPluginModel
      final Writer writer = p_model.getWriter();
      try
      {
         writer.write("<b>Simple RenderingPlugin</b>");

         final Set<String> keys = params.keySet();

         final Iterator<String> iter = keys.iterator();
         while (iter.hasNext())
         {
            String key = iter.next();
            writer.write("<br>" + key + " = " + params.get(key));
         }

         writer.write("<br><br>");
      }
      catch (IOException e)
      {
         e.printStackTrace();
      }

      return renderBody;
   }

   @Override
   public RenderingPluginType getType()
   {
      return RenderingPluginTypes.CONTENT;
   }

   @Override
   public List<RenderingPluginParameter> getParameters()
   {
      List<RenderingPluginParameter> parameters = new ArrayList<RenderingPluginParameter>();
      RenderingPluginParameterAdapter renderBodyParam = new RenderingPluginParameterImpl(RENDER_BO
      parameters.add(renderBodyParam);
      return parameters;
   }
}
```

Create a `plugin.xml` file

A `plugin.xml` file is needed whether the deployment is done by using a WAR or EAR, or by using a loose jar. If deploying an application in a WAR or EAR, include the `plugin.xml` file in the application's `WEB-INF`folder. When using a jar, include the `plugin.xml` in the root of the jar.

```
<?xml version="1.0" encoding="UTF-8"?>
<?eclipse version="3.0"?>
<plugin
   id="Test"
   name="Simple Rendering Plug-in Test"
```

```
    version="1.0.0"
    provider-name="IBM">

    <extension
      point="com.ibm.workplace.wcm.api.RenderingPlugin"
      id="SimpleRenderingPlugin">
      <provider class="test.SimpleRenderingPlugin"/>
    </extension>
</plugin>
```

- Each plug-in is represented by a single `<extension></extension>` tag.
- The value of the point attribute must be `com.ibm.workplace.wcm.api.RenderingPlugin`.
- Provide an ID value of your choice.
- Specify the provider class for your plug-in.

Naming conventions:

If you create a plug-in application with the same names and IDs as an existing plug-in, the new plug-in will not be registered. When creating plug-in applications ensure that the following are unique across your system:

- The plug-in ID, plug-in name, and extension ID of the `plugin.xml` file.
- The fully qualified class name plus path of all classes within the application.
- The file path of any files within the application.

How to create a custom workflow action class

You can create custom workflow action classes to add custom workflow actions to a workflow.

Creating the custom workflow action class

1. Create a java class that implements the interface `com.ibm.workplace.wcm.api.custom.CustomWorkflowAction`. This class must implement the following methods:
   - `public Date getExecuteDate(Document p_document) {}` (This method specifies when the custom action is run.)
   - `public CustomWorkflowActionResult execute(Document p_document) {}` (This method contains the code that runs when the custom action is run.)
2. Implement `execute()method`. This method contains the code that is run against the supplied Document. This method must return a `com.ibm.workplace.wcm.api.custom.CustomWorkflowActionResult` object to indicate the result of the custom code by using `com.ibm.workplace.wcm.api.custom.Directives`.
   - A custom workflow action result object is created by first retrieving a reference to the `WebContentCustomWorkflowService` object, and then calling the method `webContentCustomWorkflowService.getCustomWorkflowService().createResult`. If the `CustomWorkflowActionResult` does not indicate a failure, changes to the document is saved.

     See the Javadoc documentation for further information. The Javadoc files for Web Content Manager are in the *`PortalServer_root`*`/doc/Javadoc/spi_docs/com/ibm/workplace/wcm` directory.
   - Also, see the Web Content Manager Javadoc for further information on valid directives.

3. Create a custom workflow action factory class that implements the interface `com.ibm.workplace.wcm.api.custom.CustomWorkflowActionFactory`.

Create a `plugin.xml` file

A `plugin.xml` file is needed whether the deployment is done by using a WAR or EAR, or by using a loose jar. If you deploy using an application in a WAR or EAR, include the `plugin.xml` file in the application's `WEB-INF` folder. When you use a jar, include the `plugin.xml` in the root of the jar.

```
<?xml version="1.0" encoding="UTF-8"?>
<plugin id="com.ibm.workplace.wcm.sample.customworkflowaction"
name="Sample Custom Workflow Action Factory"
version="1.0.0"
provider-name="IBM">

<extension
point="com.ibm.workplace.wcm.api.CustomWorkflowActionFactory"
id="SimpleCustomWorkflowActionFactory">
<provider class="com.ibm.workplace.wcm.sample.customworkflowaction.SimpleCustomWorkflowActionFacto
</extension>

</plugin>
```

- The ID of each plug-in must be unique. You must replace the plug-in ID specified in this example, *com.ibm.workplace.wcm.sample.customworkflowaction*, with a different ID for each custom workflow you create.
- Each custom workflow action factory is represented by a single `<extension></extension>` tag.
- The value of the point attribute must be `com.ibm.workplace.wcm.api.CustomWorkflowActionFactory`.
- Provide an ID value of your choice.
- Specify the provider class for your custom workflow action factory.

Naming conventions:

If you create a new plug-in application with the same names and IDs as an existing plug-in, the new plug-in can override the first. When you create plug-in applications ensure that the following are unique across your system:

- The plug-in ID, plug-in name, and extension ID of the `plugin.xml` file.
- The fully qualified class name plus path of all classes within the application.
- The file path of any files within the application.

Creating a Text Provider class

A text provider is used to provide localized text that can be used within web content item forms. For example, a text provider can be used to localize the field labels or help text within an authoring template so that each user sees the labels or help text in their own language.

About this task

To use a text provider, you must create a text provider class and then register the text provider by deploying it on the server.

Procedure

1. Create a java class that implements the interface *com.ibm.workplace.wcm.api.plugin.textprovider.TextProvider*. This class must implement the following methods:

   public String getProviderName()
   This method returns the unique name of the text provider.

   public String getString(String key, Locale displayLocale)
   This method returns some translated text, given a key identifying the message, and a locale.

   public Collection<String> getProviderKeys()
   This method returns a list of keys that are used when accessing the text provider. These keys are displayed in the authoring UI when a user is configuring the text provider.

   public boolean isShownInAuthoringUI()
   This method is used to prevent your text provider from appearing in the authoring UI.

   See the Javadoc documentation for further information. The Javadoc files for Web Content Manager are in the *PortalServer_root*/doc/Javadoc/spi_docs/com/ibm/workplace/wcm directory.

2. Methods that are inherited from *com.ibm.portal.Localized* must also be implemented.

   public String getTitle(Locale displayLocale)
   This method returns the title for the text provider that is used to allow selection of the text provider.

   public ListModel<Locale> getLocales()
   This method returns a list of locales that are supported by this text provider.

   public String getDescription(Locale p_arg0)
   This method returns a description of the text provider.

   For example:

```
package test;

import java.io.IOException;
import java.util.ArrayList;
import java.util.Collection;
import java.util.Iterator;
import java.util.LinkedHashSet;
import java.util.List;
import java.util.Locale;
import java.util.MissingResourceException;
import java.util.Properties;
import java.util.ResourceBundle;
import java.util.Set;

import com.ibm.portal.ListModel;
import com.ibm.portal.ModelException;
import com.ibm.workplace.wcm.api.plugin.textprovider.TextProvider;

/**
 * A simple text provider implementation
 */
public class SimpleTextProvider implements TextProvider
{
   /** Path to Bundle properties file (English used for loading keyset) */
   private static final String BUNDLE_PROPERTIES = "/test/SimpleBundle_en.properties";
```

```
/** The text provider title */
public static final String PROVIDER_TITLE = "Simple Text Provider";

/** The unique text provider name */
public static final String PROVIDER_NAME = "test.SimpleTextProvider";

/** A brief description of this text provider */
private static final String PROVIDER_DESCRIPTION = "Simple Text provider";

/** The resource bundle used to lookup strings */
private static final String RESOURCE_BUNDLE_NAME = "test.SimpleBundle";

/**
 * A simple list model holding locales.
 */
protected static class SimpleLocaleListModel<K> implements ListModel<Locale>
{
   /** the list of locales of this list model */
   final List<Locale> m_localeList = new ArrayList<Locale>();

   /**
    * Constructs this simple list model holding the given locales.
    *
    * @param p_locales
    *           the locales of this list model. May be <code>null</code>.
    */
   public SimpleLocaleListModel(final Locale[] p_locales)
   {
      if (p_locales != null)
      {
         for (int i = 0; i < p_locales.length; ++i)
         {
            m_localeList.add(p_locales[i]);
         }
      }
   }

   @Override
   public Iterator<Locale> iterator() throws ModelException
   {
      return m_localeList.iterator();
   }
}

/** a list model that only contains the English language locale */
private static final ListModel<Locale> ENGLISH_ONLY = new SimpleLocaleListModel<Locale>(new

@Override
public String getProviderName()
{
   return PROVIDER_NAME;
}

@Override
public String getTitle(Locale p_displayLocale)
{
   // Perform any localization for the plug-in title here
   return PROVIDER_TITLE;
}

@Override
public String getDescription(Locale p_displayLocale)
{
   return PROVIDER_DESCRIPTION;
}
```

```
@Override
public String getString(String p_key, Locale p_displayLocale)
{
   String value;

   try
   {
      ResourceBundle bundle = ResourceBundle.getBundle(RESOURCE_BUNDLE_NAME, p_displayLocale);
      value = bundle.getString(p_key);
   }
   catch (MissingResourceException e)
   {
      // The bundle or key was not found. Return null.
      value = null;
   }
   return value;
}

@Override
public Collection<String> getProviderKeys()
{
   Collection<String> keys = null;
   try
   {
      LinkedProperties props = new LinkedProperties();
      props.load(getClass().getClassLoader().getResourceAsStream(BUNDLE_PROPERTIES));

      keys = props.getKeySet();
   }
   catch (IOException e)
   {
      // The bundle was not found. Return null.
      e.printStackTrace();
   }

   return keys;
}


@Override
public ListModel<Locale> getLocales()
{
   return ENGLISH_ONLY;
}

@Override
public boolean isShownInAuthoringUI()
{
   return true;
}

/** Used to provide the properties in order */
private class LinkedProperties extends Properties {

   /** Keys */
   private final LinkedHashSet<String> keys = new LinkedHashSet<String>();

   /**
    * @return An ordered set of keys
    */
   public Set<String> getKeySet()
   {
      return keys;
   }

   @Override
   public Object put(Object key, Object value) {
```

```
            keys.add((String) key);
            return super.put(key, value);
        }
    }
}
```

See the Javadoc documentation for further information.

3. A `plugin.xml` file is needed whether the deployment is done using a WAR or EAR, or using a loose jar. If deploying via an application in a WAR or EAR, include the `plugin.xml` file in the application's `WEB-INF` folder. When using a jar, include the `plugin.xml` in the root of the jar.

```
<?xml version="1.0" encoding="UTF-8"?>
<plugin id="com.acme"
       name="Sample Text Provider"
       version="1.0.0"
       provider-name="IBM">

 <extension
    point="com.ibm.workplace.wcm.api.TextProvider"
    id="SampleTextProvider">
    <provider class="com.acme.SampleTextprovider"/>
 </extension>

</plugin>
```

Extend the Text ProviderAdapter: If you are creating a text provider that uses a resource bundle to provide the translated strings, your text provider class can extend `com.ibm.workplace.wcm.api.plugin.textprovider.TextProviderAdapter` instead of implementing `TextProvider`. If you choose to do this, the `getString()` and `getProviderKeys()` method are implemented for you, so you do not have to override them.

```
package com.ibm.workplace.wcm.ctc;

import java.util.Locale;

import java.io.IOException;
import java.util.ArrayList;
import java.util.Collection;
import java.util.Iterator;
import java.util.LinkedHashSet;
import java.util.List;
import java.util.MissingResourceException;
import java.util.Properties;
import java.util.ResourceBundle;
import java.util.Set;

import com.ibm.portal.ListModel;
import com.ibm.portal.ModelException;
import com.ibm.workplace.wcm.api.plugin.textprovider.TextProviderAdapter;

/**
 * A simple text provider implementation that extends TextProviderAdapter.
 */
public class SimpleTextProvider extends TextProviderAdapter
{
   /** The resource bundle used to lookup strings */
   private static final String RESOURCE_BUNDLE_NAME = "test.SimpleBundle";

   /** The text provider title */
   public static final String PROVIDER_TITLE = "Simple Text Provider";

   /** A brief description of this text provider */
   private static final String PROVIDER_DESCRIPTION = "Simple Text provider";
```

```
/** The unique text provider name */
public static final String PROVIDER_NAME = "test.SimpleTextProvider";

/**
 * A simple list model holding locales.
 */
protected static class SimpleLocaleListModel<K> implements ListModel<Locale>
{
   /** the list of locales of this list model */
   final List<Locale> m_localeList = new ArrayList<Locale>();

   /**
    * Constructs this simple list model holding the given locales.
    *
    * @param p_locales
    *           the locales of this list model. May be <code>null</code>.
    */
   public SimpleLocaleListModel(final Locale[] p_locales)
   {
      if (p_locales != null)
      {
         for (int i = 0; i < p_locales.length; ++i)
         {
            m_localeList.add(p_locales[i]);
         }
      }
   }

   @Override
   public Iterator<Locale> iterator() throws ModelException
   {
      return m_localeList.iterator();
   }
}

/** a list model that only contains the English language locale */
private static final ListModel<Locale> ENGLISH_ONLY = new SimpleLocaleListModel<Locale>(new Lo

@Override
public String getProviderName()
{
   return PROVIDER_NAME;
}

@Override
public String getTitle(Locale p_displayLocale)
{
   // Perform any localization for the plug-in title here
   return PROVIDER_TITLE;
}

@Override
public String getDescription(Locale p_displayLocale)
{
   return PROVIDER_DESCRIPTION;
}

@Override
public boolean isShownInAuthoringUI()
{
   return true;
}

@Override
protected String getResourceBundleName()
{
   return RESOURCE_BUNDLE_NAME;
```

```
    }

    @Override
    public ListModel<Locale> getLocales()
    {
       return ENGLISH_ONLY;
    }
}
```

What to do next

- Each plug-in is represented by a single `<extension></extension>` tag.
- The value of the point attribute must be `com.ibm.workplace.wcm.api.TextProvider`.
- Provide an id value of your choice.
- Specify the provider class for your plug-in.

Naming conventions:

If you create a new plug-in application with the same names and IDs as an existing plugin, the new plug-in may override the first. When creating plug-in applications ensure that the following are unique across your system:

- The plug-in ID, plug-in name and extension ID of the `plugin.xml` file.
- The fully qualified class name plus path of all classes within the application.
- The file path of any files within the application.

Sorting conventions:

When sorting is applied to a set of items, the item title is used to sort the items, not the title specified in the text provider.

Related information:

Content plug-ins

Creating a Text Provider Factory class

A text provider factory is used to provide multiple text providers that can be used within web content item forms. Text provider can be used to localize the field labels or help text within an authoring template so that each user sees the labels or help text in their own language. A text provider factory can make multiple such text providers available.

About this task

To use a text provider factory, you must create a text provider factory class and then register the text provider factory by deploying it on the server.

Procedure

1. Create a java class that implements the interface *com.ibm.workplace.wcm.api.plugin.textprovider.TextProviderFactory*. This class must implement the following methods:

   `public String getProviderName()`
   This method returns the unique name of the text provider factory.

   `public List<TextProvider> getTextProviders()`
   This method returns a list of text providers supplied by this factory.

public Collection<String> getTextProvider(String p_providerName)
This method returns the text providers in the factory with the given name.

public boolean isShownInAuthoringUI()
This method is used to prevent the text providers in your text provider factory from appearing in the authoring UI.

See the Javadoc documentation for further information. The Javadoc files for Web Content Manager are in the *PortalServer_root*/doc/Javadoc/spi_docs/com/ibm/workplace/wcm directory.

2. A plugin.xml file is needed whether the deployment is done using a WAR or EAR, or using a loose jar. If deploying via an application in a WAR or EAR, include the plugin.xml file in the application's WEB-INF folder. When using a jar, include the plugin.xml in the root of the jar.

```
<?xml version="1.0" encoding="UTF-8"?>
<plugin id="com.acme"
       name="Sample text provider factory"
       version="1.0.0"
       provider-name="IBM">

 <extension
    point="com.ibm.workplace.wcm.api.TextProviderFactory"
    id="SampleTextProviderFactory">
    <provider class="com.acme.SampleTextproviderFactory"/>
 </extension>

</plugin>
```

What to do next

- Each plug-in is represented by a single <extension></extension> tag.
- The value of the point attribute must be com.ibm.workplace.wcm.api.TextProvider.
- Provide an id value of your choice.
- Specify the provider class for your plug-in.

Naming conventions:

If you create a new plug-in application with the same names and IDs as an existing plugin, the new plug-in may override the first. When creating plug-in applications ensure that the following are unique across your system:

- The plug-in ID, plug-in name and extension ID of the plugin.xml file.
- The fully qualified class name plus path of all classes within the application.
- The file path of any files within the application.

Sorting conventions:

When sorting is applied to a set of items, the item title is used to sort the items, not the title specified in the text provider factory.

Creating a file upload validation class

A file upload validation plug-in is started anytime a file is uploaded into Web Content Manager. This includes uploading files into file resource, image and stylesheet elements, and images that are uploaded into rich text or HTML elements. The plug-in is called within the "validation" processing that is used by Web Content Manager when uploading files.

About this task

To create a file upload validation plug-in, you must create a file upload validation class and then register the file upload validation class by deploying it on the server.

Procedure

1. Create a java class that implements the interface `com.ibm.workplace.wcm.api.extensions.validation.FileUploadValidationPlugin`. This class must implement the following methods:

   `public String getName()`
   This method returns the unique name of the file upload validation plug-in.

   `public boolean validate(InputStream p_inptStream, FileUploadValidationContext p_context)`
   This method throws the FileUploadValidationException.

   See the Javadoc documentation for further information. The Javadoc files for Web Content Manager are in the *`PortalServer_root`*`/doc/Javadoc/spi_docs/com/ibm/workplace/wcm` directory.

2. Implement `validate()method`. This method contains the code that will be run when the plug-in is invoked when uploading a file. If validated, the file continues to upload. If not validated then the file upload is stopped. You can display a message in the user interface by including the following code in validate method:

   ```
   throw new FileUploadValidationException( your own message );
   ```

   For example:

   ```
   package pers.smp.extension.test.validation;

   import java.io.InputStream;
   import java.util.logging.Logger;

   import com.ibm.workplace.wcm.api.extensions.validation.FileUploadValidationContext;
   import com.ibm.workplace.wcm.api.extensions.validation.FileUploadValidationException;
   import com.ibm.workplace.wcm.api.extensions.validation.FileUploadValidationPlugin;
   import com.ibm.workplace.wcm.services.validation.FileUploadValidationContextImpl;

   public class SMPValidation1 implements FileUploadValidationPlugin
   {
      private final long MAX_SIZE_IMAGES = 512 * 1024;
      private final long MAX_SIZE_FILES = 1024 * 1024;

      private static Logger s_log = Logger.getLogger(SMPValidation1.class.getName());

      public String getName()
      {
         return "SMPValidation1";
      }

      public boolean validate(InputStream p_inptStream, FileUploadValidationContext p_context)
   throws FileUploadValidationException
      {
         s_log.info("File Name : " + p_context.getFileName() );
         s_log.info("File Type : " + p_context.getMimeType() );
         s_log.info("File Size : " + p_context.getFileSize() );
         s_log.info("Document Type : " + p_context.getDocumentType() );

         boolean valid = true;
         String message = null;
   ```

```
      String mimeType = p_context.getMimeType();

      if ( mimeType != null && mimeType.startsWith( "image/" ) )
      {
         if ( ! (mimeType.equalsIgnoreCase( "image/gif") ||
mimeType.equalsIgnoreCase( "image/jpeg") ) )
         {
            throw new FileUploadValidationException( "Invalid image type : " + mimeType +
" will only accept GIF and JPG images" );
         }
         if ( p_context.getFileSize() > MAX_SIZE_IMAGES )
         {
            throw new FileUploadValidationException( "Image is too big 500K is maximum
size allowed for images.  Size is
" +  p_context.getFileSize());
         }
      }
      else
      {
         if ( p_context.getFileSize() > MAX_SIZE_FILES )
         {
            throw new FileUploadValidationException( "File is too big 1M is maximum
size allowed for
" + mimeType + ".  Size is " +  p_context.getFileSize());
         }

      }

      return valid;
   }
}
```

3. A plugin.xml file is needed whether the deployment is done using a WAR or EAR, or using a loose jar. If deploying via an application in a WAR or EAR, include the plugin.xml file in the application's "WEB-INF" folder. When using a jar, include the plugin.xml in the root of the jar.

```
<?xml version="1.0" encoding="UTF-8"?>
<plugin id="pers_smp_extension_test"
        name="SMP Test Extensions"
        version="1.0.0"
        provider-name="IBM">

  <extension
     point="com.ibm.workplace.wcm.api.FileUploadValidationPlugin" id="SMPValidation1">
     <provider class="pers.smp.extension.test.validation.SMPValidation1"/>
  </extension>

</plugin>
```

What to do next

- Each plug-in is represented by a single `<extension></extension>` tag.
- The value of the point attribute must be `com.ibm.workplace.wcm.api.FileUploadValidationPlugin`.
- Provide an ID value of your choice.
- Specify the provider class for your plug-in.

Naming conventions:

If you create a new plug-in application with the same names and IDs as an existing plug-in, the new plug-in can override the first. When creating plug-in applications ensure that the following are unique across your system:

- The plug-in ID, plug-in name and extension ID of the `plugin.xml` file.

- The fully qualified class name plus path of all classes within the application.
- The file path of any files within the application.

Creating an item validation plug-in class

An item validation plug-in is used to validate a TemplatedDocument prior to it being committed to the repository. It is invoked as part of the standard item validation step. For example, saves occurring in the authoring interface, public API or REST API.

Possibly usages of this extension point include:

- Validating cross field relationships
- Complex custom field validation
- Workflow validation for an item based on type and location in the repository

Creating the item validation plug-in class

1. Create a Java class that implements the interface `com.ibm.workplace.wcm.api.ItemValidationPlugin` . This class must implement the following methods:
   - `DocumentId<? extends AbstractAuthoringTemplate>[]validationScope()`
   - `ItemValidationResult validate(ItemContext context)`

   Note: The validate method returns a *ItemValidationResult* that reports back any errors to be shown to the current user.
2. Within the validate method any field of the item can be modified. After all validation plug-ins are started the standard item and template validation is performed. See the Web Content Manager Javadoc for further information.

   For example:

```
package com.sample;

import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
import java.util.Locale;

import com.ibm.portal.ListModel;
import com.ibm.portal.Localized;
import com.ibm.portal.ModelException;
import com.ibm.workplace.wcm.api.AbstractAuthoringTemplate;
import com.ibm.workplace.wcm.api.AuthoringTemplate;
import com.ibm.workplace.wcm.api.Content;
import com.ibm.workplace.wcm.api.Document;
import com.ibm.workplace.wcm.api.DocumentId;
import com.ibm.workplace.wcm.api.DocumentLibrary;
import com.ibm.workplace.wcm.api.DocumentTypes;
import com.ibm.workplace.wcm.api.RichTextComponent;
import com.ibm.workplace.wcm.api.WCM_API;
import com.ibm.workplace.wcm.api.Workspace;
import com.ibm.workplace.wcm.api.exceptions.ComponentNotFoundException;
import com.ibm.workplace.wcm.api.exceptions.OperationFailedException;
import com.ibm.workplace.wcm.api.exceptions.ServiceNotAvailableException;
import com.ibm.workplace.wcm.api.extensions.validation.ItemContext;
import com.ibm.workplace.wcm.api.ItemValidationPlugin;
import com.ibm.workplace.wcm.api.extensions.validation.ItemValidationResult;

/**
 * A simple item validation plug-in
 */
public class SampleItemValidator implements ItemValidationPlugin
```

```
{

   /**
    * A simple list model holding locales.
    */
   protected static class SimpleLocaleListModel<K> implements ListModel<Locale>
   {
      /** the list of locales of this list model */
      final List<Locale> m_localeList = new ArrayList<Locale>();

      /**
       * Constructs this simple list model holding the given locales.
       *
       * @param locales
       *           the locales of this list model. May be <code>null</code>.
       */
      public SimpleLocaleListModel(final Locale[] p_locales)
      {
         if (p_locales != null)
         {
            for (int i = 0; i < p_locales.length; ++i)
            {
               m_localeList.add(p_locales[i]);
            }
         }
      }

      /*
       * (non-Javadoc)
       * @see com.ibm.portal.ListModel#iterator()
       */
      @Override
      public Iterator<Locale> iterator() throws ModelException
      {
         return m_localeList.iterator();
      }
   }

   /** a list model that only contains the English language locale */
   private static final ListModel<Locale> ENGLISH_ONLY = new SimpleLocaleListModel<Locale>(new Lo

   /*
    * (non-Javadoc)
    *
    * @see com.ibm.portal.Localized#getDescription(java.util.Locale)
    */
   @Override
   public String getDescription(final Locale p_locale)
   {
      return "This is a simple item validation plugin.";
   }

   /*
    * (non-Javadoc)
    * @see com.ibm.portal.Localized#getLocales()
    */
   @Override
   public ListModel<Locale> getLocales()
   {
      return ENGLISH_ONLY;
   }

   /*
    * (non-Javadoc)
    * @see com.ibm.portal.Localized#getTitle(java.util.Locale)
    */
   @Override
```

```
public String getTitle(final Locale p_locale)
{
   return "SimpleItemValidationPlugin";
}

/**
 * @see com.ibm.workplace.wcm.api.ItemValidationPlugin#validate(com.ibm.workplace.wcm.api.e
 */
@Override
public ItemValidationResult validate(ItemContext p_itemContext)
{
   boolean isValidItem = true;
   String error = null;
   try
   {
      // Perform a very simple validation. Check for the Body element, and return an error
      // found in the field.
      Document document = p_itemContext.document();
      if (document instanceof Content)
      {
         Content content = (Content) document;

         RichTextComponent body = (RichTextComponent)content.getComponent("Body");

         if (body == null)
         {
            error = "Item validation failed: Content has no body element";
            isValidItem = false;
         }
         else
         {
            if (body.getRichText().contains("BAD BAD BAD"))
            {
               error = "Item validation failed: BAD BAD BAD message found in Body field.";
               isValidItem = false;
            }
         }
      }
   }
   catch (IllegalStateException e)
   {
      throw e;
   }
   catch (ComponentNotFoundException e)
   {
      error = "Item validation failed: Content has no body element";
      isValidItem = false;
   }

   final boolean isValidResponse = isValidItem;
   final String errorMessageResponse = error;

   /**
    * @see com.ibm.workplace.wcm.api.extensions.validation.ItemValidationResult
    *
    * A very simple ItemValidationResult that only returns errors.
    */
   return new ItemValidationResult()
   {
      @Override
      public Localized[] errorMessages()
      {
         return new Localized[]{
            new Localized()
            {
               public String getTitle(Locale locale)
               {
```

```
                    return errorMessageResponse;
                }

                public ListModel<Locale> getLocales()
                {
                    return ENGLISH_ONLY;
                }

                public String getDescription(Locale locale)
                {
                    return null;
                }
            }
        };
    }

    @Override
    public boolean isValid()
    {
        return isValidResponse;
    }

    @Override
    public Localized[] infoMessages()
    {
        return null;
    }

    @Override
    public Localized[] warningMessages()
    {
        return null;
    }

    @Override
    public Localized[] successMessages()
    {
        return null;
    }
};
}

/**
 * @see com.ibm.workplace.wcm.api.ItemValidationPlugin#validationScope()
 */
public DocumentId<? extends AbstractAuthoringTemplate>[] validationScope()
{
    try
    {
        // Find the Article authoring template in the sample out of the box library.
        Workspace workspace = WCM_API.getRepository().getSystemWorkspace();
        DocumentLibrary docLib = workspace.getDocumentLibrary("Web Content");
        workspace.setCurrentDocumentLibrary(docLib);
        DocumentId<AuthoringTemplate> templateId = workspace.findByName(DocumentTypes.AuthoringT



        // This plug-in will be invoked whenever a content item based on this template is saved.
        return new DocumentId[]{templateId};
    }
    catch (ServiceNotAvailableException e)
    {
        e.printStackTrace();
    }
    catch (OperationFailedException e)
    {
        e.printStackTrace();
```

```
        }
        return null;
    }

    @Override
    public boolean isShownInAuthoringUI()
    {
        return false;
    }
}
```

Create a `plugin.xml` file

A `plugin.xml` file is needed whether the deployment is done by using a WAR or EAR, or by using a loose jar. If you deploy using an application in a WAR or EAR, include the `plugin.xml` file in the application's `WEB-INF` folder. When you use a jar, include the `plugin.xml` in the root of the jar.

```
<?xml version="1.0" encoding="UTF-8"?>

<plug-in
   id="Test"
   name="Simple Item Validation Plug-in"
   version="1.0.0"
   provider-name="IBM">

   <extension
     point="com.ibm.workplace.wcm.api.ItemValidationPlugin"
     id="SimpleItemValidationPlugin">
     <provider class="com.sample.SampleItemValidator"/>
   </extension>
</plug-in>
```

- Each plug-in is represented by a single `<extension></extension>` tag.
- The value of the point attribute must be `com.ibm.workplace.wcm.api.ItemValidationPlugin`.
- Provide an ID value of your choice.
- Specify the provider class for your plug-in.

Naming conventions:

If you create a new plug-in application with the same names and IDs as an existing plug-in, the new plug-in can override the first. When you create plug-in applications, ensure that the following are unique across your system:

- The plug-in ID, plug-in name, and extension ID of the `plugin.xml` file.
- The fully qualified class name plus path of all classes within the application.
- The file path of any files within the application.

Creating a subscriber class

A subscriber plug-in is used to run extra functions on the subscriber that can be used to determine if the subscriber is ready for syndication when a syndication event is started.

About this task

To create a subscriber plug-in, you must create a subscriber class and then register the subscriber class by deploying it on the server.

Procedure

1. Create a java class that implements the interface com.ibm.workplace.wcm.api.extensions.syndication.SubscriberReady. This class must implement the following methods:

   public ResultDirective onSubscriberReady(SubscriberEvent eventInfo)

   - This method contains the code that is run when the syndication run starts.
   - This method is run on the subscriber.
   - The extensions are run only when there are changes in the syndicator.
   - The extension is not run every time automatic syndication queues the syndicator.

   See the Javadoc documentation for further information. The Javadoc files for Web Content Manager are in the *PortalServer_root*/doc/Javadoc/spi_docs/com/ibm/workplace/wcm directory.
2. Implement the onSubscriberReady method. This method must return a com.ibm.workplace.wcm.api.extensions.syndication.ResultDirective object to indicate whether the syndication engine can continue or stop the syndication process.
3. A plugin.xml file is needed whether the deployment is done by using a WAR or EAR, or by using a loose jar. If deploying using an application in a WAR or EAR, include the plugin.xml file in the application's WEB-INF folder. When using a jar, include the plugin.xml in the root of the jar.

```
<?xml version="1.0" encoding="UTF-8"?>
<plugin id="com.ibm.workplace.wcm.sample.subscriberready"
        name="Sample Subscriber Ready Extension"
        version="1.0.0"
        provider-name="IBM">
  <extension
     point="com.ibm.workplace.wcm.api.SubscriberReady"
     id="SubscriberReadyExtension" >
     <provider class="com.ibm.workplace.wcm.sample.subscriberready.SubscriberReadyExtension"/>
  </extension>
</plugin>
```

What to do next

- The ID of each plug-in must be unique.
- You must replace the plug-in ID specified in this example, com.ibm.workplace.wcm.sample.subscriberready, with a different ID for each SubscriberReady extension you create.
- Each SubscriberReady extension is represented by a single <extension></extension> tag.
- The value of the point attribute must be com.ibm.workplace.wcm.api.SubscriberReady.
- Provide an ID value of your choice.
- Specify the provider class for your SubscriberReady extension.

Naming conventions:

If you create a new plug-in application with the same names and IDs as an existing plug-in, the new plug-in might override the first. When you create plug-in applications ensure that the following are unique across your system:

- The plug-in ID, plug-in name and extension ID of the plugin.xml file.

- The fully qualified class name plus path of all classes within the application.
- The file path of any files within the application.

Creating a syndicator class

A syndicator plug-in is used to run extra functions on the syndicator when a syndication event is started.

About this task

To create a syndicator plug-in, you must create a syndicator class and then register the syndicator class by deploying it on the server.

Procedure

1. Create a java class that implements the interface `com.ibm.workplace.wcm.api.extensions.syndication.SyndicatorStarted`. This class must implement the following methods:

   `public ResultDirective onSyndicatorStarted(SyndicatorEvent eventInfo)`

   - This method contains the code that is run when the syndication run starts.
   - This method is run after the plug-ins for the `SubscriberReady` extension point are run.
   - The extensions are run only when there are changes on the syndicator.
   - The extension is not run every time automatic syndication queues the syndicator.

   See the Javadoc documentation for further information. The Javadoc files for Web Content Manager are in the *PortalServer_root*`/doc/Javadoc/spi_docs/com/ibm/workplace/wcm` directory.
2. Implement the `onSyndicatorStarted` method.
   - This method contains the code that is run on the syndicator when there are changes available for syndication to the subscriber.
   - This method must return a `com.ibm.workplace.wcm.api.extensions.syndication.ResultDirective` object to indicate whether the syndication engine can continue or stop the syndication process.
3. A `plugin.xml` file is needed whether the deployment is done by using a WAR or EAR, or by using a loose jar. If deployed by using an application in a WAR or EAR, include the `plugin.xml` file in the application's `WEB-INF` folder. When you use a jar, include the `plugin.xml` in the root of the jar.

```
<?xml version="1.0" encoding="UTF-8"?>
<plugin id="com.ibm.workplace.wcm.sample.syndicatorstarted"
        name="Sample Syndicator Started Extension"
        version="1.0.0"
        provider-name="IBM">
  <extension
     point="com.ibm.workplace.wcm.api.SyndicatorStarted"
     id="SyndicatorStartedExtension" >
     <provider class="com.ibm.workplace.wcm.sample.syndicatorstarted.SyndicatorStartedExtensio
  </extension>
</plugin>
```

What to do next

- The ID of each plug-in must be unique.

- You must replace the plug-in ID specified in this example, com.ibm.workplace.wcm.sample.syndicatorstarted, with a different ID for each SyndicatorStarted extension you create.
- Each SyndicatorStarted extension is represented by a single <extension></extension> tag.
- The value of the point attribute must be com.ibm.workplace.wcm.api.SyndicatorStarted.
- Provide an ID value of your choice.
- Specify the provider class for your SyndicatorStarted extension.

Naming conventions:

If you create a new plug-in application with the same names and IDs as an existing plug-in, the new plug-in might override the first. When you create plug-in applications, ensure that the following are unique across your system:

- The plug-in ID, plug-in name, and extension ID of the plugin.xml file.
- The fully qualified class name plus path of all classes within the application.
- The file path of any files within the application.

Creating a context processor class

When configured, a context processor plug-in is started by the web content viewer portlet before rendering and allows the current context, such as the item to display, to be modified.

About this task

To create a context processor plug-in, you must create a context processor class and then register the context processor class by deploying it on the server and selecting it from within a web content viewer portlet.

Procedure

1. Create a Java class that implements the interface com.ibm.workplace.wcm.api.ContextProcessor. This class must implement the following method:

```
/**
    * Processes the supplied <i>ContextProcessorParams</i> and updates parameters within
    * as necessary
    *
    * @param p_currentSession The current Http Session
    * @param p_contextProcessorParams The editable <i>ContextProcessorParams</i> object
    */
   public void process(HttpSession p_currentSession, ContextProcessorParams p_contextProcessorPar
```

   See the Javadoc documentation for further information. The Javadoc files for Web Content Manager are in the *PortalServer_root*/doc/Javadoc/spi_docs/com/ibm/workplace/wcm directory.
2. Implement the process method. This method contains the code that is run when the plug-in is started and modifies the current context by using the ContextProcessorParams object before the current context is rendered.
3. A plugin.xml file is needed whether the deployment is done by using a WAR or EAR, or by using a loose JAR file. If deployed by using an application in a WAR or EAR, include the plugin.xml file in the application "WEB-INF" folder. When a JAR file is used, include the plugin.xml in the root of the JAR file.

```
<?xml version="1.0" encoding="UTF-8"?>
<plugin id="SampleContextProcessorPluginId"
name="SampleContextProcessor" provider-name="IBM" version="1.0.0">
    <extension point="com.ibm.workplace.wcm.api.ContextProcessor"
    id="SampleContextProcessorPlugin" >
        <processor class="com.acme.SampleContextProcessor"/>
    </extension>
</plugin>
```

4. Edit the settings of the web content viewer portlet you want to associate with your context processor:
   a. Go to the "configuration" or "edit shared settings" view of a web content viewer portlet.
   b. Go to **Advanced Options** > **Plugins**
   c. Select a context processor plug-in.

What to do next

- Each plug-in is represented by a single `<extension></extension>` tag.
- The value of the point attribute must be "`com.ibm.workplace.wcm.api.ContextProcessor`".
- Provide an ID value of your choice.
- Specify the provider class for your plug-in.

Naming conventions:

If you create a plug-in application with the same names and IDs as an existing plug-in, the new plug-in might override the first. When you create plug-in applications, ensure that the following are unique across your system:

- The plug-in ID, plug in name, and extension ID of the `plugin.xml` file.
- The fully qualified class name plus path of all classes within the application.
- The file path of any files within the application.

Creating a content page resolution filter class

A content page resolution filter is used to customize the behavior of the content page resolution filter chain. This method is used to tailor the response to a web content request in several ways, including overriding the content item that is displayed or the portal page that is used to display a content item in the web content viewer.

About this task

The content page resolution filter chain is composed of filters that are based on the Intercepting Filter design pattern, which provides a mechanism for intercepting a request and manipulating the request and its response. When used with requests for web content, the content page resolution filter chain has default filters that process any URL parameters that are contained in the web content request and then determine which portal page has a matching web content association. The default filters occur at the end of the filter chain.

You can customize the content page resolution filter chain by creating custom filters that are registered with the portal through the Eclipse plug-in framework with the extension ID `com.ibm.workplace.wcm.api.ContentPageResolutionFilter`. The sequence of filters in the filter chain is specified by a weight value associated with each filter. To insert custom filters into the filter chain before the default

filters, you can use the `weight` attribute in the `plugin.xml` file. If the `weight` attribute is not present, filter sequence is determined by the `getFilterChainWeight` method of each custom filter.

Custom filters can perform various actions:

- Modify parameters before calling the default filters.
- Modify the result of the default filters.
- Handle exceptions that are generated by the default filters.
- Determine whether the default filters are started.
- Modify the content path that is used as input for the default filters.
- Explicitly set a target page for displaying content.
- Determine which web content page are used, if the default filters find more than one matching web content page for the request.
- Modify the presentation template selection.
- Set HTTP response status codes.
- Send redirects to external web resources.

To use a content page resolution filter, you must create a content page resolution filter class and then register the filter by deploying it on the server.

Procedure

1. Create a java class that implements the interface *com.ibm.workplace.wcm.api.extensions.resolution.ContentPageResolutionFilter*. This class must implement the following methods:

   `public int getFilterChainWeight() {}`
   This method returns the weight that is applied to the content page resolution filter elements in the filter chain. The less it is weighted, the earlier the filter is inserted into the chain.If the `weight` parameter is defined in the `plugin.xml` file, that value overrides the value that is returned by this method.

   `public void resolve(ContentPageResolutionRequest request, ContentPageResolutionResponse response, ContentPageResolutionFilterChain chain) {}`
   This method is started during `ContentPageResolution` processing. The `response` parameter is used to modify the content item that is displayed, the portal page where the content is displayed, and the presentation template that is used to render the content item. The request extends the resolver interface with an extra method that gets the content item that is addressed. The filter chain contains the subsequent filters that can be started if needed.

   See the Javadoc documentation for further information. The Javadoc files for Web Content Manager are in the *`PortalServer_root`*`/doc/Javadoc/spi_docs/com/ibm/workplace/wcm` directory.

2. A `plugin.xml` file is needed whether the deployment is done by using a WAR or EAR, or by using a loose jar. If deploying with an application in a WAR or EAR, include the `plugin.xml` file in the application's "WEB-INF" folder. When using a jar, include the `plugin.xml` in the root of the jar.

   ```
   <?xml version="1.0" encoding="UTF-8"?>
   <plugin id="com.example"
          name="Sample Content Page Resolution Filter"
          version="1.0.0"
          provider-name="IBM">
   ```

```
 <extension
    point="com.ibm.workplace.wcm.api.ContentPageResolutionFilter"
    id="SampleContentPageResolutionFilter">
    <provider class="com.example.SampleContentPageResolutionFilter"
              weight="1"/>
 </extension>

</plugin>
```

When creating plug-ins, note the following:

- Each plug-in is represented by a single `<extension>` tag.
- The value of the extension point attribute must be `com.ibm.workplace.wcm.api.ContentPageResolutionFilter`.
- Provide an ID value of your choice.
- Specify the filter class for your plug-in.
- The `weight` parameter overrides the value of the `getFilterChainWeight` method.

Naming conventions:

If you create a new plug-in application with the same names and IDs as an existing plug-in, the new plug-in can override the first. When creating plug-in applications ensure that the following are unique across your system:

- The plug-in ID, plug-in name, and extension ID of the `plugin.xml` file.
- The fully qualified class name and path of all classes within the application.
- The file path of any files within the application.

Examples

- The `LocaleDependantSelectionFilter` example performs the default page resolution and selects a target page from the candidate pages, according to the user's locale.

```
package com.ibm.workplace.wcm.extension.resolution;

import java.util.List;
import java.util.Locale;

import javax.naming.InitialContext;
import javax.naming.NamingException;

import com.ibm.portal.ObjectID;
import com.ibm.portal.model.CorLocalizedContextHome;
import com.ibm.portal.model.LocalizedContext;
import com.ibm.workplace.wcm.api.extensions.resolution.ContentPageResolutionFilter;
import com.ibm.workplace.wcm.api.extensions.resolution.ContentPageResolutionFilterChain;
import com.ibm.workplace.wcm.api.extensions.resolution.ContentPageResolutionRequest;
import com.ibm.workplace.wcm.api.extensions.resolution.ContentPageResolutionResponse;
import com.ibm.workplace.wcm.api.extensions.resolution.exceptions.ContentPageResolutionException;

public class LocaleDependantSelectionFilter implements ContentPageResolutionFilter
{
   CorLocalizedContextHome localizedContextHome;

   public LocaleDependantSelectionFilter()
   {
      try
      {
         InitialContext ctx = new InitialContext();
         localizedContextHome = (CorLocalizedContextHome) ctx.lookup(CorLocalizedContextHome.JNDI_NAME);
      }
      catch(NamingException e)
      {
         e.printStackTrace();
      }
   }

   public int getFilterChainWeight()
   {
      return 2;
   }
```

```
   public void resolve(ContentPageResolutionRequest request, ContentPageResolutionResponse response,
ContentPageResolutionFilterChain chain)
         throws ContentPageResolutionException
   {
      // do standard resolution first
      chain.resolve(request, response);

      // check if more than one candidate pages
      List<ObjectID> candidates = response.getCandidatePageIds();
      if (candidates.size() > 1)
      {
         LocalizedContext context = localizedContextHome.getLocalizedContext(request.getContext());
         Locale locale = context.getPreferredSupportedLocale();
         String lang = locale.getLanguage();
         // find page with matching unique name
         for (ObjectID pageId : candidates)
         {
            if (pageId.getUniqueName().endsWith("." + lang))
            {
               response.setPageID(pageId);
               break;
            }
         }
      }
   }
}
```

- The `ChangeContentPathFilter` example overrides the path of the content that is rendered but the page resolution is performed on the originally requested content.

```
package com.ibm.workplace.wcm.extension.resolution;

import com.ibm.workplace.wcm.api.extensions.resolution.ContentPageResolutionFilter;
import com.ibm.workplace.wcm.api.extensions.resolution.ContentPageResolutionFilterChain;
import com.ibm.workplace.wcm.api.extensions.resolution.ContentPageResolutionRequest;
import com.ibm.workplace.wcm.api.extensions.resolution.ContentPageResolutionResponse;
import com.ibm.workplace.wcm.api.extensions.resolution.exceptions.ContentPageResolutionException;

public class ChangeContentPathFilter implements ContentPageResolutionFilter
{
   public int getFilterChainWeight()
   {
      return 3;
   }

   public void resolve(ContentPageResolutionRequest request, ContentPageResolutionResponse response,
ContentPageResolutionFilterChain chain)
         throws ContentPageResolutionException
   {
      // resolve page using requested content
      chain.resolve(request, response);

      // but instead of england render UK
      if ("countries/world/europe/england".equals(response.getContentPath()))
      {
         response.setContentPath("countries/world/europe/uk");
      }
   }
}
```

- The `ResolveToSpecificPageFilter` example resolves to a specific page. No default resolution is performed in this case.

```
package com.ibm.workplace.wcm.extension.resolution;
import java.util.Arrays;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import com.ibm.portal.ObjectID;
import com.ibm.portal.identification.Identification;
import com.ibm.portal.serialize.SerializationException;
import com.ibm.workplace.wcm.api.extensions.resolution.ContentPageResolutionFilter;
import com.ibm.workplace.wcm.api.extensions.resolution.ContentPageResolutionFilterChain;
import com.ibm.workplace.wcm.api.extensions.resolution.ContentPageResolutionRequest;
import com.ibm.workplace.wcm.api.extensions.resolution.ContentPageResolutionResponse;
import com.ibm.workplace.wcm.api.extensions.resolution.exceptions.ContentPageResolutionException;

public class ResolveToSpecificPageFilter implements ContentPageResolutionFilter
{
   private Identification identification;

   public ResolveToSpecificPageFilter()
   {
      try
      {
          InitialContext ctx = new InitialContext();
          identification = (Identification) ctx.lookup("portal:service/Identification");
      }
      catch (NamingException nx)
      {
          nx.printStackTrace();
      }
   }

   public int getFilterChainWeight()
   {
      return 4;
      }
```

```
 public void resolve(ContentPageResolutionRequest request, ContentPageResolutionResponse response, ContentPageResolutionFilterChain chain) throws ContentPageResolutionException

 {

      try
      {
      // always resolve to page with unique name my.default.page
      ObjectID pageId = identification.deserialize("my.default.page");
      response.setPageID(pageId);
      response.setCandidatePageIds(Arrays.asList(new ObjectID[]{pageId}));
      }
      catch (SerializationException e)
      {
          throw new ContentPageResolutionException(e);
      }
  }
}
```

- The `SetResponseCodePageResolutionFilter` example checks for the requested web content item. If it does not exist, the code throws an exception that results in sending a 404 (Not found) HTTP response status code. If the content exists, resolution is delegated to other filters in the chain.

```
package com.ibm.workplace.wcm.extension.resolution;

import java.util.Arrays;
import java.util.HashSet;
import java.util.Iterator;
import java.util.Locale;
import java.util.Set;
import javax.servlet.http.HttpServletResponse;
import com.ibm.portal.ListModel;
import com.ibm.portal.LocalizedStatus;
import com.ibm.portal.ModelException;
import com.ibm.workplace.wcm.api.extensions.resolution.ContentPageResolutionFilter;
import com.ibm.workplace.wcm.api.extensions.resolution.ContentPageResolutionFilterChain;
import com.ibm.workplace.wcm.api.extensions.resolution.ContentPageResolutionRequest;
import com.ibm.workplace.wcm.api.extensions.resolution.ContentPageResolutionResponse;
import com.ibm.workplace.wcm.api.extensions.resolution.ResolvedItem;
import com.ibm.workplace.wcm.api.extensions.resolution.exceptions.ContentPageResolutionException;

public class SetResponseCodePageResolutionFilter implements ContentPageResolutionFilter

{
  public void resolve(ContentPageResolutionRequest request, ContentPageResolutionResponse response, ContentPageResolutionFilterChain chain) throws ContentPageResolutionException
  {
      if (!itemExists(request.getItem()))
      {
          // send 404
          throw new ContentPageResolutionException(new ContentNotFoundException());
      }
     else
     {
         // forward to the chain if the web content exists
         chain.resolve(request, response);
     }
 }

 private boolean itemExists(ResolvedItem item)
 {
        return (item.getItemID() != null) && (item.getItemPath() != null);
 }
 public int getFilterChainWeight()
 {
         return 1;
 }

 private static class ContentNotFoundException extends Exception implements LocalizedStatus
 {
        private static final long serialVersionUID = 70L;
        private static final Set<Locale> SUPPORTED_LOCALES = new HashSet<Locale>(Arrays.asList(new Locale[] { Locale.ENGLISH }));
   private static final String MESSAGE = "The requested web content does not exist";

    public ContentNotFoundException()
     {
              super(MESSAGE);
     }

     public int getStatus()
     {
             return HttpServletResponse.SC_NOT_FOUND;
     }
      public String getTitle(Locale locale)
     {
              return MESSAGE;
      }
      public String getDescription(Locale locale)
      {
              return MESSAGE;
      }
      public ListModel<Locale> getLocales()
      {
              return new ListModel<Locale>()
              {
                      public Iterator<Locale> iterator() throws ModelException
                      {
                              return SUPPORTED_LOCALES.iterator();
                      }
              };
     }
   }
  }
```

- The `SendRedirectPageResolutionFilter` example checks for the requested web content item. If the content does not exist, the code sends a redirect to `http://www.ibm.com`. If the content exists, the resolution is delegated to other filters in the chain.

```
package com.ibm.workplace.wcm.extension.resolution;

import java.io.UnsupportedEncodingException;
import java.net.URI;
import java.net.URISyntaxException;
import java.net.URLEncoder;
import com.ibm.portal.resolver.exceptions.ResolutionException;
```

```
import com.ibm.portal.resolver.helper.CORResolutionService;
import com.ibm.portal.state.exceptions.StateException;
import com.ibm.workplace.wcm.api.extensions.resolution.ContentPageResolutionFilter;
import com.ibm.workplace.wcm.api.extensions.resolution.ContentPageResolutionFilterChain;
import com.ibm.workplace.wcm.api.extensions.resolution.ContentPageResolutionRequest;
import com.ibm.workplace.wcm.api.extensions.resolution.ContentPageResolutionResponse;
import com.ibm.workplace.wcm.api.extensions.resolution.ResolvedItem;
import com.ibm.workplace.wcm.api.extensions.resolution.exceptions.ContentPageResolutionException;

public class SendRedirectPageResolutionFilter implements ContentPageResolutionFilter
{
 // the URL to redirect to
 private static final String URL = "http://www.ibm.com";

 public void resolve(ContentPageResolutionRequest request, ContentPageResolutionResponse response, ContentPageResolutionFilterChain chain) throws ContentPageResolutionException
 {
  if (!itemExists(request.getItem()))
  {
   try
   {
    // encode to URL, this is important to prevent that the ':'
    // character appears in the path
    String encodedURL = URLEncoder.encode(URL, "UTF-8");
    final URI redirectURI = new URI(com.ibm.portal.resolver.Constants.SCHEME_REDIRECT, encodedURL, null);
    CORResolutionService.SINGLETON.resolve(request.getResolved(), redirectURI, request.getVerb(), request.getResolutionParameters(), request.getAcceptedBindings(),request.getContext());
   } catch (UnsupportedEncodingException uenc)
   {
    // should never happens as long as UTF-8 is supported
    throw new ContentPageResolutionException(uenc);
   } catch (URISyntaxException e)
   {
    throw new ContentPageResolutionException(e);
   } catch (StateException e)
   {
    throw new ContentPageResolutionException(e);
   } catch (ResolutionException e)
   {
    // do not catch the resolution exception
    // as this is used internally to trigger the redirect
    throw new ContentPageResolutionException(e);
   }
  } else
  {
   chain.resolve(request, response);
  }
 }

 private boolean itemExists(ResolvedItem item)
 {
  return (item.getItemID() != null) && (item.getItemPath() != null);
 }

 public int getFilterChainWeight()
 {
  return 1;
 }
}
```

- The following example provides the `plugin.xml` file that registers the previous sample filters.

  Registering all of these filters in one system is not recommended. The filters perform overlapping operations and are also exclusive in some cases. To use one of the filters in your system, remove the other unused filters in the `plugin.xml` file before deploying the file.

```
<?xml version="1.0" encoding="UTF-8"?>
<?eclipse version="3.0"?>
<plugin
   id="com.ibm.workplace.wcm.extension.resolution"
   name="Content Page Resolution Filter"
   version="1.0.0"
   provider-name="IBM">

   <extension point="com.ibm.workplace.wcm.api.ContentPageResolutionFilter"
       id="SendRedirectPageResolutionFilter">
       <provider class="com.ibm.workplace.wcm.extension.resolution.SendRedirectPageResolutionFilter"
                 weight="1"/>
   </extension>

   <extension point="com.ibm.workplace.wcm.api.ContentPageResolutionFilter"
        id="SetResponseCodePageResolutionFilter">
        <provider class="com.ibm.workplace.wcm.extension.resolution.SetResponseCodePageResolutionFilter"
                 weight="1"/>
   </extension>

   <extension point="com.ibm.workplace.wcm.api.ContentPageResolutionFilter"
      id="LocaleDependantSelectionFilter">
      <provider class="com.ibm.workplace.wcm.extension.resolution.LocaleDependantSelectionFilter"
                 weight="2"/>
    </extension>

   <extension point="com.ibm.workplace.wcm.api.ContentPageResolutionFilter"
      id="ChangeContentPathFilter">
      <provider class="com.ibm.workplace.wcm.extension.resolution.ChangeContentPathFilter"
                 weight="3"/>
    </extension>

   <extension point="com.ibm.workplace.wcm.api.ContentPageResolutionFilter"
      id="ResolveToSpecificPageFilter">
      <provider class="com.ibm.workplace.wcm.extension.resolution.ResolveToSpecificPageFilter"
                 weight="4"/>
    </extension>
</plugin>
```

Related tasks:

 Intercepting Filter design pattern

Creating a content URL generation filter class

A content URL generation filter is used to customize the URLs that are generated by a web content viewer. By creating a plug-in that implements a content URL generation filter, you can tailor the URLs to content items.

About this task

The web content viewer generates a content URL whenever there is a URL to web content within content that the viewer is displaying.

The content URL generation filter chain is a chain of filters that are based on the Intercepting Filter design pattern. This design pattern provides a mechanism for intercepting a request and manipulating the request and its response. When used with requests for content URL generation, the content URL generation filter chain has default filters that run the following actions:

- Process any request to generate a URL that target a web content item.
- Create a URL based on the configuration of the web content viewer and any parameters set on the URL generation tags, such as the `URLCmpnt` tag.

The default filters occur at the end of the filter chain.

You can customize the content URL generation filter chain by creating custom filters. These filters are registered with the portal through the Eclipse plug-in framework with the extension ID `com.ibm.workplace.wcm.api.ContentUrlGenerationFilter`. The sequence of filters in the filter chain is specified by a weight value that is associated with each filter plug-in. To insert custom filters into the filter chain before the default filters, you can use the `weight` attribute in the `plugin.xml` file. If the `weight` attribute is not present, filter sequence is determined by the `getFilterChainWeight` method of each custom filter factory.

Custom content URL generation filters can run various actions:

- Modify parameters before you call the default URL generation filters.
- Modify the URL that is generated by the default filters.
- Handle exceptions that are generated by the default filters.
- Determine whether the default filters are started.
- Modify the content path that is used as input for the default URL generation filters.
- Generate any type of URL without using the default URL generation filters.

Important: You can use custom content URL generation filters to modify only URLs that are generated by web content tags, such as the `Placeholder` or the `URLCmpnt` tag. The Web Content Viewer does not call custom content URL generation filters when it generates, for example, the URLs of a page navigation component.

To use a content URL generation filter, you must create two classes:

- A content URL generation filter that is used to create URLs.
- A content URL generation filter factory that is used to create new instances of the filter.

You must deploy both classes on the server and register the filter factory by using a `plugin.xml` file.

Procedure

1. Create a Java class that implements the interface *com.ibm.workplace.wcm.api.extensions.url.ContentUrlGenerationFilterFactory*. This class must implement the following methods:

   `public ContentUrlGenerationFilter getFilter(RenderRequest portletRequest, RenderResponse portletResponse) throws ContentUrlFilterInstantiationException`
   This method is called by the content URL generation chain once for each content item that is rendered. The method creates an instance of a content URL generation filter, which generates all URLs that appear in the rendered web content.

   `public int getFilterChainWeight() {}`
   This method returns the weight that is applied to the content URL generation filter elements in the filter chain. The less it is weighted, the earlier the filter is inserted into the chain. If the `weight` parameter is defined in the `plugin.xml` file, that value overrides the value that is returned by this method.

   See the Javadoc documentation for further information. The Javadoc files for Web Content Manager are in the *`PortalServer_root`*`/doc/Javadoc/spi_docs/com/ibm/workplace/wcm` directory.

2. Create a Java class that implements the interface *com.ibm.workplace.wcm.api.extensions.url.ContentUrlGenerationFilter*. This class must implement the following methods:

   `public void writeURL(ContentUrlGenerationRequest request, ContentUrlGenerationResponse response, ContentUrlGenerationFilterChain chain) throws ContentUrlGenerationException, IOException`
   This method is started during content URL generation processing and is started once per content URL. The response parameter is used to write to the URL. The request parameter contains the following information:
   - Information about the item for which the URL is generated.
   - Any web content viewer configuration and related information that might affect the generation of the URL.

   The filter chain contains the subsequent filters that can be started when needed.

   `public void dispose()`
   This method is started by the filter chain after all URLs that appear in the rendered content items are written. The method enables filters to run a cleanup of their internal state.

   For more information about this class, see the Javadoc documentation.

3. A `plugin.xml` file is needed whether the deployment is done by using a WAR or EAR file, or by using a JAR file. If you deploy with an application in a WAR or EAR file, include the `plugin.xml` file in the `WEB-INF` folder. When you use a JAR file, include the `plugin.xml` in the root of the JAR file.

   ```
   <?xml version="1.0" encoding="UTF-8"?>
   <plugin id="com.example.content.url"
          name="Sample content URL generation filter"
          version="1.0.0"
          provider-name="IBM">
   ```

```
 <extension
    point="com.ibm.workplace.wcm.api.ContentUrlGenerationFilter"
    id="SampleContentUrlGenerationFilter">
    <factory class="com.example.SampleContentUrlGenerationFilterFactory"
             weight="1"/>
 </extension>

</plugin>
```

When you create plug-ins, note the following characteristics:

- Each plug-in is represented by a single `<extension>` tag.
- The value of the extension point attribute must be `com.ibm.workplace.wcm.api.ContentUrlGenerationFilter`.
- Provide an ID value of your choice.
- Specify the filter factory class for your plug-in.
- The `weight` parameter overrides the value of the `getFilterChainWeight` method.

Naming conventions:

If you create a new plug-in application with the same names and IDs as an existing plug-in, the new plug-in might override the first. When you create plug-in applications ensure that the following elements are unique across your system:

- The plug-in ID, plug-in name, and extension ID of the `plugin.xml` file.
- The fully qualified class name and path of all classes within the application.
- The file path of any files within the application.

"Example 1: Append a prefix to a content URL"
This example demonstrates a content URL generation filter that appends a prefix to each content URL that is written. This type of content URL generation filter is useful when used with an HTTP server that dynamically rewrites incoming URLs.

"Example 2: Generate a friendly URL for web content" on page 3240
This example demonstrates a content URL generation filter that generates a friendly URL for web content.

Related concepts:

"Friendly URLs and Web Content Viewers" on page 2040
Friendly URLs provide a way for you to define a custom address for a portal page that is easy to remember and share. The Web Content Viewer expands on friendly URL support so you can specify extra path information in the friendly URL.

Example 1: Append a prefix to a content URL

This example demonstrates a content URL generation filter that appends a prefix to each content URL that is written. This type of content URL generation filter is useful when used with an HTTP server that dynamically rewrites incoming URLs.

Factory class

```
/*****************************************************************
 * Copyright IBM Corp. 2011                                      *
 *****************************************************************/
package com.ibm.workplace.wcm.api.samples;

import javax.portlet.*;
import com.ibm.workplace.wcm.api.extensions.url.*;

public class RewriteUrlGenerationFilterFactory implements ContentUrlGenerationFilterFactory {
 @Override
 public ContentUrlGenerationFilter getFilter(RenderRequest portletRequest, RenderResponse portletResponse)
```

```
            throws ContentUrlFilterInstantiationException {
  return new RewriteUrlGenerationFilter();
 }

 @Override
 public int getFilterChainWeight() {
  return 5;
 }
}
```

Filter class

```
************************************
 * Copyright IBM Corp. 2011                                    *
 ***************************************************************/
package com.ibm.workplace.wcm.api.samples;

import java.io.*;
import com.ibm.workplace.wcm.api.extensions.url.*;

public class RewriteUrlGenerationFilter implements ContentUrlGenerationFilter {

 /** that static prefix that is prepended to all URLs */
 private static final String PREFIX = "/content/";

 @Override
 public void dispose() {
  // no cleanup required for this filter
 }

 @Override
 public void writeURL(ContentUrlGenerationRequest request, ContentUrlGenerationResponse response,
                           ContentUrlGenerationFilterChain chain) throws ContentUrlGenerationException,
                           IOException {

  final String contentPath = request.getContentPath(true);
  if (contentPath != null && !contentPath.isEmpty()) {
   // write the prefix
   final Writer out = response.getWriter();
   out.write(PREFIX);

   // write path to content
   if (contentPath.charAt(0) == '/') {
    // Omit a leading / to avoid 2 / characters
    out.write(contentPath, 1, contentPath.length() - 1);
   } else {
    out.write(contentPath);
   }
  } else {
   // let the other filters handle prefix URLs
   chain.writeURL(request, response);
  }
 }
}
```

plugin.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<?eclipse version="3.0"?>
<plugin id="com.ibm.workplace.wcm.api.samples.plugin"
        name="My content URL plugin"
        version="1.0.0" provider-name="IBM">
    <extension point="com.ibm.workplace.wcm.api.ContentUrlGenerationFilter" id="MyRewriteUrlGenerationFilter">
        <factory class="com.ibm.workplace.wcm.api.samples.RewriteUrlGenerationFilterFactory"
                 weight="5"/>
    </extension>
</plugin>
```

Example 2: Generate a friendly URL for web content

This example demonstrates a content URL generation filter that generates a friendly URL for web content.

The filter determines the target portal page that is used to render the web content. If there is a web content page that has a content association that references a parent site area of the content, the filter creates a friendly URL to the content.

Friendly URLs for web content are constructed according to several factors. For example, the friendly URL `/wps/myportal/home/news/news+1` is determined by the following details:

- Content path: `/Library1/News/News 1`

- Content association on the web content page where the content is rendered: `/Library1/News` site area
- Friendly URL of this page: `/home/news`

To use the sample filter that is described here, each page for which a content URL is generated must have these characteristics:

- The page must be a web content page with a friendly name.
- The page must have a default content association that references the parent site area of the content item.

The friendly URLs that are generated by this sample filter do not contain any portlet state information. When a user clicks one of these friendly URLs from within a web content viewer that does maintain state information, the state information in the viewer is cleared.

Important:

- These sample filter classes are not provided by default with IBM Web Content Manager. Example code is provided here that you can take and use to implement the classes yourself.
- Some web content URLs strictly require state information, for example the URLs that are generated by page navigation components. Therefore, web content URLs that are not generated by web content tags, such as the `Placeholder` or the `URLCmpnt` tag, are not processed by custom content URL generation filters.

  "Example 2: Filter factory class"
  This sample demonstrates a filter factory class. This filter class creates an instance of the filter for each request to render web content that contains URLs to other web content items.

  "Example 2: Filter class" on page 3245
  This sample filter generates the friendly URL.

  "Example 2: `plugin.xml` file" on page 3254
  You can use this sample `plugin.xml` file to register the sample filter.

Related concepts:

"Friendly URLs and Web Content Viewers" on page 2040
Friendly URLs provide a way for you to define a custom address for a portal page that is easy to remember and share. The Web Content Viewer expands on friendly URL support so you can specify extra path information in the friendly URL.

Related tasks:

"Using friendly URLs without state information" on page 1281
By default, WebSphere Portal Express URLs include navigational state information. If you configure pages for friendly URLs, the portal appends the state information to the friendly URLs. Some scenarios require short and fully human readable URLs that omit the state information. For such scenarios, you can configure friendly URLs so that the portal does not show that state information.

Example 2: Filter factory class:

This sample demonstrates a filter factory class. This filter class creates an instance of the filter for each request to render web content that contains URLs to other web content items.

The factory also obtains references to portal services and models that are required by the filter. These references are required to look up web content pages according to these criteria:

- The path of the web content for which the URL is generated.
- The configuration of the web content viewer.

By default, this sample filter is enabled after you deploy the sample classes. However, you can disable the sample filter by setting the following portlet preference in the web content viewer configuration:

- Preference: **DISABLE_FriendlyUrlGenerationFilter**
- Value: `true`

```
/*****************************************************************
 * Copyright IBM Corp. 2011                                      *
 *****************************************************************/
package com.ibm.workplace.wcm.api.samples;
import java.util.logging.*;
import javax.naming.*;
import javax.portlet.*;
import com.ibm.portal.*;
import com.ibm.portal.identification.*;
import com.ibm.portal.portlet.service.*;
import com.ibm.portal.portlet.service.model.*;
import com.ibm.portal.resolver.friendly.service.*;
import com.ibm.portal.services.contentmapping.*;
import com.ibm.portal.state.exceptions.*;
import com.ibm.portal.state.service.*;
import com.ibm.workplace.wcm.api.*;
import com.ibm.workplace.wcm.api.exceptions.*;
import com.ibm.workplace.wcm.api.extensions.url.*;

/**
 * Factory for the friendly content URL generation filter.
 *
 * The filter factory is used to create a new instance of the filter for each
 * request to render a web content that contains URLs to other web content
 * items. This sample filter can be disabled by setting a portlet preference on
 * the Web Content Viewer. To disable the filter set a preference with the name
 * {@value #DISABLE_FILTER} and the value "true". By default the filter will be
 * enabled after the deployment of the sample classes.
 *
 * Furthermore the factory is used to obtain references to different portal
 * services and models that are required by the filter to do the lookup of web
 * content pages based on the path of the web content the URL is generated for
 * and the configuration of the Web Content Viewer portlet.
 */
public class FriendlyUrlGenerationFilterFactory implements ContentUrlGenerationFilterFactory {

 /**
  * name of the portlet preference that can be used to check if the filter is
  * disabled
  */
 public final static String DISABLE_FILTER = "DISABLE_FriendlyUrlGenerationFilter";

 /** logger */
 private static final Logger LOGGER = Logger.getLogger(FriendlyUrlGenerationFilterFactory.class.getName());

 /** lock object for Identification lookup */
 private static final Object LOCK_IDENTIFICATION = new Object();

 /** the identification service */
 private static volatile Identification IDENTIFICATION;

 /** lock object for MappingURLTreeModelProvider lookup */
 private static final Object LOCK_MAPPING_URL_MODEL_PROVIDER = new Object();

 /** provider for url mapping model */
 private static volatile MappingURLTreeModelProvider MAPPING_URL_MODEL_PROVIDER;

 /** lock object for StateManagerHome lookup */
 private static final Object LOCK_STATE_MGR_SRV = new Object();

 /** state manager service */
 private static volatile PortletStateManagerService STATE_MGR_SRV;

 /** lock object for friendly selection service lookup */
 private static final Object LOCK_FRIENDLY_SEL_SRV = new Object();

 /** friendly selection service */
 private static volatile PortletFriendlySelectionServiceHome FRIENDLY_SEL_SRV;

 /** lock object for friendly selection service lookup */
 private static final Object LOCK_CONTENT_MAPPING_HOME = new Object();

 /** content mapping info home */
 private static volatile ContentMappingInfoHome CONTENT_MAPPING_HOME;

 /** lock object for web content service lookup */
```

```
private static final Object LOCK_WEB_CONTENT_SRV = new Object();

/** web content service */
private static volatile WebContentService WEB_CONTENT_SRV;

/** lock object for content model provider lookup */
private static final Object LOCK_CONTENT_MODEL_PROVIDER = new Object();

/** content model provider */
private static volatile ContentModelProvider CONTENT_MODEL_PROVIDER;

@Override
public ContentUrlGenerationFilter getFilter(final RenderRequest request, final RenderResponse response)
  throws ContentUrlFilterInstantiationException {

 final boolean isLogging = LOGGER.isLoggable(Level.FINEST);
 if (isLogging) {
  LOGGER.entering(getClass().getName(), "getFilter");
 }

 ContentUrlGenerationFilter result = null;

 // check if the filter is enabled. This filter can be disabled by
 // setting portlet preference to a value of 'true'
 if (!Boolean.valueOf(request.getPreferences().getValue(DISABLE_FILTER, Boolean.FALSE.toString()))) {
  try {
   Workspace workspace = getWebContentService().getRepository().getWorkspace();
   // user access is sufficient to lookup a item
   workspace.useUserAccess(true);
   result = new FriendlyUrlGenerationFilter(getFriendlySelectionServiceHome()
                       .getPortletFriendlySelectionService(request, response), getContentModelProvider()
                       .getContentModel(request, response), workspace, getIdentification(), getStateManageService()
                       .getPortletStateManager(request, response), getMappingURLTreeModelProvider()
                       .getMappingURLTreeModel(request, response), getContentMappingInfoHome());
  } catch (UnknownAccessorTypeException e) {
   throw new ContentUrlFilterInstantiationException(e);
  } catch (CannotInstantiateAccessorException e) {
   throw new ContentUrlFilterInstantiationException(e);
  } catch (PortletServiceUnavailableException e) {
   throw new ContentUrlFilterInstantiationException(e);
  } catch (ServiceNotAvailableException e) {
   throw new ContentUrlFilterInstantiationException(e);
  } catch (OperationFailedException e) {
   throw new ContentUrlFilterInstantiationException(e);
  } catch (StateManagerException e) {
   throw new ContentUrlFilterInstantiationException(e);
  } catch (ModelException e) {
   throw new ContentUrlFilterInstantiationException(e);
  } catch (StateException e) {
   throw new ContentUrlFilterInstantiationException(e);
  } catch (NamingException e) {
   throw new ContentUrlFilterInstantiationException(e);
  }
 }

 if (isLogging) {
  LOGGER.exiting(getClass().getName(), "getFilter", result);
 }
 return result;
}

@Override
public int getFilterChainWeight() {
 return 4;
}

/**
 * Get the identification service
 *
 * @return a reference to the identification service
 * @throws NamingException
 *             if creation of {@link InitialContext} or context lookup fails
 */
private static Identification getIdentification() throws NamingException {
 if (IDENTIFICATION == null) {
  synchronized (LOCK_IDENTIFICATION) {
   if (IDENTIFICATION == null) {
    final Context ctx = new InitialContext();
    IDENTIFICATION = (Identification) ctx.lookup(Identification.JNDI_NAME);
   }
  }
 }
 return IDENTIFICATION;
}

/**
 * Get the provider for URL mapping model
 *
 * @return A reference to the provider for the URL mapping model
 * @throws NamingException
```

```
     *             if creation of {@link InitialContext} or context lookup fails
     * @throws PortletServiceUnavailableException
     *             if the service is not available
     */
    private static MappingURLTreeModelProvider getMappingURLTreeModelProvider() throws NamingException,
      PortletServiceUnavailableException {
     if (MAPPING_URL_MODEL_PROVIDER == null) {
      synchronized (LOCK_MAPPING_URL_MODEL_PROVIDER) {
       if (MAPPING_URL_MODEL_PROVIDER == null) {
        final Context ctx = new InitialContext();
        final PortletServiceHome psh = (PortletServiceHome) ctx
          .lookup(MappingURLTreeModelProvider.JNDI_NAME);
        MAPPING_URL_MODEL_PROVIDER = psh.getPortletService(MappingURLTreeModelProvider.class);
       }
      }
     }
     return MAPPING_URL_MODEL_PROVIDER;
    }

    /**
     * Get the state manager service
     *
     * @return state manager service
     * @throws NamingException
     *             if creation of {@link InitialContext} or context lookup fails
     * @throws PortletServiceUnavailableException
     *             if the service is not available
     */
    private static PortletStateManagerService getStateManageService() throws NamingException,
      PortletServiceUnavailableException {
     if (STATE_MGR_SRV == null) {
      synchronized (LOCK_STATE_MGR_SRV) {
       if (STATE_MGR_SRV == null) {
        final Context ctx = new InitialContext();
        final PortletServiceHome psh = (PortletServiceHome) ctx
          .lookup(PortletStateManagerService.JNDI_NAME);
        STATE_MGR_SRV = psh.getPortletService(PortletStateManagerService.class);
       }
      }
     }
     return STATE_MGR_SRV;
    }

    /**
     * Get the friendly selection service home interface
     *
     * @return Home interface of friendly selection service
     * @throws NamingException
     *             if creation of {@link InitialContext} or context lookup fails
     * @throws PortletServiceUnavailableException
     *             if the service is not available
     */
    private static PortletFriendlySelectionServiceHome getFriendlySelectionServiceHome() throws NamingException,
      PortletServiceUnavailableException {
     if (FRIENDLY_SEL_SRV == null) {
      synchronized (LOCK_FRIENDLY_SEL_SRV) {
       if (FRIENDLY_SEL_SRV == null) {
        final Context ctx = new InitialContext();
        final PortletServiceHome psh = (PortletServiceHome) ctx
          .lookup(PortletFriendlySelectionServiceHome.JNDI_NAME);
        FRIENDLY_SEL_SRV = psh.getPortletService(PortletFriendlySelectionServiceHome.class);
       }
      }
     }
     return FRIENDLY_SEL_SRV;
    }

    /**
     * Get the content mapping info home interface
     *
     * @return Home interface of content mapping info service
     * @throws NamingException
     *             if creation of {@link InitialContext} or context lookup fails
     */
    private static ContentMappingInfoHome getContentMappingInfoHome() throws NamingException {
     if (CONTENT_MAPPING_HOME == null) {
      synchronized (LOCK_CONTENT_MAPPING_HOME) {
       if (CONTENT_MAPPING_HOME == null) {
        final Context ctx = new InitialContext();
        CONTENT_MAPPING_HOME = (ContentMappingInfoHome) ctx.lookup(ContentMappingInfoHome.JNDI_NAME);
       }
      }
     }
     return CONTENT_MAPPING_HOME;
    }

    /**
     * Get the web content service
     *
     * @return Reference to the web content service
```

```
  * @throws NamingException
  *             if creation of {@link InitialContext} or context lookup fails
  */
 private static WebContentService getWebContentService() throws NamingException {
  if (WEB_CONTENT_SRV == null) {
   synchronized (LOCK_WEB_CONTENT_SRV) {
    if (WEB_CONTENT_SRV == null) {
     final Context ctx = new InitialContext();
     WEB_CONTENT_SRV = (WebContentService) ctx.lookup("portal:service/wcm/WebContentService");
    }
   }
  }
  return WEB_CONTENT_SRV;
 }

 /**
  * Get the content model provider
  *
  * @return Reference to the content model provider
  * @throws NamingException
  *             if creation of {@link InitialContext} or context lookup fails
  * @throws PortletServiceUnavailableException
  *             if the service is not available
  */
 private static ContentModelProvider getContentModelProvider() throws NamingException,
   PortletServiceUnavailableException {
  if (CONTENT_MODEL_PROVIDER == null) {
   synchronized (LOCK_CONTENT_MODEL_PROVIDER) {
    if (CONTENT_MODEL_PROVIDER == null) {
     final Context ctx = new InitialContext();
     final PortletServiceHome psh = (PortletServiceHome) ctx.lookup(ContentModelProvider.JNDI_NAME);
     CONTENT_MODEL_PROVIDER = psh.getPortletService(ContentModelProvider.class);
    }
   }
  }
  return CONTENT_MODEL_PROVIDER;
 }
}
```

Example 2: Filter class:

This sample filter generates the friendly URL.

The filter runs several steps to create the friendly URL:

1. The filter determines the target portal page from one of the following sources:
   - The configuration of the web content viewer.
   - Any web content pages that have a content association to the content for which the URL is generated.
   - A target page that is specified by the `UrlCmpnt` tag.
2. If a target page is identified, the filter verifies that the page is a web content page with a content association. The filter then validates that the content for which the URL is generated is a child of the site area that is mapped to the page. If the content to render is not a child of the site area that is associated with the page, the filter writes a new URL.
3. The filter then writes the friendly URL by combining the following information:
   - The friendly URL name of the target page.
   - The path to the content, relative to the site area that is associated with the target page.

```
/*****************************************************************
 * Copyright IBM Corp. 2011, 2014                                *
 *****************************************************************/
package com.ibm.workplace.wcm.api.samples;

import java.io.*;
import java.net.*;
import java.util.*;
import java.util.logging.*;
import java.util.regex.*;
import javax.portlet.*;
import com.ibm.portal.*;
import com.ibm.portal.content.*;
import com.ibm.portal.identification.*;
import com.ibm.portal.mappingurl.*;
import com.ibm.portal.resolver.friendly.*;
```

```
import com.ibm.portal.resolver.friendly.accessors.url.*;
import com.ibm.portal.resolver.friendly.helper.*;
import com.ibm.portal.resolver.friendly.service.*;
import com.ibm.portal.serialize.*;
import com.ibm.portal.services.contentmapping.*;
import com.ibm.portal.services.contentmapping.exceptions.*;
import com.ibm.portal.state.*;
import com.ibm.portal.state.accessors.selection.*;
import com.ibm.portal.state.exceptions.*;
import com.ibm.workplace.wcm.api.*;
import com.ibm.workplace.wcm.api.exceptions.*;
import com.ibm.workplace.wcm.api.extensions.url.*;
import com.ibm.workplace.wcm.api.extensions.url.PortletContextSharingConfig.PublishConfig;

/**
 * Content URL generation filter that tries to generate stateless friendly URLs
 * for web content pages.
 *
 * The filter that writes the friendly URL does the following steps to generate
 * the friendly URL
 *
 * <ol>
 * <li>It determines the target portal page from one of the following sources The
 * Web Content Viewer configuration Web content pages that have a content
 * mapping for the content the URL is generated for A target page specification
 * from the WCM [UrlCmpnt] tag
 * </li>
 * <li>If a page could be determined it checks if the page is a web content page
 * i.e. if the page has a content mapping assign. It then validates that the
 * content the URL is generated for is a children of the site area mapped to the
 * page. In case the content is not a children of the site area mapped to the
 * page new URL is written by this filter.
 * </li>
 * <li>Finally the friendly URL is written that is build from the friendly URL
 * name of the target page appended with the content path relative to the site
 * area mapped to the target page.</li>
 * </ol>
 * <p>
 * <b>Note:</b> In order to use the following sample filter all pages a content URL is
 * generated for need to be web content pages with a friendly name assigned and
 * a default content mapping that points to a parent of the content.</p>
 */
public class FriendlyUrlGenerationFilter implements ContentUrlGenerationFilter {

 /** logger */
 private static final Logger LOGGER = Logger.getLogger(FriendlyUrlGenerationFilter.class.getName());

 /** the path separator / */
 private static final String PATH_SEPARATOR = "/";

 /** regular expression pattern to split a path into segments */
 private static final Pattern PATH_SEPARATOR_PATTERN = Pattern.compile(PATH_SEPARATOR);

 /** friendly selection service */
 private final FriendlySelectionService friendlySelectionService;

 /** content model */
 private final ContentModel<ContentNode> contentModel;

 /** WCM workspace */
 private final Workspace workspace;

 /** identifcation service */
 private final Identification identification;

 /** state manager */
 private final PortletStateManager stateManager;

 /** url mapping model */
 private final MappingURLTreeModel urlMappingModel;

 /** content mapping info home */
 private final ContentMappingInfoHome contentMappingInfoHome;

 /** selection accessor */
 private final SelectionAccessorFactory selectionFactory;

 /** factory for friendly URLs */
 private final FriendlyURLFactory friendlyUrlFactory;

 /** the currently selected page */
 private ObjectID currentPage;

 /**
  * Create a new filter instance. This should be called once per render
  * request
  *
  * @param friendlySelectionService
  *            The friendly selection service
  * @param contentModel
```

```
 *            The content model
 * @param workspace
 *            The WCM workspace
 * @param identification
 *            The identificaton service
 * @param stateManager
 *            The state manager service
 * @param urlMappingTreeModel
 *            The url mapping model
 * @param contentMappingInfoHome
 *            The content mapping home interface
 *
 * @throws CannotInstantiateAccessorException
 *             If instantiation of state selection accessor factory fails
 * @throws UnknownAccessorTypeException
 *             If instantiation of state selection accessor factory fails
 */
public FriendlyUrlGenerationFilter(final FriendlySelectionService friendlySelectionService,
  final ContentModel<ContentNode> contentModel, final Workspace workspace,
  final Identification identification, final PortletStateManager stateManager,
  final MappingURLTreeModel urlMappingModel, final ContentMappingInfoHome contentMappingInfoHome)
  throws UnknownAccessorTypeException, CannotInstantiateAccessorException {

 final boolean isLogging = LOGGER.isLoggable(Level.FINEST);
 if (isLogging) {
  LOGGER.entering(getClass().getName(), "<init>", new Object[] { friendlySelectionService, contentModel,
    workspace, identification, stateManager, urlMappingModel, contentMappingInfoHome });
 }

 this.friendlySelectionService = friendlySelectionService;
 this.friendlyUrlFactory = friendlySelectionService.getURLFactory();
 this.contentModel = contentModel;
 this.workspace = workspace;
 this.identification = identification;
 this.stateManager = stateManager;
 this.urlMappingModel = urlMappingModel;
 this.contentMappingInfoHome = contentMappingInfoHome;
 this.selectionFactory = stateManager.getAccessorFactory(SelectionAccessorFactory.class);

 if (isLogging) {
  LOGGER.exiting(getClass().getName(), "<init>");
 }
}

@Override
public void dispose() {
 final boolean isLogging = LOGGER.isLoggable(Level.FINEST);
 if (isLogging) {
  LOGGER.entering(getClass().getName(), "dispose");
 }

 // dispose all request specific services
 this.friendlySelectionService.dispose();
 this.stateManager.dispose();

 if (isLogging) {
  LOGGER.exiting(getClass().getName(), "dispose");
 }
}

@Override
public void writeURL(final ContentUrlGenerationRequest request, final ContentUrlGenerationResponse response,
  final ContentUrlGenerationFilterChain chain) throws ContentUrlGenerationException, IOException {

 final boolean isLogging = LOGGER.isLoggable(Level.FINEST);
 if (isLogging) {
  LOGGER.entering(getClass().getName(), "writeURL", new Object[] { request.getContentPath(false) });
 }

 // as we need to use the path to lookup the item in WCM we need the
 // decoded version
 final String contentPath = request.getContentPath(false);
 if (contentPath != null) {
  // Check if we should generate a URL that publishes to the
  // current
  // or another page or uses the dynamic publishing
  final PortletContextSharingConfig ctxSharingConfig = request.getPortletContextSharingConfig();
  final PublishConfig publishConfig = ctxSharingConfig.getPublishConfig();
  final PortletRequest portletRequest = request.getPortletRenderRequest();
  final PortletResponse portletResponse = request.getPortletRenderResponse();

  ObjectID targetPageId = null;
  try {
   // determine the target page. The target page is determined
   // from either a dynamic target page override (i.e. on the
   // UrlCmpnt tag), a web content mapping on a page or from the
   // portlet configuration

   // check if a dynamic page target as been set as it can be
   // set on the WCM UrlCmpnt tag
```

```
   final TargetPageConfig targetPageDynamic = request.getDynamicTargetPageOverride();
   if (targetPageDynamic != null) {
    // lookup the page from the dynamic target page override
    targetPageId = getTargetPage(portletRequest, portletResponse, targetPageDynamic);
   } else {
    if (publishConfig.getMode() == PublishConfig.MODE_DYNAMIC) {
     // lookup the target page from content mappings
     targetPageId = lookupTargetPage(portletRequest, portletResponse, contentPath);
    } else {
     // target page is determined from portlet
     // configuration
     final TargetPageConfig targetPagePortletConfig = publishConfig.getTargetPage();
     if (targetPagePortletConfig != null) {
      // lookup the page from the portlet target page
      // configuration
      targetPageId = getTargetPage(portletRequest, portletResponse, targetPagePortletConfig);
     }
    }
   }

   if (targetPageId != null) {
    // check if the path of the content is a children of the
    // site area mapped to the page and get the path relative to
    // this site area
    final String relativePathInfo = getRelativePathInfo(contentPath, targetPageId);
    if (relativePathInfo != null) {
     // write the friendly URL to the page and the
     // relative path information added
     final FriendlyURL url = this.friendlyUrlFactory
       .newURL(com.ibm.portal.state.Constants.Clone.EMPTY_COPY);
     url.setSelection(targetPageId);
     if(!relativePathInfo.isEmpty()) {
      url.setPathInfo(relativePathInfo);
     }
     url.writeDispose(response.getWriter());
    } else {

     if (isLogging) {
      LOGGER.logp(Level.FINEST, getClass().getName(), "writeURL",
        "Content [{0}] is not a children of the site area mapped to page with ID [{1}]",
        new Object[] { contentPath, targetPageId });
     }

     // the content is not a children of the site area
     // mapped to the target page so forward the request to
     // the chain of URL
     chain.writeURL(request, response);
    }
   } else {

    if (isLogging) {
     LOGGER.logp(Level.FINEST, getClass().getName(), "writeURL",
       "No target page could be determined for content [{0}]", new Object[] { contentPath });
    }

    // no target page could be determined
    // let the content URL generation chain handle the
    // request
    chain.writeURL(request, response);
   }
  } catch (SerializationException e) {
   throw new ContentUrlGenerationException(e);
  } catch (ModelException e) {
   throw new ContentUrlGenerationException(e);
  } catch (StateException e) {
   throw new ContentUrlGenerationException(e);
  } catch (ContentMappingException e) {
   throw new ContentUrlGenerationException(e);
  } catch (WCMException e) {
   throw new ContentUrlGenerationException(e);
  }
 } else {
  // no content path was given
  // let the content URL generation chain handle the request
  chain.writeURL(request, response);
 }

 if (isLogging) {
  LOGGER.exiting(getClass().getName(), "writeURL");
 }
}

/**
 * Lookup the best matching target web content page for the content
 *
 * @param portletRequest
 *            The current portlet request
 * @param portletResponse
 *            The current portlet request
 * @param contentPath
```

```
 *            The path of the content
 * @return The {@link ObjectID} of page found or <code>null</code>
 *
 * @throws ContentMappingException
 *             If an error occurred loading a content mapping
 * @throws ModelException
 *             If an exception occurred while accessing a model object
 * @throws WCMException
 *             If an exception occurred while accessing the WCM repository
 * @throws StateException
 *             If an error occurred working with the portal state objects
 */
protected ObjectID lookupTargetPage(final PortletRequest portletRequest, final PortletResponse portletResponse,
  final String contentPath) throws ContentMappingException, ModelException, WCMException, StateException {

 final boolean isLogging = LOGGER.isLoggable(Level.FINEST);
 if (isLogging) {
  LOGGER.entering(getClass().getName(), "lookupTargetPage", new Object[] { contentPath });
 }

 ObjectID result = null;

 // get the ID of the published item addressed by the content path
 final DocumentIdIterator documentsIt = this.workspace.findByPath(contentPath,
   Workspace.WORKFLOWSTATUS_PUBLISHED);
 if (documentsIt.hasNext()) {
  // get the IDs of the content and all its parents
  final LinkedList<String> resourceIds = new LinkedList<String>();
  final DocumentId documentId = documentsIt.next();
  resourceIds.push(documentId.getId());

  // load the IDs of the parents of the item
  DocumentId parentId = documentId;
  do {
   Document doc = this.workspace.getById(parentId);
   parentId = null;
   if (doc instanceof Content) {
    parentId = ((Content) doc).getDirectParent();
   } else if (doc instanceof ContentLink) {
    parentId = ((ContentLink) doc).getParentId();
   } else if (doc instanceof SiteFrameworkContainer) {
    parentId = ((SiteFrameworkContainer) doc).getParent();
   }
   if (parentId != null) {
    resourceIds.push(parentId.getId());
   }
  } while (parentId != null);

  // add the library of the content to the beginning
  resourceIds.push(documentId.getContainingLibrary().getId());

  if (isLogging) {
   LOGGER.logp(Level.FINEST, getClass().getName(), "lookupTargetPage",
     "Lookup up best matching web content page for resources [{0}] using the following IDs [{1}]",
     new Object[] { contentPath, resourceIds });
  }

  // lookup the best matching web content page
  final ContentMappingLocator contentMappinglocator = this.contentMappingInfoHome.getContentMappingLocator();
  final LongestPathMatch match = contentMappinglocator.getLongestPathMatch(resourceIds,
    getCurrentPage(portletRequest, portletResponse), new ContentMappingFilter() {
     public void filterEntitledMappings(List<? extends ContentMapping> mappings) {
      // filter out pages we cannot locate e.g. the
      // user doesn't have access to or if the page is
      // disabled
      final Locator<ContentNode> contentNodeLocator = FriendlyUrlGenerationFilter.this.contentModel
        .getLocator();
      final Iterator<? extends ContentMapping> mappingsIt = mappings.iterator();
      while (mappingsIt.hasNext()) {
       if (contentNodeLocator.findByID(mappingsIt.next().getResourceID()) == null) {
        mappingsIt.remove();
       }
      }
     }
    });

  // if at least one match was found take the suggest content
  // mapping further candidates might be found
  final ContentMapping contentMapping = match.getContentMapping();
  if (contentMapping != null) {
   result = contentMapping.getResourceID();
  }
 }

 if (isLogging) {
  LOGGER.exiting(getClass().getName(), "lookupTargetPage", result);
 }
 return result;
}
```

```
/**
 * Get the {@link ObjectID} of the target page from a target page
 * configuration.
 *
 * @param portletRequest
 *            The current portlet request
 * @param portletResponse
 *            The current portlet request
 * @param targetPageConfig
 *            The target page configuration
 * @return The {@link ObjectID} of the target page
 *
 * @throws SerializationException
 *             If the a page ID given as a character string cannot be
 *             serialized to an {@link ObjectID}
 * @throws ModelException
 *             If an exception occurred while accessing a model object
 * @throws StateException
 *             if an error occurred working with the portal state objects
 */
protected ObjectID getTargetPage(final PortletRequest portletRequest, final PortletResponse portletResponse,
  final TargetPageConfig targetPageConfig) throws SerializationException, ModelException, StateException {

 final boolean isLogging = LOGGER.isLoggable(Level.FINEST);
 if (isLogging) {
  LOGGER.entering(getClass().getName(), "getTargetPage", new Object[] { targetPageConfig });
 }

 ObjectID result = null;
 if (targetPageConfig != null) {
  if (targetPageConfig.useCurrentPage()) {
   result = getCurrentPage(portletRequest, portletResponse);
  } else {
   final String pagePath = targetPageConfig.getPagePath();
   if (pagePath != null && !pagePath.isEmpty()) {
    // try to lookup the page treating the path as a URL mapping
    result = getPageByUrlMapping(portletRequest, portletResponse, pagePath);
    if (result == null) {
     // if no mapping was found, check if the path is a
     // valid friendly URL
     final List<ObjectID> pages = getPagesByFriendlyUrl(portletRequest, portletResponse, pagePath);
     if (pages != null && !pages.isEmpty()) {
      // if multiple pages are found for simplicity use
      // the
      // first page more advance URL generation filter
      // could
      // do a disambiguation here and e.g. let the user
      // choose
      // what page to use
      result = pages.get(0);
     }
    }
   } else {
    result = getPageById(targetPageConfig.getPageId());
   }
  }
 }

 if (isLogging) {
  LOGGER.exiting(getClass().getName(), "getTargetPage", result);
 }
 return result;
}

/**
 * Get the {@link ObjectID} of the page with the given ID or unique name
 *
 * @param pageId
 *            The ID or unique name of the page
 * @return The {@link ObjectID} of the page
 *
 * @throws SerializationException
 *             If the a page ID given as a character string cannot be
 *             serialized to an {@link ObjectID}
 */
protected ObjectID getPageById(final String pageId) throws SerializationException {

 final boolean isLogging = LOGGER.isLoggable(Level.FINEST);
 if (isLogging) {
  LOGGER.entering(getClass().getName(), "getPageById", new Object[] { pageId });
 }

 ObjectID result = null;
 if (pageId != null && !pageId.isEmpty()) {
  // de-serialize the ID
  result = this.identification.deserialize(pageId);
 }

 if (isLogging) {
  LOGGER.exiting(getClass().getName(), "getPageById", result);
```

```
 }
 return result;
}

/**
 * Get the {@link ObjectID} of the current page
 *
 * @param portletRequest
 *            The current portlet request
 * @param portletResponse
 *            The current portlet request
 *
 * @return The {@link ObjectID} of the current page
 *
 * @throws StateException
 *             if an error occurred working with the portal state objects
 */
protected ObjectID getCurrentPage(final PortletRequest portletRequest, final PortletResponse portletResponse)
  throws StateException {

 final boolean isLogging = LOGGER.isLoggable(Level.FINEST);
 if (isLogging) {
  LOGGER.entering(getClass().getName(), "getCurrentPage");
 }

 if (currentPage == null) {
  final SelectionAccessor selectionAcc = this.selectionFactory.getSelectionAccessor(this.stateManager
    .getStateHolder());
  try {
   currentPage = selectionAcc.getSelection();
  } finally {
   selectionAcc.dispose();
  }
 }

 if (isLogging) {
  LOGGER.exiting(getClass().getName(), "getCurrentPage", currentPage);
 }
 return currentPage;
}

/**
 * Get the list of {@link ObjectID} of all page that are addressed by the
 * passed friendly name
 *
 * @param portletRequest
 *            The current portlet request
 * @param portletResponse
 *            The current portlet request
 * @param friendlyName
 *            The friendly name
 * @return List of all pages that are addressed by the passed friendly name
 *
 * @throws ModelException
 *             If looking up the page from a friendly URL fails
 * @throws StateException
 *             if the state could not be accessed
 */
protected List<ObjectID> getPagesByFriendlyUrl(final PortletRequest portletRequest,
  final PortletResponse portletResponse, final String friendlyName) throws ModelException, StateException {

 final boolean isLogging = LOGGER.isLoggable(Level.FINEST);
 if (isLogging) {
  LOGGER.entering(getClass().getName(), "getPagesByFriendlyUrl", new Object[] { friendlyName });
 }

 List<ObjectID> result = null;

 if (friendlyName != null && !friendlyName.isEmpty()) {
  final SelectionResult bean = new DefaultSelectionResult();
  this.friendlySelectionService.resolve(bean, friendlyName);
  // the resulting node list is already AC filtered as a
  // result of using a performing navigation model.
  final List<ObjectID> nodelist = bean.getNodes();
  if (nodelist != null && !nodelist.isEmpty() && bean.getFriendlyPath() != null) {
   result = nodelist;
  }
 }

 if (isLogging) {
  LOGGER.exiting(getClass().getName(), "getPagesByFriendlyUrl", result);
 }
 return result;
}

/**
 * Get the {@link ObjectID} of the page addressed by the passed compound
 * name of a url mapping or <code>null</code> if no corresponding URL
 * mapping or page exists or if the current user does not have access to it.
 *
```

```
 * @param portletRequest
 *            The current portlet request
 * @param portletResponse
 *            The current portlet request
 * @param urlMapping
 *            The compound name of the url mapping
 * @return {@link ObjectID} of the page or <code>null</code>
 * @throws ModelException
 *             If an exception occurred while accessing the url mapping
 *             model
 */
protected ObjectID getPageByUrlMapping(final PortletRequest request, final PortletResponse response,
  final String urlMapping) throws ModelException {

 final boolean isLogging = LOGGER.isLoggable(Level.FINEST);
 if (isLogging) {
  LOGGER.entering(getClass().getName(), "getPageByUrlMapping", new Object[] { urlMapping });
 }

 ObjectID result = null;
 if (urlMapping != null && !urlMapping.isEmpty()) {
  final BestMatchResult searchResult;
  // different to friendly names a URL mapping must not begin with a /
  if (urlMapping.charAt(0) == PATH_SEPARATOR.charAt(0)) {
   searchResult = this.urlMappingModel.getLocator().findBestMatch(urlMapping.substring(1));
  } else {
   searchResult = this.urlMappingModel.getLocator().findBestMatch(urlMapping);
  }

  if (searchResult != null) {
   final Context mappingCtx = searchResult.getContext();
   if (ObjectTypeConstants.PORTAL_URL.getType().equals(mappingCtx.getAssignedObjectType())) {
    final PortalURL url = (PortalURL) mappingCtx.getAssignedObject();
    result = url.getReferencedResourceID();
   }
  }
 }

 if (isLogging) {
  LOGGER.exiting(getClass().getName(), "getPageByUrlMapping", result);
 }
 return result;
}

/**
 * Returns the path for the given content path relative to the site area
 * mapped to the target page.
 *
 * Returns <code>null</code> if there is no content mapping set for the
 * target page that is appropriate for the targeted content item.
 *
 * @param contentPath
 *            The fully qualified path of the target content item. Must not
 *            be <code>null</code>.
 * @param pageId
 *            The object ID of the target page. Must not be
 *            <code>null</code>.
 * @return The relative path which is the remainder of the content path
 *         after cutting off the content mapping prefix.
 *         May return <code>null</code>.
 *         Returns an empty string if the given path points directly
 *         to the site area that is mapped to the target page.
 * @throws ContentMappingException
 *             If an exception occurred during lookup of the content mapping
 * @throws WCMException
 *             If an exception occurred while accessing the WCM repository
 * @throws UnsupportedEncodingException
 *             A requested character encoding is not supported
 */
protected String getRelativePathInfo(final String contentPath, final ObjectID pageId)
  throws ContentMappingException, WCMException, UnsupportedEncodingException {

 final boolean isLogging = LOGGER.isLoggable(Level.FINEST);
 if (isLogging) {
  LOGGER.entering(getClass().getName(), "getRelativePathInfo", new Object[] { contentPath, pageId });
 }

 String result = null;

 final ContentMapping contentMapping = getDefaultContentMapping(pageId);
 if (contentMapping != null) {
  // lookup the path of the site area mapped to the page
  String pathMapping = contentMapping.getContentPath();
  if (pathMapping == null || pathMapping.isEmpty()) {
   // lets lookup the path from the id
   final String mappedId = contentMapping.getContentID();
   if (mappedId != null && !mappedId.isEmpty()) {
    pathMapping = this.workspace.getPathById(this.workspace.createDocumentId(mappedId), false, true);
   }
  }
```

```
  if (isLogging) {
   LOGGER.logp(Level.FINEST, getClass().getName(), "getRelativePathInfo",
     "Page with ID [{0}] is mapped to [{1}]", new Object[] { pageId, pathMapping });
  }

  // calculate relative path = contentPath - mappingPath
  if (pathMapping != null && !pathMapping.isEmpty()) {
   // check if the content path is a children of the mapped path
   // to do this split the path into its segments
   if (pathMapping.charAt(0) == PATH_SEPARATOR.charAt(0)) {
    pathMapping = pathMapping.substring(1);
   }
   final String[] partsPathMapping = PATH_SEPARATOR_PATTERN.split(pathMapping);
   // also split path of content
   String pathContent = contentPath;
   if (pathContent.charAt(0) == PATH_SEPARATOR.charAt(0)) {
    pathContent = pathContent.substring(1);
   }
   final String[] partsPathContent = PATH_SEPARATOR_PATTERN.split(pathContent);

   // check if the content is a children of the mapped path
   if (partsPathMapping.length <= partsPathContent.length) {
    boolean isDescendant = true;
    for (int i = 0; i < partsPathMapping.length && isDescendant; i++) {
     if (!partsPathMapping[i].equalsIgnoreCase(partsPathContent[i])) {
      isDescendant = false;
     }
    }
    if (isDescendant) {
     // determine how many descendant levels are between the
     // content and the mapped site area
     final int descendantLevels = partsPathContent.length - partsPathMapping.length;
     if (descendantLevels > 0) {
      // build children path which is
      // everything after the parent
      final StringBuilder tmp = new StringBuilder();
      for (int i = 0; i < descendantLevels; i++) {
       tmp.append(PATH_SEPARATOR);
       tmp.append(URLEncoder.encode(partsPathContent[partsPathMapping.length + i], "UTF-8"));
      }
      result = tmp.toString();
     } else if (descendantLevels == 0) {
      // the content path points directly to the
      // site area that is mapped to the page
      result = "";
     }
    }
   }
  }

  if (isLogging) {
   LOGGER.exiting(getClass().getName(), "getRelativePathInfo", result);
  }
  return result;
 }

 /**
  * Get the default content mapping of a page or <code>null</code> if no such
  * mapping exists
  *
  * @param pageId
  *            The {@link ObjectID} of the page
  * @return The default mapping of the page or <code>null</code> if no
  *         default mapping could be determined.
  *
  * @throws ContentMappingDataBackendException
  *             If an exception occurred during lookup of the content mapping
  */
 protected ContentMapping getDefaultContentMapping(final ObjectID pageId) throws ContentMappingDataBackendException {

  final boolean isLogging = LOGGER.isLoggable(Level.FINEST);
  if (isLogging) {
   LOGGER.entering(getClass().getName(), "getDefaultContentMapping", new Object[] { pageId });
  }

  // get the page default content mapping as friendly url path info is
  // only set for default or system content mapping
  final ContentMappingInfo contentMappingInfo = this.contentMappingInfoHome.getContentMappingInfo(pageId);
  ContentMapping result = contentMappingInfo.getDefaultContentMapping();
  if(result == null) {
      // use system mapping as default
      result = contentMappingInfo.getSystemContentMapping();
  }

  if (isLogging) {
   LOGGER.exiting(getClass().getName(), "getDefaultContentMapping", result);
```

```
  }
  return result;
 }
}
```

Example 2: `plugin.xml` file:

You can use this sample `plugin.xml` file to register the sample filter.

```
<?xml version="1.0" encoding="UTF-8"?>
<?eclipse version="3.0"?>
<plugin id="com.ibm.workplace.wcm.api.samples.plugin" name="My content URL plug-in"
        version="1.0.0" provider-name="IBM">
     <extension point="com.ibm.workplace.wcm.api.ContentUrlGenerationFilter"
                id="MyFriendlyUrlGenerationFilter">
        <factory class="com.ibm.workplace.wcm.api.samples.FriendlyUrlGenerationFilterFactory"
                 weight="4"/>
     </extension>
</plugin>
```

Deploying custom plug-in applications

You must deploy your custom plug-in applications on your server before they can be used in your web content system.

To ensure that the new web content class is available each time your server is started, register the ear file in the WebSphere Integrated Solutions Console:

1. Select **Applications**.
2. Select **New Application**.
3. Select **New Enterprise Application**.
4. Browse for the ear file.
5. Select **next** until you reach **step 2. Map modules to servers**.
6. Select the check box for the custom action module.
7. In the Clusters and servers section, select **WebSphere_Portal** and select **Apply**.
8. Keep selecting **Next** until the end when you select **Finish**.
9. The screen titled Installing is shown. Select **Manage Applications**.
10. Locate and click the application that you installed. The default name is the display name that is defined in the `application.xml` file in your ear file.
11. Under Detail Properties select **Startup behavior**.
12. Under General Properties modify the Startup order to be the same weight as "wcm" and select **Apply**. By default, the weight is 20.
13. Select **Save directly to master configuration**.
14. To immediately start the application you installed, select the application and click **Start**.

To update an existing ear file:

1. Select **Applications**.
2. Select **Application Types**.
3. Select **WebSphere enterprise applications**.
4. Search for the Application name.
5. Select the check box for the custom action application and click **Update**.
6. Select **Replace the entire application** and browse for the ear file.
7. Keep selecting **Next** until the end when you select **Finish**.

8. The screen that is titled 'updating' is shown. Select **Save to Master Configuration** and then click **save**.

IBM Digital Data Connector (DDC) for WebSphere Portal Express

You can use the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework to integrate data from external data sources on your portal pages by using IBM Web Content Manager presentation components. External data means that the data does not need to be stored directly in IBM Web Content Manager. For example, you can use DDC to render social data that you have on your IBM Connections server or on other social platforms in the context of your portal pages. Other possible data sources include news feeds, task lists, product catalog information, to name just a few.

With Digital Data Connector, your website designers can use Web Content Manager presentation components to generate the web page markup for your external data. They can use all the Web Content Manager data management facilities for managing your external data visualizations. These facilities include content syndication, version handling, workflow, and targeting. They can manage the design components in the same way as your other Web Content Manager content and design components. The major benefits of this approach include the following:

- Your Web Content Manager designers can fully control the visual appearance of the integrated data.
- They can visualize the external data in the same way in which they visualize data that is stored in Web Content Manager.
- As a result, they can visualize the external data in a way that is consistent with the corporate design of your overall website by reusing existing Web Content Manager components.
- To quickly adjust existing visualizations of your data or create new visualizations for new kinds of external data, you no longer need the help of software developers or the IT department. Your website designers can start working on the presentation templates directly from your portal pages that show the data. They use the inline editing capabilities of Web Content Manager.
- Your website designers make updates to the Web Content Manager design components in project scope. This way, they can keep updates in draft stage until all updates to the project are completed, approved, and finally published.

You can use Digital Data Connector in the following ways:

- You can code a Java plug-in, a so-called DDC plug-in, that hooks into the DDC. The plug-in loads the external data and transforms it into a generic DDC data structure, so-called bean lists. You can then have the bean lists rendered on your portal pages by using standard Web Content Manager rendering methods.
- You can use the generic XML DDC plug-in that is built into Digital Data Connector. You can use this plug-in to integrate remote XML data without writing or deploying extra Java code. It supports various parameters that you can use to specify from where the plug-in obtains the XML data and how it transforms the data. The transformation turns a specific XML document format into the generic DDC bean list data structure. With DDC, you can define these transformations in a declarative way so that you can use arbitrary XML formats without having to write transformation code.
- You can also use a combination of the two approaches.

"Technical concepts" on page 3257
Before you use the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework, you can familiarize yourself with its main technical concepts and building blocks.

"The rendering flow" on page 3259
Read about the rendering flow of IBM Digital Data Connector (DDC) for WebSphere Portal Express.

"Implementing user interactions" on page 3259
With IBM Digital Data Connector (DDC) for WebSphere Portal Express, you can use IBM Web Content Manager presentation components for visualizing external data. As a further benefit, you can also add user interfaces for creating, modifying, and deleting external data to your DDC list appearance components. This way, your users can use these features to create, modify, or delete external data items from your lists.

"Working with list-rendering profiles" on page 3271
You can create new or derived list-rendering profiles. After you create a list-rendering profile, you deploy it.

"Integrating remote XML data" on page 3273
The IBM Digital Data Connector (DDC) for WebSphere Portal Express framework provides a generic XML DDC plug-in that is ready to use for integrating external XML data of your choice. You can use this plug-in to render external XML data on your portal pages without having to write custom Java code.

CF06 "Integrating remote JSON data" on page 3286
The IBM Digital Data Connector (DDC) for WebSphere Portal Express framework provides a generic JSON DDC plug-in that is ready to use for integrating external JSON data of your choice. You can use this plug-in to render external JSON data on your portal pages without having to write custom Java code.

"Creating and deploying custom Digital Data Connector plug-ins" on page 3296
You can deploy custom IBM Digital Data Connector (DDC) for WebSphere Portal Express plug-ins as plug-ins into the WebSphere Portal application extension registry.

CF03 "Creating and deploying custom attribute value processor plug-ins" on page 3296
You can deploy custom attribute value processor plug-ins for DDC into the WebSphere Portal application extension registry. Attribute value processor plug-ins can be used to process the value of an item attribute after the value is determined by the list rendering profile.

"Digital Data Connector cache tuning" on page 3297
To improve performance, you can tune the caches for the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework.

"Hints and tips for Digital Data Connector" on page 3298
Learn about things that are useful to know when you work with the social rendering integration with IBM Digital Data Connector (DDC) for WebSphere Portal Express.

Related concepts:

The list-rendering context

Related information:

The action URL plug-in

The render parameter plug-in

The render URL plug-in

The request attribute plug-in

The session attribute plug-in

The resource URL plug-in

Technical concepts

Before you use the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework, you can familiarize yourself with its main technical concepts and building blocks.

To integrate external data of your choice into your WebSphere Portal Express, you hook into the Digital Data Connector framework. To do so, you have the following alternatives:

- You can write a DDC plug-in for the WebSphere extension registry. This approach provides full flexibility but requires creating and deploying Java code.
- If your external data is available in XML format, you can use the generic XML DDC plug-in to integrate your data. In this case, you do not have to write any code. Instead, you integrate data that comes from a specific XML data source by deploying a list-rendering profile.
- You can also use a combination of both approaches. In such a setup, you typically create a custom DDC plug-in. This custom plug-in delegates the loading and transformation of XML data to the generic XML DDC plug-in. You can use the custom plug-in to determine the correct source URL based on the current rendering context. For example, the plug-in can generate a specific product query URL to an e-commerce server. It bases the query on user preferences, request attributes, user agent, and other context information. Furthermore, the custom plug-in can modify the bean list objects that the generic XML DDC plug-in delegate returns. For example, the plug-in can add extra attributes that are computed from the data that is contained in the original bean list object.

The following list shows and describes the main building blocks of the Digital Data Connector framework:

DDC plug-in
: This term refers to Java plug-ins that are hooked into the Digital Data Connector framework. DDC plug-ins implement the `com.ibm.portal.wcm.plr.BeanListProvider` Java interface as defined by the public Digital Data Connector Java API. DDC plug-ins load external data and transform this data into a generic data structure that can be rendered by using list appearance components. List appearance components are described later in this list. WebSphere Portal Express Version 8.5 comes with two preinstalled DDC plug-ins:
  - A DDC plug-in for integrating social data made available by remote Connections servers. This DDC plug-in is used by the social lists feature.
  - A generic XML DDC plug-in. You can use this provider directly for integrating arbitrary remote XML data. You do not have to write or deploy more Java code.

Bean list
: This term refers to the abstract data structure that a DDC plug-in generates based on external data. DDC always represents external data as a list. You can generate detail views for individual data items by using lists of length 1. Individual bean list objects can be transformed into markup by using list appearance components. List appearance components are described later in this list.

List-rendering profile

A list-rendering profile defines the set of attributes available in the beans that are contained in bean lists that a DDC plug-in generates. Furthermore, the generic XML DDC plug-in supports list-rendering profiles to define the mapping between the XML data structure and the actual attribute values. You define this mapping by associating the attribute definitions in the profile with individual XPath expressions.

Digital Data Connector selection rule

This term refers to specific rules that you can create in the WebSphere Portal Express Personalization component or in Web Content Manager Personalization components. A Digital Data Connector selection rule is a Personalization rule of type **Select Action** defined to work on the Pluggable Resources resource collection. You do not have to add any additional properties or selection criteria to this rule. This rule is used to trigger the Digital Data Connector framework when you render the selection rule in a personalization component.

List appearance

This term refers to a Web Content Manager Personalization component that contains or references a Digital Data Connector selection rule. The list appearance defines the visual design of your list by defining the markup fragments that are generated during rendering of the lists. The individual data fragments that the DDC plug-in loads can be referenced by the list appearance by using the `[AttributeResource]` tag. This reference mechanism works the same way as when you render Personalization components that select data from the Web Content or the Web Components resource collections. The set of supported `attributeName` parameter values for the `[AttributeResource]` tag depends on the specific DDC plug-in and the list-rendering profile that you use.

The list appearance also defines the paging behavior of the list.

The term "list appearance" is also used for components that generate details views of individual items. In this case, the individual items are represented by a list of length 1. List appearance components that generate such details views can also be referred to as "details appearance" components.

List-rendering context

This term refers to the context that the `[Plugin:ListRenderingContext]` rendering plug-in generates to control the contents of your lists. This context includes all of the following information:

- Which DDC plug-in you want to be called for building the bean list object from the actual external data
- The selected list-rendering profile
- A list of custom attributes
- Access to the current portlet request and response objects
- A reference to the currently rendered list definition content item.

The addressed DDC plug-in then evaluates the list-rendering context so that it can query the appropriate set of remote information. You can establish the list-rendering context by adding a `[Plugin:ListRenderingContext]` Web Content Manager tag to your presentation templates. You add this tag before you include the Digital Data Connector design component.

List definition

The list definition is a Web Content Manager content item. It holds the following items:

- Information that is needed to establish the required list-rendering context
- A reference to a list appearance component that is responsible for generating a specific visual design for presenting the data that is contained in the resulting bean list object.

To render an individual list, you render the corresponding list definition content item in the Web Content Viewer portlet. To generate the actual list markup, the presentation template that is used to render the list definition content item proceeds by two steps:

1. First, it writes the `[Plugin:ListRenderingContext]` to establish the list-rendering context.
2. Then, it renders the component reference to the list appearance component.

The rendering flow

Read about the rendering flow of IBM Digital Data Connector (DDC) for WebSphere Portal Express.

The rendering flow is as follows:

1. The Web Content Viewer portlet renders a list definition content item.
2. The Web Content Manager rendering engine retrieves the associated presentation template to render the appropriate content item.
3. The presentation template renders a `[Plugin:ListRenderingContext]` tag. This tag establishes the current list-rendering context.
4. The presentation template renders a component reference that points to a list appearance component.
5. The selection rule that is contained in the list appearance component triggers the bean list creation process in the DDC framework. The framework identifies the target DDC plug-in by inspecting the list-rendering context.
6. The addressed DDC plug-in determines which data to load, and how to filter and sort the data, based on information that is pulled from the provided list-rendering context.
7. The DDC plug-in loads the data and transforms it into a corresponding bean list data structure. The generic XML DDC plug-in implements this step by applying the list-rendering profile that the list-rendering context references to the loaded XML data.
8. The DDC plug-in passes the resulting bean list object back to the list appearance component.
9. The Web Content Manager rendering engine produces the markup for the data that is contained in the bean list. Web Content Manager design components can access the individual attributes of the items in the bean list by using the `[AttributeResource]` tag.

Implementing user interactions

With IBM Digital Data Connector (DDC) for WebSphere Portal Express, you can use IBM Web Content Manager presentation components for visualizing external data. As a further benefit, you can also add user interfaces for creating, modifying,

and deleting external data to your DDC list appearance components. This way, your users can use these features to create, modify, or delete external data items from your lists.

In this scenario, the Web Content Viewer portlet acts as a mediator. It forwards information provided by the user to the specified data sink and sets the result information either as a private render parameter or as a session attribute. You can plug in your own data sinks by implementing the `com.ibm.portal.resolver.data.FormDataDataSink` interface. This interface is defined in the public POC resolution framework API. For more detailed information, refer to the *Package `com.ibm.portal.resolver.data` summary*.

"Sending data to the Web Content Viewer portlet"
You can use IBM Web Content Manager presentation components to create user interfaces for creating, modifying, and deleting external data. With this approach, you use Web Content Manager design components to generate specific HTML form markup.

"Setting dynamic Digital Data Connector filter values" on page 3270
IBM Digital Data Connector (DDC) for WebSphere Portal Express defines a dedicated public render parameter that can be used by DDC plug-ins for filtering lists.

Related information:

Package com.ibm.portal.resolver.data summary

Sending data to the Web Content Viewer portlet

You can use IBM Web Content Manager presentation components to create user interfaces for creating, modifying, and deleting external data. With this approach, you use Web Content Manager design components to generate specific HTML form markup.

The HTML form markup triggers a generic portlet action on the Web Content Viewer portlet. The form can use individual input fields to collect information from your web site users in combination with hidden input fields that provide the interaction context. The interaction context includes a target URI that identifies the actual data sink that receives and processes the information that users post by using your form.

In this scenario, the Web Content Viewer portlet acts as a mediator. It forwards the information that the user typed into the HTML form to the specified target URI and sets the result information either as a private render parameter or as a session attribute. After the user submits the form, the portal renders the page with the updates. The list rendering appearance component can then access the result information by using the session attribute or render parameter plug-in tags.

"Creating the HTML form" on page 3261
You can use IBM Web Content Manager design components to generate the HTML forms for the user interfaces that provide the user interactions.

"Receiving the result of an HTML form submission" on page 3262
The interaction requests that the Web Content Viewer portlet mediates result in response information. To access that result information, use the IBM Web Content Manager rendering plug-ins.

"The generic XML Digital Data Connector data sink" on page 3263
The IBM Digital Data Connector (DDC) for WebSphere Portal Express framework contains a generic data sink. You can use the data sink in combination with HTML forms generated by your IBM Web Content Manager design components.

Related information:

The session attribute plug-in

The render parameter plug-in

Creating the HTML form:

You can use IBM Web Content Manager design components to generate the HTML forms for the user interfaces that provide the user interactions.

About this task

To generate the HTML form markup by using IBM Digital Data Connector (DDC) for WebSphere Portal Express design components, perform the following tasks:

Procedure

1. Set the form `method` attribute to the value `post`.
2. Set the form `enctype` attribute to the value `multipart/form-data`.
3. Set the form `action` attribute to a portlet action URL that addresses the web content viewer and triggers its `post` portlet action.
4. Add a hidden input field with the name `_charset_` as the first input field to the form. This input field defines the character set that the browser uses to encode the form data. Initialize it with the character encoding of the page. To do so, use the following HTML code:

   ```
   <input type="hidden" name="_charset_"
          value="[Plugin:EvaluateEL
          value="${pageContext.response.characterEncoding}"
          compute="once"]"/>
   ```

5. To specify the target URI for the interaction, use a second hidden input field named `action.uri`. The target URI identifies the actual recipient of the data that is posted by this form, which must be a data sink. To do so, use the following HTML code:

   ```
   <input type="hidden" name="action.uri" value="..."/>
   ```

6. Add more hidden or visible form input fields to the form. Use these input fields to assemble all the information that is necessary for the recipient of this form post. You can use the `action.uri` parameter to specify the recipient of this form post operation. Example: To trigger the creation of a forum topic reply, proceed as follows:
   a. Define text input fields for gathering the title and body information from the user.
   b. Add further hidden input fields for identifying the target forum topic. These input fields are typically filled by using a corresponding `[AttributeResource]` tags.

Results

The target URI needs to identify a data sink that accepts multipart data requests. You can use data sinks of the following types:

- You can use a data sink that IBM WebSphere Portal Express provides
- You can use a custom implementation of the `com.ibm.portal.resolver.data.FormDataDataSink` interface. This interface is defined in the public POC resolution framework API. For more detailed information, refer to the *Package `com.ibm.portal.resolver.data` summary*.

Here is an example of an HTML form for sending data to the Web Content Viewer portlet:

```
<form method="post" enctype="multipart/form-data"
      action="[Plugin:ActionURL action="post" copyCurrentParams="true"
      param="resultSessionAttribute=myResult" compute="always"]">
     <input type="hidden" name="_charset_" value="[Plugin:EvaluateEL
            value="${pageContext.response.characterEncoding}" compute="once"]"/>
     <input type="hidden" name="action.uri" value="$DATA_SINK_URI"/>
     <input type="hidden" name="myAction" value="createComment"/>
     <input type="hidden" name="myResourceId"
            value="[AttributeResource attributename="id"]"/>
     My Text: <input type="text" name="myTextField"/><br/>
     <input type="submit" value="Submit"/>
</form>
```

Replace the variable `$DATA_SINK_URI` by the appropriate value for your environment.

Related information:

Package com.ibm.portal.resolver.data summary

Receiving the result of an HTML form submission:

The interaction requests that the Web Content Viewer portlet mediates result in response information. To access that result information, use the IBM Web Content Manager rendering plug-ins.

About this task

When you post data to the Web Content Viewer portlet as described under *Sending data to the Web Content Viewer portlet*, the addressed data sink serves the result back in the form of a data source. The Web Content Viewer portlet performs the following operations on that data source:

1. The portlet transforms that data source into a String value by converting the source into a character source.
2. The portlet sets the resulting String value as a corresponding session attribute or private render parameter.

The URL generated for the action parameter of the HTTP form controls whether the resulting string is stored in a session attribute or set as a private render parameter. You can specify the behavior by setting the corresponding `resultSessionAttribute` and `resultRenderParameter` parameters in the action URL plug-in tag that you use to generate the form action URL.

Results

You can then access the session attribute and the render parameter in your web content by using the session attribute rendering plug-in and the render parameter rendering plug-in, respectively. For example, if the data source of a data sink returns JSON data, you can process the data further by using JavaScript that is contained in your web content.

Related information:

The action URL plug-in

The session attribute plug-in

The render parameter plug-in

The generic XML Digital Data Connector data sink:

The IBM Digital Data Connector (DDC) for WebSphere Portal Express framework contains a generic data sink. You can use the data sink in combination with HTML forms generated by your IBM Web Content Manager design components.

The data sink plugs into the POC resolver framework. You can address it through the `action.uri` parameter that you defined in your DDC HTML forms.

The data sink supports simple HTTP PUT, POST, and DELETE operations for XML-based remote APIs. This approach follows the Representation State Transfer (REST) design pattern as used in many existing remote APIs. You can use the data sink to transform form post parameters into XML documents that you can then send to remote service end points. You can control the transformation by using list-rendering profiles. You address the generic XML DDC data sink by using the following URI: `ddc:operation:blp:ibm.portal.ddc.xml`.

The result data that the addressed service returns is transformed into a generic JSON object. You can access this object in the Web Content Manager design components after the page refresh that is triggered by the form post.

"Parameters that the generic XML data sink supports"
The generic XML IBM Digital Data Connector (DDC) for WebSphere Portal Express data sink supports the parameters that are listed here.

"The generic XML data sink result object" on page 3269
The IBM Digital Data Connector (DDC) for WebSphere Portal Express data sink returns a JSON string. The JSON string contains the result information for the outbound interaction call that was performed.

Related tasks:

"Creating the HTML form" on page 3261
You can use IBM Web Content Manager design components to generate the HTML forms for the user interfaces that provide the user interactions.

Parameters that the generic XML data sink supports:

The generic XML IBM Digital Data Connector (DDC) for WebSphere Portal Express data sink supports the parameters that are listed here.

`ddc.method`
Use this parameter to identify the HTTP method that you want to be used for sending the data to the outbound target. Supported values are `put`, `post`, and `delete`.

`ddc.uri.target`
Use this parameter to specify a URI that identifies the target of the outbound request. The target can be an HTTP or HTTPs URL, for example, to a remote REST service.

`ddc.uri.template`
Use this parameter to specify a URI that identifies the template XML document that is required for `set`, `addChild`, `addPredecessor`, and `addSuccessor` actions. This document is the starting point for creating the final XML document that is sent to the target service. You can modify the template for starting a specific operation by using form post parameters. The URI used in this parameter needs to identify a well-formed XML document.

`ddc.profile.in`
Use this parameter to define the name of the list-rendering profile that you

want to apply to the template XML document. You can use the item attribute definitions and corresponding XPath statements of this profile to modify specific fragments of the template XML document. This modification is done before the resulting XML document is sent to the target service.

ddc.profile.out
Use this parameter to define the name of the list-rendering profile that you want to apply to the data returned by the target service. You can use the item attribute definitions and corresponding XPath statements of this profile to extract specific XML fragments of XML documents that are served in response to the update operation. If you do not set this parameter, the data sink uses the same profile as the one that the `ddc.profile.in` parameter references.

ddc.itemattribute.value.*suffix*
Use this parameter to specify the value that you want to use for modifying the template XML document that is required for `set`, `addChild`, `addPredecessor`, and `addSuccessor` actions. The value can come from user input or can be specified through hidden form input field that is filled by IBM Web Content Manager tags. For example, such a tag can be the `[AttributeResource]` tag.

Depending on the value of the `valueType` definition of the related operation, specify one of the following values:

- A URI that identifies the item attribute value for the operation. This URI must be a URI that the POC resolver framework can resolve.
- Plain text that represents the item attribute value for the operation.

The *suffix* value does not imply any semantics. Use it to correlate this attribute value specification with an item attribute operation specification. Both specifications must be represented by a `ddc.itemattribute.value.`*suffix* and `ddc.itemattribute.operation.`*suffix* parameter, respectively. For example, the following attribute value specification is correlated with the following item attribute operation specification by using the common suffix `xyz`:
`addc.itemattribute.value.xyz` and `ddc.itemattribute.operation.xyz`.

ddc.itemattribute.template.suffix
Use this parameter to specify an optional item attribute value template. The generic XML DDC data sink data sink replaces all occurrences of the `${value}` string in the template with the value of the related `ddc.itemattribute.value.`*suffix* parameter. This step creates the final value to use for modifying the template XML document that is used for `set`, `addChild`, `addPredecessor`, and `addSuccessor` actions.

If the related `ddc.itemattribute.value.`*suffix* defines more than one value by using a separator specified by the `valueSeparator` parameter, the generic data sink applies each individual value to the template.

Depending on the value of the `templateType` definition of the related operation, specify one of the following values:

- A URI that identifies the item attribute value template for the operation. This URI must be a URI that the POC resolver framework can resolve.
- Plain text that represents the template for the operation.

The *suffix* value does not imply any semantics. Use it to correlate this attribute value with a corresponding item attribute value specification and item attribute operation specification. Both specifications must be represented by a `ddc.itemattribute.value.`*suffix* and `ddc.itemattribute.operation.`*suffix* parameter, respectively. For example, the following attribute value specification is correlated with the following item attribute operation specification by using

the common suffix xyz:
addc.itemattribute.value.xyz and ddc.itemattribute.operation.xyz.

ddc.itemattribute.operation.*suffix*
Use this parameter to specify a modification of the template document. The value for this parameter is comma-separated list of name-value pairs. You can specify the following keys:

action
Use this key to identify the action that you want to be completed on the template document. The data sink completes the following actions on the template XML document. The node selection is based on the itemAttribtueName value and the list-rendering profile that is attached through the ddc.profile.in parameter. You can specify the following values:

set
Use this value to modify the node selected by the itemAttributeName value.

addChild
Use this value to add a child node selected by the itemAttributeName value.

addPredecessor
Use this value to add a predecessor sibling to the node selected by the itemAttributeName value.

addSuccessor
Use this value to add a successor sibling to the node selected by the itemAttributeName value.

remove
Use this value to remove the selected node.

In contrast to the actions listed previously, the get action is not performed on the template document, but on the resulting XML document that the addressed service returns. If you use this action, the node selection is based on the itemAttribtueName value and the list-rendering profile that is attached through the ddc.profile.out parameter.

get
Use this value to add a corresponding entry to the result JSON object that the data sink generates. The entry holds the string value of the selected node in the result XML document.

itemAttributeName
Use this key to reference an individual item attribute definition in the associated list-rendering profile. The XPath that is associated with this item attribute definition is used to select the target node for this operation. In this context, you can use only ItemAttribute definitions. The following definitions are not supported: AssociatedItemAttribute, ComputeItemAttribute, ConstructedItemAttributes.

templateType
Use this key to specify the type of the value of the ddc.uri.template parameter that was listed earlier. You can specify the following values:

text
If you specify the value text, the value of the

ddc.itemattribute.template.*suffix* parameter is used as a normal string that represents the item attribute template. This is the default template type.

uri
If you specify the value uri, the data sink parses the value of the ddc.itemattribute.template.*suffix* parameter as a URI. The data sink determines the actual value by reading the data that is served from this URI. The URI needs to be a URI that the POC resolver framework can resolve.

valueType
Use this key to specify the type of the value of the ddc.itemattribute.value.*suffix* parameter that is related to the operation. You can specify one of the following values:

text
If you specify the value text, the value of the ddc.itemattribute.value.*suffix* parameter is used as a normal string. This is the default value type.

uri
If you specify the value uri, the data sink parses value of the ddc.itemattribute.value.*suffix* parameter as a URI. The data sink determines the actual value by reading the data that is served from this URI. The URI must be a URI that the POC resolver framework can resolve.

valueSeparator
Use this key to enable the operation to support multiple values. The generic data sink computes the individual values by splitting the correlated parameter value using the separator that this key specifies. If the separator consists of multiple characters, each character is treated as a separate separator. Example: To split an input value on all occurrences of commas (**,**) and semicolons (**;**), you combine both separators in the value for this parameter as follows: **;,**. The data sink completes the operation for each fragment of the separated value. Use this parameter only for the following actions: addChild, addPredecessor, addSuccessor.

ddc.state.operation.suffix
Use this parameter to specify a modification of the portlet render state. The value for this parameter is comma-separated list of name-value pairs. You can specify the following keys:

action
Use this key to identify the action that you want to be completed on the portlet render state. You can use the following values:

set
Use this value to set a private render parameter.

add
Use this value to add a private render parameter.

remove
Use this value to remove a private render parameter.

renderParameterName
Use this key to specify the name of the private render parameter that you want to modify.

condition
Use this key to specify the condition under which you want this operation to be performed. You can use the following values:

success
If you specify this value, the modification to the portlet render state is performed only if the overall data sink operation succeeded.

error
If you specify this value, the modification to the portlet render state is performed only if the overall data sink operation resulted in an error.

always
If you specify this value, the modification to the portlet render state is performed independent of whether the overall data sink result succeeded or resulted in an error.

ddc.state.value.suffix
Use this parameter to specify the value for the state operation as described in the corresponding `ddc.state.operation` parameter.

ddc.passback.*name*
Use this parameter to specify custom pass-back parameters. The generic XML DDC data sink does not analyze these parameters. It only adds them to the interaction result JSON object as a `passbackData` member using `name` as an identifier. This parameter can be useful for retaining specific state information after the operation is completed, for example when you create a new resource.

ddc.resultheaders
Use this parameter to specify the names of HTTP response header fields that you want to be included in the result object. The HTTP response header fields refer to the outbound call of the generic XML DDC data sink. To specify more than one header field, use multiple `ddc.resultheaders` parameters. After processing the response from the remote REST service, the data sink adds the specified header fields from the response to the interaction result JSON object as a `resultHeaderData` member. You can use this parameter to access information from the outbound call that the data sink makes.

Example: Creating a resource

Typical XML REST services support the creation of new resources by sending a corresponding resource XML document to the service. To do so, the client typically uses an HTTP POST request. The following HTML fragment shows an HTML form that you can use to create a comment on a Connections blog post:

```
<form id="[AttributeResource attributeName='id']CreateBlogPostComment"
      method="POST" enctype="multipart/form-data"
      action="[Plugin:ActionURL copyCurrentParams="true"
      param="resultSessionAttribute=myResult" compute="always"]">
  <input type="hidden" name="_charset_" value="[Plugin:EvaluateEL
         value="${pageContext.response.characterEncoding}" compute="once"]"/>
  <input type="hidden" name="action.uri" value="ddc:operation:blp:ibm.portal.ddc.xml"/>
  <input type="hidden" name="ddc.uri.template"
         value="wcmrest:LibraryHTMLComponent/[Component
         name="yourLibrary/interaction
         templates/create comment" format="id"]"/>
  <input type="hidden" name="ddc.profile.in"
         value="ibm.portal.sr.blogs.post.comments"/>
  <input type="hidden" name="ddc.itemattribute.operation.addComment"
         value="action=set,itemAttributeName=body"/>
  <textarea title="Content" name="ddc.itemattribute.value.addComment"
            placeholder=" "></textarea>
  <input type="hidden" name="ddc.uri.target" value="[AttributeResource
```

```
        attributeName="rawCommentsEditLink"]"/>
  <input type="hidden" name="ddc.method" value="post"/>
  <input type="submit" class="lotusBtn" value="Submit" name="submitButton"/>
</form>
```

The form sets the `action`, `method`, and `enctype` parameters. The form also specifies the `_charset_` and `action.uri` input fields. For more information about these fields, read *Creating the HTML form*. Additionally, the form specifies the following behavior:

- The form sets the `action.uri` parameter to the value `ddc:operation:blp:ibm.portal.ddc.xml`. As a result, the generic XML Digital Data Connector data sink processes the form.
- The template document that represents the blog post comment XML entry is referenced by using the `ddc.uri.template` parameter. In this example, the template document is served from a Web Content Manager HTML component that is referenced by a corresponding `wcmrest` URI. For more information about the Web Content Manager REST URI format, read *REST service for Web Content Manager* in the Web Content Manager documentation. In this example, the referenced HTML component contains the following XML fragment:

  ```
  <?xml version='1.0' encoding='UTF-8'?>
  <entry xmlns='http://www.w3.org/2005/Atom'
  xmlns:thr="http://purl.org/syndication/thread/1.0">
     <content type='html'>
     </content>
  </entry>
  ```
- The profile for this template document is identified by using the `ddc.profile.in` parameter. As the example posts a blog post comment document, the profile `ibm.portal.sr.blogs.post.comments` is used.
- The parameters `ddc.itemattribute.operation.addComment` and `ddc.itemattribute.value.addComment` define the update of the `<content>` element in the template document. The operation parameter is set to the value `action=set,itemAttributeName=body`. As a result, the value that the user entered in the `ddc.itemattribute.value.addComment` text area is used to update the XML node that the item attribute named `body` selected. The referenced profile holds the information about where to locate the `<content>` element in the template.
- To create the blog post comment, the modified template document needs to be sent to the blog post edit link. To do so, set the `ddc.method` parameter to the value `post`. To extract the blog post edit link from the current DDC list-rendering context, use the tag `[AttributeResource attributeName="rawCommentsEditLink"]`.

Example: Deleting an existing resource

Typical XML REST services support the deletion of existing resources by sending a corresponding an HTTP DELETE request to the resource URL. The following HTML fragment shows an HTML form that you can use to delete a comment from an IBM Connections blog post:

```
<form id="[AttributeResource attributeName="id"
      separator=","]DeleteBlogComment" method="POST"
      enctype="multipart/form-data" action="[Plugin:ActionURL
      action="post" copyCurrentParams="true"
      param="resultSessionAttribute=myResult" compute="always"]">
    <input type="hidden" name="_charset_" value="[Plugin:EvaluateEL
           value="${pageContext.response.characterEncoding}" compute="once"]"/>
    <input type="hidden" name="action.uri"
           value="ddc:operation:blp:ibm.portal.ddc.xml"/>
    <input type="hidden" name="ddc.method" value="delete"/>
```

```
    <input type="hidden" name="ddc.uri.target"
           value="[AttributeResource attributeName="rawEditLink"
                   separator=","]" />
</form>
```

The form specifies the following behavior:

- The form sets the `uri` parameter to the value `ddc:operation:blp:ibm.portal.ddc.xml`. As a result, the generic XML IBM Digital Data Connector (DDC) for WebSphere Portal Express data sink processes the form.
- To delete the resource, you do not need to send XML data to the data sink. Therefore, the form does not specify the parameters `ddc.uri.template` or `ddc.profile.in` or any of the `ddc.itemattribute` parameters.
- To delete the blog post comment, you need to send an HTTP delete request to the blog post edit link. To do so, set the `ddc.method` parameter to the value `delete`. To extract the blog post edit link from the current DDC list-rendering context, use the tag `[AttributeResource attributeName="rawCommentsEditLink"]`.

Related concepts:

"REST service for Web Content Manager" on page 3306
Application developers can use Representational State Transfer (REST) services to work with Web Content Manager. The REST service for Web Content Manager provides authoring access to content items and elements. The service follows the Atom Publication Protocol, and Atom feeds, and entries are accessible in XML (application/atom+xml) and JSON (application/json) format.

"How to use REST with content items" on page 3348
You can use the Web Content Manager REST service to create, read, update, and delete content items.

Related tasks:

"Creating the HTML form" on page 3261
You can use IBM Web Content Manager design components to generate the HTML forms for the user interfaces that provide the user interactions.

The generic XML data sink result object:

The IBM Digital Data Connector (DDC) for WebSphere Portal Express data sink returns a JSON string. The JSON string contains the result information for the outbound interaction call that was performed.

The JSON object makes the following data available:

`status`
: This member provides the return status of the interaction. If the target service responded to the outbound request with an error response, the status is set to the value `error`. Otherwise, the value is set to `success`.

`httpStatusCode`
: This member provides the HTTP status code that is returned for the HTTP outbound call by the remote REST service. Designers can use it to distinguish between the following types of operations:
  - Operations that are not available, for example because the item to operate on was not found
  - Operations that are not permitted, because the user does not have sufficient privileges on the remote service.

message
This member returns the error message in case of an error.

resultAttributeData
This JSON member contains a JSON object with the name-value pairs of attributes that the data sink reads from the XML document that the target service returns. It contains the values for all attributes that the corresponding `ddc.itemattribute.operation` parameter references by using the action value `get`.

passbackData
This member contains a JSON object with the name-value pairs for all parameters of the processed operation that you specified by using the `ddc.passback.`*`name`* parameters.

resultHeaderData
This member contains a JSON object with the name-value pairs that represent the HTTP response header field names and values that you specified by using `ddc.resultheaders` parameters for the processed operation. The HTTP response header values refer to the response of the remote REST service.

Setting dynamic Digital Data Connector filter values

IBM Digital Data Connector (DDC) for WebSphere Portal Express defines a dedicated public render parameter that can be used by DDC plug-ins for filtering lists.

You can set the value of the DDC filter public render parameter by submitting an HTML form that addresses the URI `ddc:filter` by using an HTTP get method. You can create the HTML form by using IBM Web Content Manager design components. The following HTML fragment shows a simple search form:

```
<form action="." method="get">
   <input type="hidden" name="uri" value="ddc:filter" />
   <input type="hidden" name="prefix" value="<your DDC filter prefix>" />
   <input type="hidden" name="action" value="set" />
   <input name="value" type="text" value="Enter keywords" title="Search keywords" />
   <input type="submit" value="Submit" name="submitButton" />
</form>
```

When you define such HTML form, take care of the following aspects:

- Set the form action attribute to a period (`.`) and the form method attribute to `get`.
- Add a form parameter with the name `uri` and set it to the value `ddc:filter`.
- Add a form with the name `filter` and specify the filter value prefix that is recognized by the target DDC plug-in. Using such a prefix enables individual DDC plug-ins to track their filter parameters if multiple plug-ins are active on the same portal page. To determine the appropriate prefix value, read the documentation of the DDC plug-in that you use.
- Add a form parameter with the name `action` and specify the action that you want to be completed. To set the value, specify `set`. To remove the value, specify `remove`.
- Add a form parameter with the name `value` to specify the filter value that you want to be set. With the `remove` action, this parameter is ignored.

After the user submits this form, the DDC filter public render parameter is updated in the user's rendering state, and the portal now renders the page in its new state. The parameter value that is set is concatenated from the values of the `prefix` parameter and of the `value` parameter. You can access the full concatenated value in your Web Content Manager design components by using the `RenderParam`

plug-in. To access the value for a specific prefix, you can use the `ListRenderingContext` plug-in with the action attribute set to `getFilter`.

DDC plug-ins can evaluate the active filter parameter value by using the `com.ibm.portal.wcm.plr.ListRenderState` interface that is defined by the public Digital Data Connector API.

Working with list-rendering profiles

You can create new or derived list-rendering profiles. After you create a list-rendering profile, you deploy it.

"Structure of list-rendering profiles"
A list-rendering profile consists of a set of name-value pairs.

"Creating list-rendering profiles" on page 3272
When you create a list-rendering profile, you can either create a completely new profile or create a derived profile by extending an existing profile. A derived profile includes the item attribute and list property declarations of other list-rendering profiles by referencing these profiles.

"Deploying list-rendering profiles" on page 3272
You can choose between two ways of deploying your custom list-rendering profiles.

Structure of list-rendering profiles

A list-rendering profile consists of a set of name-value pairs.

These name-value pairs define the following aspects:

- The name of the profile in a format so that the profile can be referenced from the `[Plugin:ListRenderingContext]` tag.
- The process of dividing a single XML document into a list of separate items.
- The extraction of individual item attributes from data that is contained in those items. The resulting set of item attribute defines the set of attributes that you can access in your IBM Digital Data Connector (DDC) for WebSphere Portal Express list designs by using the `[AttributeResource attributeName=""]` tag.
- The extraction of list properties from the complete XML document. The resulting set of list properties defines the set of key values that you can use with the `[Plugin:ListRenderingContext action="getListProperty" key=""]` tag.
- The translated item attribute titles that you want to be shown to your site designers when they use the **Insert a tag** dialog in the Web Content Manager authoring portlet. The list of attribute titles helps your site designers choose the correct `attributeName` parameter values for the `[AttributeResource]` tag.

WebSphere Portal Express Version 8.5 supports the following two types of list-rendering profiles:

- An XPath centric type of profile. It uses XPath statements to define the data transformation. This data transformation contains the aspects of dividing the XML document and the extraction of item attributes and list properties as listed before.
- A custom type of profile. It does not define a data transformation, but registers only the set of supported item attributes that are shown in the **Insert a tag** dialog. When you use a custom type profile, the corresponding bean list provider must provide the respective values for the attribute names that you want the custom profile to support.

Creating list-rendering profiles

When you create a list-rendering profile, you can either create a completely new profile or create a derived profile by extending an existing profile. A derived profile includes the item attribute and list property declarations of other list-rendering profiles by referencing these profiles.

About this task

When you create a new or derived list-rendering profile, make sure to define an item attribute that is named `id` and to define a name for your profile.

To define a derived profile, use the `Extends` key.

When you model your derived profile from an existing profile, you reference the other profile by specifying its profile name. When you reference another profile, the plug-in imports all `ItemAttribute`, `AssociatedItemAttribute`, `ConstructedItemAttribute`, `ComputedItemAttribute`, and `ListProperty` and XML namespace declarations into the current profile. To include multiple profiles, specify a comma-separated list of profile names. You can overwrite individual imported properties in the profiles. When you create a derived profile, you modify existing item attributes or list property declarations of the existing profile or add more item attribute or list property declarations. To do so, you can extend your profile from the existing profile and add or overwrite the item attribute and list property declarations of that profile as required. The following conditions apply:

- All changes to profiles from which the extended profile is derived are effective on the extended profile if the extended profile does not explicitly overwrite these declarations.
- Your profile can extend from any other profile, independent of how you deployed the individual profiles. But make sure that you do not create any dependency cycles.
- The profiles override each other in the order in which they occur in the comma-separated list. A profile that occurs earlier in the sequence overrides subsequent ones.

  Example: To include the item attribute and list property declarations of profiles that are named `yourCo.Atom` and `yourCo.Base`, specify `xyz.Extends=yourCo.Atom,yourCo.Base`.

  Note: If your `yourCo.Atom` and `yourCo.Base` define the same attribute, the definition in the `yourCo.Atom` profile overrides the definition that is given in the `yourCo.Base` profile.

Related information:

The action URL plug-in

Deploying list-rendering profiles

You can choose between two ways of deploying your custom list-rendering profiles.

About this task

- You can deploy a list-rendering profile by creating all entries that are contained in the list-rendering profile as custom properties in the WP List Rendering Profile Service resource environment provider in the WebSphere Integrated Solutions Console.
- You can implement a plug-in for the extension point with the ID `com.ibm.portal.wcm.plr.ListRenderingProfileProvider`. These extensions can

make multiple list-rendering profiles available to the IBM Digital Data Connector (DDC) for WebSphere Portal Express run time through the following method:

```
com.ibm.portal.wcm.plr.ListRenderingProfileProvider.getListRenderingProfiles()
```

This method is defined in the public Java API for the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework. With this approach, your plug-in returns the individual list-rendering profile entries as properties contained in `java.util.Properties` objects. By using this approach, you can package the required list-rendering profiles together with the DDC plug-in.

List-rendering profiles are cached by the DDC framework. After deploying a list-rendering profile, you must make the framework aware of changes to the list-rendering profile. To do so, either restart the portal or render the IBM Web Content Manager tag `[Plugin:ListRenderingContext action="reloadProfiles"]`.

Integrating remote XML data

The IBM Digital Data Connector (DDC) for WebSphere Portal Express framework provides a generic XML DDC plug-in that is ready to use for integrating external XML data of your choice. You can use this plug-in to render external XML data on your portal pages without having to write custom Java code.

About this task

The generic XML DDC plug-in supports the concept of list-rendering profiles. It therefore makes it possible to integrate XML data. The XML data can be served in arbitrary XML document formats. A list-rendering profile describes the transformation between a specific XML document format and the generic bean list data structure that the DDC framework supports. In addition, the list-rendering profile enumerates all attribute names of the items in the resulting bean list that you can add to the result markup. This way, the IBM Web Content Manager **Insert Tag** user interface can present a dedicated selection box. That selection box contains all supported attributes for a specific list-rendering profile when you author DDC list appearance components.

"The generic XML Digital Data Connector plug-in" on page 3274
You can use the IBM Digital Data Connector (DDC) for WebSphere Portal Express plug-in directly to integrate remote XML data without coding. It makes it possible to integrate XML data that is served in arbitrary XML document formats by supporting the concept of list-rendering profiles.

"Syntax for XPath based list-rendering profiles" on page 3275
An XPath based list-rendering profile contains a set of name-value pairs called entries. This set of entries defines the set of available list properties and item attributes that are available for transforming external data into bean lists.

"Syntax for custom list-rendering profiles" on page 3283
Custom IBM Digital Data Connector (DDC) for WebSphere Portal Express plug-ins that do not delegate the initial bean list computation to the generic XML DDC plug-in can use a custom list-rendering profile. This case typically occurs when you integrate non-XML data.

"The Atom list-rendering profile" on page 3284
IBM Digital Data Connector (DDC) for WebSphere Portal Express provides a list-rendering profile that is ready to use for access to feeds that comply with the Atom syndication format standard.

The generic XML Digital Data Connector plug-in

You can use the IBM Digital Data Connector (DDC) for WebSphere Portal Express plug-in directly to integrate remote XML data without coding. It makes it possible to integrate XML data that is served in arbitrary XML document formats by supporting the concept of list-rendering profiles.

A list-rendering profile describes the transformation between a specific XML document format and the generic bean list data structure that this DDC plug-in generates. You identify the generic XML DDC plug-in by its name `ibm.portal.ddc.xml`. You can address it by using the `[Plugin:ListRenderingContext extension-id="ibm.portal.ddc.xml"]` tag. The generic XML DDC plug-in supports the attributes that are shown in the following list. You can specify these attributes when you set up the list-rendering context by using the `ListRenderingContext` tag that IBM Web Content Manager provides.

source
: You can set this attribute to identify the source URI that serves the XML data. The URI must be accessible through the portal resource addressability framework. Supported URI schemes include `http`, `https`, and `dav`. If you access external XML data by using `http` or `https` URLs, make sure to adjust your outbound HTTP connection to allow outbound access to these URLs. For more information, read the information about proxy configuration in the IBM WebSphere Portal Express documentation. For more information, read *Outbound HTTP connection*. Example:

```
[Plugin:ListRenderingContext action="set" extension-id="ibm.portal.ddc.xml"
   profile="ibm.portal.atom"
   attribute="source=https://www.ibm.com/connections/communities/service/atom/catalog/public"
   compute="always"
```

sortCriteria
: Set this attribute to sort the content of a bean list that is based on a specific attribute. This sorting attribute must be defined in the list-rendering profile that the current list-rendering context uses. This built-in sorting capability sorts only the data that is contained in the current bean list. Therefore, do not use this attribute if the bean list represents only a fragment of a larger result set. In such a situation, provide the sorting process by the backend service. If you do not specify this attribute, the bean list maintains the sequence of items as served by the external data source. To specify the sort order, use the `sortOrder` attribute.

sortOrder
: Use this attribute in combination with the `sortCriteria` attribute to specify the sort order. Supported values are `asc` and `desc` for sorting in ascending or descending order. Example:

```
[Plugin:ListRenderingContext action="set" extension-id="ibm.portal.ddc.xml"
   profile="ibm.portal.atom"
   attribute="source=https://www.ibm.com/connections/communities/service/atom/catalog/public"
   attribute="sortCriteria=title" attribute="sortOrder=asc"]
```

cacheScope
: Set this attribute to identify the cache scope for bean lists that result from a specific list-rendering context. Supported values are as follows:

  public
  : This value is the default value. If you set this attribute to `public`, the cached bean list values are shared across users.

  private
  : If you set this attribute to `private`, individual bean list objects are cached per user. Specify `private` if the bean lists contain user-specific information.

Example:

```
[Plugin:ListRenderingContext action="set" extension-id="ibm.portal.ddc.xml"
   profile="ibm.portal.atom"
   attribute="source=https://www.ibm.com/connections/communities/service/atom/catalog/my"
   attribute="cacheScope=private" compute="always"]
```

cacheType
Set this attribute to identify the cache type that you want to be used for bean lists that result from a specific list-rendering context. Supported values are as follows:

invalidateOnLogin
If you use this setting, cached bean list objects are invalidated if the user for whom the bean list was computed logs in to portal. Use this value only in combination with the setting `cacheScope=private`.

default
If you use this setting, resulting cached bean lists are not invalidated during user login.

invalidateCache
Use this attribute to invalidate the bean list. If you set this attribute to the value `always`, the bean list is always invalidated before the new list-rendering context is set up. In other words, the bean list is recomputed for each rendering. To achieve a conditional invalidation, you can set the attribute to a different value. In this case, the cache is invalidated only if a portlet session attribute or a request attribute with the specified value is available in the current execution context.

clearCache
Use this attribute to clear the bean list cache. If you set this attribute to the value `always`, the cache is always cleared before the new list-rendering context is set up. To achieve a conditional clearing action, you can set the attribute to a different value. In this case, the cache is cleared only if a portlet session attribute or a request attribute with the specified value is available in the current execution context.

Related concepts:

"Outbound HTTP connection" on page 2984
Applications in your IBM WebSphere Portal Express and the related user activities can require outbound HTTP connections to remote computer systems. The outbound HTTP connection service provides an administration infrastructure with a central point of administration for all outbound HTTP connections that are defined in the portal environment.

Syntax for XPath based list-rendering profiles

An XPath based list-rendering profile contains a set of name-value pairs called entries. This set of entries defines the set of available list properties and item attributes that are available for transforming external data into bean lists.

The following list of list-rendering profile entries show a sample XPath based list-rendering profile that you can use for transforming Atom feed documents. For more information, read *The Atom Syndication Format*.

```
sample_atom.BeanListProviderID=ibm.portal.ddc.xml
sample_atom.NamespaceMapping.atom=http://www.w3.org/2005/Atom

sample_atom.ListItemSelection=//atom:entry

# Item Attribute Declarations:
sample_atom.ItemAttribute.id=./atom:id
sample_atom.ItemAttribute.title=./atom:title
```

```
sample_atom.ItemAttribute.summary=./atom:summary
sample_atom.ItemAttribute.subtitle=./atom:subtitle
sample_atom.ItemAttribute.authorName=./atom:author/atom:name
sample_atom.ItemAttribute.authorEmail=./atom:author/atom:email
sample_atom.ItemAttribute.authorUri=./atom:author/atom:uri
sample_atom.ItemAttribute.contributorName=./atom:contributor/atom:name
sample_atom.ItemAttribute.contributorEmail=./atom:contributor/atom:email
sample_atom.ItemAttribute.contributorUri=./atom:contributor/atom:uri
sample_atom.ItemAttribute.updated=./atom:updated
sample_atom.ItemAttribute.updated.Type=Date
sample_atom.ItemAttribute.published=./atom:published
sample_atom.ItemAttribute.published.Type=Date
sample_atom.ItemAttribute.content=./atom:content
sample_atom.ItemAttribute.categoryTerms=./atom:category/@term
sample_atom.ItemAttribute.categorySchemes=./atom:category/@scheme
sample_atom.ItemAttribute.categoryLabels=./atom:category/@label
sample_atom.ItemAttribute.selfLink=./atom:link[@rel='self']/@href
sample_atom.ItemAttribute.enclosureLink=./atom:link[@rel='enclosure']/@href
sample_atom.ItemAttribute.alternateLink=./atom:link[@rel='alternate']/@href
sample_atom.ItemAttribute.editLink=./atom:link[@rel='edit']/@href

# List Property Declarations:
sample_atom.ListProperty.author=/atom:feed/atom:author
sample_atom.ListProperty.id=/atom:feed/atom:id
sample_atom.ListProperty.selfLink=/atom:feed/atom:link[@rel='self']/@href
sample_atom.ListProperty.title=/atom:feed/atom:title
sample_atom.ListProperty.updated=/atom:feed/atom:updated
sample_atom.ListProperty.updated.Type=Date
```

The names that you use in the list-rendering profile entries must adhere to a specific syntax to be parsed correctly by the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework.

The format for the entry names is *profile-id*.*key*[.*name*[.*meta-data-key*]]. The meanings of the portions are described in the following list:

profile.id
This key specifies an internal ID. It is used only to correlate the entries that belong to the same profile.

key
This key identifies the list-rendering profile aspect that is affected by this entry. Valid values are `Name`, `BeanListProviderID`, `ResourceBundleBaseName`, `NamespaceMapping`, `ListItemSelection`, `ItemAttribute`, `AssociatedItemAttribute`, `ComputedItemAttribute`, `ConstructedItemAttribute`, `ListProperty`, `ComputedListProperty`, `Extends`, `ShowInAuthoringUI`, `Shared`, `Escape`.

name
This key specifies the name of the artifact that is to be defined, for example the attribute name. Names must not contain a period (`.`).

meta-data-key
Some list-rendering profiles entries support metadata that can be associated by using corresponding metadata keys. Valid values are `Type`, `Default`, `Format`, `Depends`, `ShowInAuthoringUI`.

The set of valid values for a list-rendering profile entry depend on the key that is used in the entry name. For more information, read the following topics.

"XPath list-rendering profile keys"
The following list shows the set of list-rendering profile entry keys that are available in the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework.

"XPath list-rendering profile metadata keys" on page 3281
Some item attribute and list property declarations support metadata that can be associated by using corresponding metadata keys.

Related information:

The Atom Syndication Format

XPath list-rendering profile keys:

The following list shows the set of list-rendering profile entry keys that are available in the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework.

`Name`
This value specifies the name of this list-rendering profile. Specifying the name is mandatory for each rendering profile. You can reference this value through the profile attribute of the `[Plugin:ListRenderingContext]` tag and through the `Extends` list-rendering profile entry of other list-rendering profiles. The `Extends` list-rendering profile entry is described later in this list. If this profile does not reference a resource bundle through the `ResourceBundleBaseName` key, this name also shows up in the **Insert a tag** dialog of the IBM Web Content Manager authoring user interface. The name must be unique among all list-rendering profiles in the system.

Example: To set the name of the profile that is identified by profile ID `xyz` to `yourCo.ProductList`, specify `xyz.name=yourCo.ProductList`.

`BeanListProviderID`
Specifying the ID for the DDC plug-in is mandatory for each list-rendering profile. It specifies the extension ID of the DDC plug-in that is used with this profile. The DDC plug-in also gets the calls for computing all computed item attributes. Computed item attributes are described later in this list.

Example: To use the generic XML DDC plug-in, specify `xyz.BeanListProviderID=ibm.portal.ddc.xml`.

`ResourceBundleBaseName`
The resource bundle base name is optional. It specifies the base name of the Java resource bundle that serves the translated strings for the name of this profile and all item attributes that are defined in this profile. The translated strings include associated, computed, and constructed item attributes. The resource bundle base name must identify a Java resource bundle that is available in the portal class path.

Example: To use the resource bundle `yourCo.Bundle` for the profile that is identified by profile ID `xyz`, you specify `xyz.ResourceBundleBaseName=yourCo.Bundle`

`ShowInAuthoringUI`
Valid values are `true` and `false`. If you set this attribute to `false`, the list-rendering profile does not show up in the **Select the target component for this tag** selection list of the **Insert a Tag** dialog of the Web Content Manager authoring user interface.

`NamespaceMapping`
The namespace mapping is optional. It specifies an individual XML namespace

mapping that you want to be used when you evaluate XPath expressions on XML documents that are transformed with this profile. For the name, specify the local name of the namespace mapping. For the value, specify the associated XML namespace URI. The namespace mapping is ignored by profiles of type Custom.

Example: To map the XML namespace URI `http://www.w3.org/2005/Atom` to the local name `atom` in the profile that is identified by the profile ID `xyz`, specify `xyz.NamespaceMapping.atom=http://www.w3.org/2005/Atom`.

`ListItemSelection`

Specifying the list item selection is mandatory for profiles of type XPath. It defines the XPath statement that is used to break down the source XML document into a list of individual XML fragments. Each fragment serves the data for a corresponding item in the resulting bean list. The DDC plug-in evaluates the XPath statement against the root element of the source XML document. For the value, specify a valid XPath statement. Profiles of type Custom ignore the list item selection.

Example: To turn all entries in an Atom feed into list items in the profile that is identified by the profile ID `xyz`, specify `xyz.ListItemSelection=//atom:entry`.

`ItemAttribute`

This key declares an attribute available in all items in the list. The attribute values can then be accessed in your Web Content Manager design components by using the `[AttributeResource attributeName=""]` tag by setting `attributeName` to the name value of the corresponding `ItemAttribute` declaration. For example, to access the value of an item attribute declared with the key `xyz.ItemAttribute.myTitle`, you can use the following Web Content Manager tag `[AttributeResource attributeName="myTitle"]`. Furthermore, declared attributes are listed in the **Select attribute resource tag type** selection box of the **Insert a Tag** dialog of the Web Content Manager authoring user interface when you select the containing profile in the **Select the target component for this tag** selection box.

Important: Each profile must define at least one `ItemAttribute` named `id`. This ID is used to uniquely identify an individual item in the list.

In profiles of type XPath, the value for an item attribute declaration is an XPath statement that is used for extracting the declared attribute value from the XML source document. The DDC plug-in evaluates the XPath statement for each item in the lists relative to the corresponding XML fragments that are identified by using the `ListItemSelection` specification.

Example: To extract the title of the entries in an Atom feed XML document, specify `xyz.ItemAttribute.title=./atom:title`.

`AssociatedItemAttribute`

This key declares an attribute that is available in all items in the list. You can access the attribute values accessed in your Web Content Manager design components in the same way as item attributes. They also appear in the same places in the Web Content Manager authoring user interface. The difference is that in XPath based profiles, you can use associated item attribute declarations to extract data not directly from the source XML document. Instead, you can extract data from a linked XML source document that is referenced by the source XML document. To define an associated item attribute, you must build the value for such an entry by combining a source URL with an XPath statement to select the attribute value from the linked document. You specify the source URL by using the name of the attribute that selects the source URL

from the current bean list. You specify the ItemAttribute reference that provides the document URL by using the pattern {$item-attribute-name}

Example: To obtain the full product description from a list of products, specify as follows:

```
xyz.ItemAttribute.detailsLink=./atom:link[@rel="details"]/@href
xyz.AssociatedItemAttribute.productFullDescription={$detailsLink}//Attribute[@id="desc"]
```

The DDC plug-in computes the values for this attribute in the following sequence:

1. The DDC plug-in evaluates the value of the detailsLink item attribute.
2. The DDC plug-in loads the XML document from that URL.
3. The DDC plug-in evaluates the XPath statement detailsLink on that document.

ConstructedItemAttribute

This key declares an attribute that is available in all items in the list. If you define such an attribute, you can access the attribute values in your Web Content Manager design components in the same way as item attributes. The attribute values also appear in the same places as item attributes in the Web Content Manager authoring user interface. The difference between the two attribute types is as follows: You can build the values for constructed item attributes by combining the values from one or more other item attributes that have a static string. To define a constructed item attribute, you must provide a template string that is used to build the actual attribute values. The template string can contain arbitrary text that is mixed with item attribute references. To specify the ItemAttribute references, you provide the document URL by using the pattern {$item-attribute-name}. Example: To construct an image URL from an image location and image file name item attribute, specify as follows:

```
xyz.ItemAttribute.imageLocation=./imageLocation
xyz.ItemAttribute.imageFileName=./imageName
xyz.ConstructedItemAttribute.imageURL={$imageLocation}/{$imageName}
```

ComputedItemAttribute

This key declares an attribute that is available in all items in the list. You can then access the attribute values in your Web Content Manager design components in the same way as item attributes. They attribute values also appear in the same places as item attributes in the Web Content Manager authoring user interface. The difference between the two types of attribute types is as follows: The values for computed item attributes are not extracted from XML documents but computed based on the item attribute values and other context information. The DDC plug-in that is associated to the list-rendering profile through the BeanListProviderID key computes the attribute values. To serve such data, the DDC plug-in must implement the optional com.ibm.portal.wcm.plr.ComputedAttributeValueProvider Java interface. This interface is defined by the public IBM Digital Data Connector (DDC) for WebSphere Portal Express APIs. The computation is typically done based on the non-computed attribute values in combination with configuration data or other extra context information.

Note: The generic XML DDC plug-in supports no computed item attributes.

To define a computed item attribute, you must declare on which data the attribute value computation depends:

- The computed item attribute value depends only on the other item attribute values. In this case, you must set it to value {$lazy}.

- The computed item attribute value depends on other item attribute values and on the current rendering context that is the current portlet request and response objects. These objects are not cached in the bean list cache, but they are recomputed for each rendering request. If the computed item attribute value depends on the current request context, you must set the value of such a computed item attribute value to the string {$default}.

Example: To define an attribute that represents a stateful portal resource URL that points to the current item, you can declare a computed item attribute such as this one: xyz.ComputedItemAttribute.portalLink={default}.

During run time, the DDC framework then requests the actual values for this attribute. It does so by starting the following method on the specific DDC plug-in plug-in that associated with this list-rendering profile:
com.ibm.portal.wcm.plr.ComputedAttributeValueProvider.getComputedItemAttributeValue()

ListProperty

This key declares a property of the list. You can then access the attribute values in your Web Content Manager design components. You do so by using the [Plugin:ListRenderingContext action="getListProperty" key=""] tag and by setting the key to the name value of the corresponding ListProperty declaration. For example, to access the value of a list property that is declared with the key xyz.ListProperty.myProperty, you use the following Web Content Manager tag: [Plugin:ListRenderingContext action="getListProperty" key="myProperty"] The value for a list property declaration is an XPath statement that is used for extracting the declared attribute value from the XML source document. The plug-in evaluates the XPath statement relative to the root of the source XML document.

Example: To get the link to the next feed page of an Atom feed, specify
xyz.ListProperty.nextLink=./atom:feed/atom:link[@rel=\"next\"]/@href

ComputedListProperty

This key declares a list property of the list. You can then access the attribute values in your Web Content Manager design components in the same way as list properties. The difference between list properties and computed list properties as follows: The values for computed list properties are not extracted from XML documents, but computed based on the item attribute values and other context information. The DDC plug-in that is associated to the list-rendering profile through theBeanListProviderID key computes the property values. To serve such data, the DDC plug-in must implement the optional com.ibm.portal.wcm.plr.ComputedAttributeValueProvider Java interface. This interface is defined by the public APIs of Digital Data Connector APIs. The computation is usually done based on the non-computed list property values in combination with configuration data or other extra context information.

Note: The generic XML DDC plug-in does not support computed list properties. To define a computed list property, you must declare on which of the following two types of data the property value computation depends:

- The computed item attribute value depends only on the other list property values. In this case, you must set it to value {$lazy}.
- The computed list property value depends on other list property values and on the current rendering context, that is the current portlet request and response objects. These objects are not cached in the bean list cache, but they are recomputed for each rendering request. If the computed list property value depends on the current request context, you must set the value of such a computed list property value to the value {$default}.

Extends
Use this key to define a derived profile. A derived profile includes the item attribute, list property declarations, and XML namespace mappings of other list-rendering profiles by referencing these profiles. For more information, read *Creating list-rendering profiles*.

Related tasks:

"Creating list-rendering profiles" on page 3272
When you create a list-rendering profile, you can either create a completely new profile or create a derived profile by extending an existing profile. A derived profile includes the item attribute and list property declarations of other list-rendering profiles by referencing these profiles.

XPath list-rendering profile metadata keys:

Some item attribute and list property declarations support metadata that can be associated by using corresponding metadata keys.

You can use them in combination with `ItemAttribute`, `AssociatedItemAttribute`, `ConstructedItemAttribute`, `ComputedItemAttribute`, and `Listproperty` declarations that are listed under *List-rendering profile keys*. The following list describes the supported metadata key values:

Type
Use this metadata key to specify whether the attribute value is of type `String`, `Date`, `XML`, or `NodeList`. You can format values of type `Date` in flexible ways. You do so by using the format parameter of the `[AttributeResource attributeName="" format=""]` tag. Values of type `XML` are written as serialized XML. Values of type `NodeList` are typically meant to be processed further during the computation of computed item attributes. The IBM Digital Data Connector (DDC) for WebSphere Portal Express plug-in can access the attribute value as an object of type `org.w3c.dom.NodeList`.

Example:
```
xyz.ItemAttribute.updated=./atom:updated
xyz.ItemAttribute.updated.Type=Date
```

Use this metadata key to specify of which type the attribute value is. Four types are available:

String
Values of type `String` are written as plain text.

Boolean
If values of type Boolean are available, they are returned as `true` or `false`.

Date
You can format values of type `Date` in flexible ways. You do so by using the format parameter of the `[AttributeResource attributeName="" format=""]` tag.

XML
Values of type `XML` are written as serialized XML.

NodeList
Values of type `NodeList` are typically meant to be processed further during the computation of computed item attributes. The DDC plug-in can access the attribute value as an object of type `org.w3c.dom.NodeList`.

Example:
```
xyz.ItemAttribute.updated=./atom:updated
xyz.ItemAttribute.updated.Type=Date
```

Default

Use this metadata key to specify a static default value that you want to be returned if the appropriate value cannot be retrieved. Example:

```
xyz.ItemAttribute.rating=./avgRating
xyz.ItemAttribute.rating.default=0.5
```

Format

Use this metadata key to specify the format that you want to use to parse an XML fragment into a date object. Example:

```
xyz.ItemAttribute.updated=./atom:updated
xyz.ItemAttribute.updated.Type=Date
xyz.ItemAttribute.updated.Format=yyyy-MM-dd'T'HH:mm:ss.SSS'Z'
```

FormatLocale

This metadata key is available starting with IBM WebSphere Portal Express V 8.5 Cumulative Fix 02. Use this metadata key to specify the locale that you want to use for parsing an XML fragment into a date object. Example:

```
xyz.ItemAttribute.updated=./atom:updated
xyz.ItemAttribute.updated.Type=Date
xyz.ItemAttribute.updated.Format=yyyy-MM-dd'T'HH:mm:ss.SSS'Z'
xyz.ItemAttribute.updated.FormatLocale=de
```

If you do not specify this metadata, the locale `en` is used by default. You can change the system-wide default locale value by specifying a custom property named `ddc.date.format.default.locale` in the portal Configuration Service. To do so, access the WebSphere Integrated Solutions Console, select the WP Config Service resource environment provider, and set the property to the required locale.

Depends

To enforce a specific computation order for the non-lazy computed item attributes and list properties in the current profile, set this metadata on the `ComputedItemAttribute` and `ComputedListProperty`. Such a computation order can be required if specific computed attributes must access the value of other computed attributed values. In this case, the attribute that requires the value of another computed attributed must list the other attribute in the `Depends` metadata key. If the attribute depends on multiple other computed attributes, you can list multiple attribute names as a comma-separated list. Example: You defined computed attributes named `link` and `proxiedLink`. You want to make sure that the value for the `link` attribute is computed before the value for the `proxiedLink` attribute. In this case, specify as follows:

```
xyz.ItemAttribute.link={computed}
xyz.ItemAttribute.proxiedLink={computed}
xyz.ItemAttribute.proxiedLink.depends=link
```

Note: You do not have to flag dependencies on non-computed attributes. Non-computed attributes are always extracted from the XML before the computed attributes are resolved.

Escape

Use this metadata to define the escaping that you want applied when you write the values of this attribute. Specify one of the values `xml`, `json`, `javascript`, or `none`. The default value is `none`.

Shared

Use this metadata in context of `AssociatedItemAttribute` declarations to define the cache scope for the associated XML document. If this document can be shared among multiple users, you can set this metadata to the value `true`. If

the associated document can contain different data for different users, cache it per user. In this case, you can leave this metadata undefined or explicitly set it to `false`.

Note: Associated documents are cached in the `com.ibm.workplace.wcm.pzn.plr.xml.DocumentCache`. Associated documents that are cached in scope `Shared` are not automatically invalidated during user login.

CF03 Processors

Use this metadata on an `ItemAttribute`, `AssociatedItemAttribute`, `ConstructedItemAttribute`, or `ComputedItemAttribute` to specify one or multiple IDs of attribute value processor plug-ins that you want to run when the attribute value is retrieved. The ID specified must reference the unique ID of attribute value processor plug-ins that are deployed to your system. If you specify multiple IDs, separate them using a comma. Example:

```
xyz.ItemAttribute.updated=./atom:title
xyz.ItemAttribute.updated.Processors=com.acme.TransformText
```

For more information, read *Creating and deploying custom attribute value processor plug-ins*.

Related tasks:

CF03 "Creating and deploying custom attribute value processor plug-ins" on page 3296
You can deploy custom attribute value processor plug-ins for DDC into the WebSphere Portal application extension registry. Attribute value processor plug-ins can be used to process the value of an item attribute after the value is determined by the list rendering profile.

Syntax for custom list-rendering profiles

Custom IBM Digital Data Connector (DDC) for WebSphere Portal Express plug-ins that do not delegate the initial bean list computation to the generic XML DDC plug-in can use a custom list-rendering profile. This case typically occurs when you integrate non-XML data.

Custom list-rendering profiles declare the set of available item attributes only. This way, the IBM Web Content Manager authoring user interface can still fill the corresponding **Select attribute resource tag type** selection box of the **Insert a Tag** dialog.

Custom list-rendering profiles are built in the same way as XPath based profiles, but with the following exceptions:

- To identify the profile as a custom list-rendering profile, a custom profile must contain the profile entry `<profile-id>.Type=Custom`.
- Custom list-rendering profile support only the following reduced set of list-rendering profile entry keys: `Name`, `BeanListProviderID`, `ResourceBundleBaseName`, `ItemAttribute`, `IsShownInAuthoringUI`, and `Extends`. The following keys are not supported: `NamespaceMapping`, `ListItemSelection`, `AssociatedItemAttribute`, `ComputedItemAttribute`, `ConstructedItemAttribute`, `ListProperty`, `ComputedListProperty`.
- The values that are used for item attribute declarations by using the `ItemAttribute` key have no significance.

The following list of list-rendering profile entries shows a sample custom list-rendering profile that you can use for transforming local file system information.

```
samples_files.Name=samples.files
samples_files.BeanListProviderID=samples.Files
samples_files.Type=Custom

samples_files.ItemAttribute.id=.
samples_files.ItemAttribute.title=.
samples_files.ItemAttribute.path=.
samples_files.ItemAttribute.encodedPath=.
samples_files.ItemAttribute.absolutePath=.
samples_files.ItemAttribute.uri=.
samples_files.ItemAttribute.parent=.
samples_files.ItemAttribute.encodedParent=.
samples_files.ItemAttribute.length=.
samples_files.ItemAttribute.suffix=.
samples_files.ItemAttribute.lastModified=.
samples_files.ItemAttribute.type=.
```

The Atom list-rendering profile

IBM Digital Data Connector (DDC) for WebSphere Portal Express provides a list-rendering profile that is ready to use for access to feeds that comply with the Atom syndication format standard.

This profile is named `ibm.portal.atom`. The profile is deployed as custom properties of the WP List Rendering Profile Service resource environment provider in the WebSphere Integrated Solutions Console. To display the profile, open the Custom Properties view of that resource environment provider in the WebSphere Integrated Solutions Console and activate a key filter by specifying `atom.*`.

To use this profile in your lists, set the profile attribute of the `ListRenderingContext` rendering plug-in tag to the value `ibm.portal.atom`. Example:

```
[Plugin:ListRenderingContext action="set" extension-id="ibm.portal.ddc.xml"
   profile="ibm.portal.atom"
   attribute="source=https://www.ibm.com/connections/communities/service/atom/catalog/public"
   compute="always"]
```

The Atom list-rendering profile declares the following attributes:

Alternate URL
`alternateLink`
: Link to an alternate representation of the entry or feed, for example a permalink to the HTML version of the entry, or the front page of the weblog.

Author's email address
`authorEmail`
: Email address of the feed entry author.

Author's Name
`authorName`
: Name of the feed entry author.

Author URL
`authorUri`
: URL of the feed entry author.

Category Labels
`categoryLabels`
: Provides a human-readable label for display of the categories.

Category Schemes
categorySchemes
Identifies the categorization schemes via a URI.

Category Terms
categoryTerms
Identifies the categories.

Content
content
The feed entry content.

null
contributorEmail
null

Contributor's email address
authorEmail
Email address of the feed entry contributor.

Contributor's Name
contributorName
Name of the feed entry contributor.

Contributor URL
contributorUri
URL of the feed entry contributor.

Edit URL
editLink
Link to edit the feed entry.

Enclosure URL
enclosureLink
Link to a related resource that might be large and might require special handling, for example an audio or video recording.

ID
id Identifier of the feed entry.

Published Date
published
The date on which the feed entry was published.

Feed URL
selfLink
Link to the feed itself.

Subtitle
subtitle
Subtitle of the feed entry.

Summary
summary
Summary of the feed entry.

Title
title Title of the feed entry.

Updated Date
updated
The date when the feed entry was last updated.

Related information:

Integrating remote JSON data

The IBM Digital Data Connector (DDC) for WebSphere Portal Express framework provides a generic JSON DDC plug-in that is ready to use for integrating external JSON data of your choice. You can use this plug-in to render external JSON data on your portal pages without having to write custom Java code.

About this task

The generic JSON DDC plug-in supports the concept of list-rendering profiles. It therefore makes it possible to integrate JSON data. The JSON data can be served in arbitrary JSON document formats. A list-rendering profile describes the transformation between a specific JSON document format and the generic bean list data structure that the DDC framework supports. In addition, the list-rendering profile enumerates all attribute names of the items in the resulting bean list that you can add to the result markup. This way, the IBM Web Content Manager **Insert Tag** user interface can present a dedicated selection box. That selection box contains all supported attributes for a specific list-rendering profile when you author DDC list appearance components.

CF06 "The generic JSON Digital Data Connector plug-in"
You can use the IBM Digital Data Connector (DDC) for WebSphere Portal Express plug-in directly to integrate remote JSON data without coding. It makes it possible to integrate JSON data that is served in arbitrary JSON document formats by supporting the concept of list-rendering profiles.

CF06 "Syntax of the `BasicJSONSelection`" on page 3288
A `BasicJSONSelection` defines the syntax that you can use to access data in a JSON object.

CF06 "Syntax for `BasicJSONSelection` based list-rendering profiles" on page 3289
A list-rendering profile that is based on `BasicJSONSelection` contains a set of name-value pairs called entries. This set of entries defines the set of available list properties and item attributes that are available for transforming external data into bean lists.

The generic JSON Digital Data Connector plug-in

You can use the IBM Digital Data Connector (DDC) for WebSphere Portal Express plug-in directly to integrate remote JSON data without coding. It makes it possible to integrate JSON data that is served in arbitrary JSON document formats by supporting the concept of list-rendering profiles.

A list-rendering profile describes the transformation between a specific JSON document format and the generic bean list data structure that this DDC plug-in generates. You identify the generic JSON DDC plug-in by its name `ibm.portal.ddc.json`. You can address it by using the `[Plugin:ListRenderingContext extension-id="ibm.portal.ddc.json"]` tag. The generic JSON DDC plug-in supports the attributes that are shown in the following list. You can specify these attributes when you set up the list-rendering context by using the `ListRenderingContext` tag that IBM Web Content Manager provides.

`source`
You can set this attribute to identify the source URI that serves the JSON data. The URI must be accessible through the portal resource addressability framework. Supported URI schemes include `http`, `https`, and `dav`. If you access external JSON data by using `http` or `https` URLs, make sure to adjust your outbound HTTP connection to allow outbound access to these URLs. For more

information, read the information about proxy configuration in the IBM WebSphere Portal Express documentation. For more information, read *Outbound HTTP connection*. Example:

```
[Plugin:ListRenderingContext action="set" extension-id="ibm.portal.ddc.json"
   profile="yourprofile.name"
   attribute="source=https://www.your_company.com/your_json_feed"
   compute="always"
```

sortCriteria
Set this attribute to sort the content of a bean list that is based on a specific attribute. This sorting attribute must be defined in the list-rendering profile that the current list-rendering context uses. This built-in sorting capability sorts only the data that is contained in the current bean list. Therefore, do not use this attribute if the bean list represents only a fragment of a larger result set. In such a situation, provide the sorting process by the backend service. If you do not specify this attribute, the bean list maintains the sequence of items as served by the external data source. To specify the sort order, use the `sortOrder` attribute.

sortOrder
Use this attribute in combination with the `sortCriteria` attribute to specify the sort order. Supported values are `asc` and `desc` for sorting in ascending or descending order. Example:

```
[Plugin:ListRenderingContext action="set" extension-id="ibm.portal.ddc.json"
   profile="yourprofile.name"
   attribute="source=https://www.your_company.com/your_json_feed"
   attribute="sortCriteria=title" attribute="sortOrder=asc"]
```

cacheScope
Set this attribute to identify the cache scope for bean lists that result from a specific list-rendering context. Supported values are as follows:

public
This value is the default value. If you set this attribute to `public`, the cached bean list values are shared across users.

private
If you set this attribute to `private`, individual bean list objects are cached per user. Specify `private` if the bean lists contain user-specific information.

Example:

```
[Plugin:ListRenderingContext action="set" extension-id="ibm.portal.ddc.json"
   profile="yourprofile.name"
   attribute="source=https://www.your_company.com/your_json_feed"
   attribute="cacheScope=private" compute="always"]
```

cacheType
Set this attribute to identify the cache type that you want to be used for bean lists that result from a specific list-rendering context. Supported values are as follows:

invalidateOnLogin
If you use this setting, cached bean list objects are invalidated if the user for whom the bean list was computed logs in to portal. Use this value only in combination with the setting `cacheScope=private`.

default
If you use this setting, resulting cached bean lists are not invalidated during user login.

invalidateCache
Use this attribute to invalidate the bean list. If you set this attribute to the value `always`, the bean list is always invalidated before the new list-rendering

context is set up. In other words, the bean list is recomputed for each rendering. To achieve a conditional invalidation, you can set the attribute to a different value. In this case, the cache is invalidated only if a portlet session attribute or a request attribute with the specified value is available in the current execution context.

clearCache
Use this attribute to clear the bean list cache. If you set this attribute to the value `always`, the cache is always cleared before the new list-rendering context is set up. To achieve a conditional clearing action, you can set the attribute to a different value. In this case, the cache is cleared only if a portlet session attribute or a request attribute with the specified value is available in the current execution context.

listItemSelection
Use this attribute only if you do not specify a profile in your list rendering context. If you set this attribute to a `BasicJSONSelection` expression, it serves as a selector statement that is used to break down the source JSON document into a list of individual JSON objects. Each object serves the data for a corresponding item in the resulting bean list. The DDC plug-in evaluates the `BasicJSONSelection` statement against the root element of the source JSON object. For the value, specify a valid `BasicJSONSelection` statement.

Consequently if you do not specify a profile in your list rendering context, you must specify `BasicJSONSelection` expressions in order to access data in the list designs for you Digital Data Connector. You do so by using the `[AttributeResource attributeName="<basic-json-selection>"]` tag.

Related concepts:

"Outbound HTTP connection" on page 2984
Applications in your IBM WebSphere Portal Express and the related user activities can require outbound HTTP connections to remote computer systems. The outbound HTTP connection service provides an administration infrastructure with a central point of administration for all outbound HTTP connections that are defined in the portal environment.

Syntax of the `BasicJSONSelection`

A `BasicJSONSelection` defines the syntax that you can use to access data in a JSON object.

A `BasicJSONSelection` resembles a basic way for accessing data in JSON feeds. You can use this syntax in `BasicJSONSelection` based list-rendering profiles or directly in an `[AttributeResource]` tag in your IBM Web Content Manager designs.

A `BasicJSONSelection` is made up by a list of selectors that are separated by the period (`.`) character.

You can use the following types of selectors:

Name selector
This selector returns the JSON member with the name from the JSON object. You can apply this selector only to JSON objects but not to JSON arrays.

The format of this selector is `name1.name2`. For the sample JSON object provided later, the `BasicJSONSelection store.address.country` returns `United States of America`.

Index selector
This selector returns a specific item from a JSON array. You can apply this

selector only to JSON arrays. The format of this selector is `[n]` where `n` denotes the nth element in a JSON Array.

For the sample JSON object provided later, the `BasicJSONSelection` `store.book.[0].author` returns `Nigel Rees`.

Attribute selector

This selector returns the first item from a JSON array that has a member the value of which matches the value that is specified in brackets. The format of this selector is `[member=value]` where `member=value` denotes the name of the member that has the value `value`. For the sample JSON object provided later, the `BasicJSONSelection` `store.book.[category=fiction].author` returns `Evelyn Waugh`.

Here is a sample JSON object:

```
{ "store": {
    "address":{
      "street": "Your Co Avanue",
      "city": "Your City",
      "zipcode": "12345",
      "state": "CA",
      "country": "United States of America"
    },
    "book": [
      { "category": "reference",
        "author": "Nigel Rees",
        "title": "Sayings of the Century",
        "isbn": "0-553-22222-3"},
      { "category": "fiction",
        "author": "Evelyn Waugh",
        "title": "Sword of Honour",
        "isbn": "0-553-11111-3"},
      { "category": "fiction",
        "author": "J. R. R. Tolkien",
        "title": "The Lord of the Rings",
        "isbn": "0-395-19395-8"}
    ]
    }
}
```

Syntax for `BasicJSONSelection` based list-rendering profiles

A list-rendering profile that is based on `BasicJSONSelection` contains a set of name-value pairs called entries. This set of entries defines the set of available list properties and item attributes that are available for transforming external data into bean lists.

The following list of list-rendering profile entries shows a sample `BasicJSONSelection` based list-rendering profile that you can use for transforming JSON documents:

```
sample_json.Name=sample.profile.json
sample_json.BeanListProviderID=ibm.portal.ddc.json
sample_json.Type=BasicJSONSelection
sample_json.ListItemSelection=children

# Item Attribute Declarations:
sample_json.ItemAttribute.id=id
sample_json.ItemAttribute.title=title
sample_json.ItemAttribute.authorName=author.name
sample_json.ItemAttribute.authorEmail=author.email


# List Property Declarations:
sample_json.ListProperty.id=id
```

The names that you use in the list-rendering profile entries must adhere to a specific syntax to be parsed correctly by the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework.

The format for the entry names is *profile-id*.*key*[.*name*[.*meta-data-key*]]. The meanings of the portions are described in the following list:

profile.id
This key specifies an internal ID. It is used only to correlate the entries that belong to the same profile.

key
This key identifies the list-rendering profile aspect that is affected by this entry. Valid values are `Name`, `BeanListProviderID`, `ResourceBundleBaseName`, `NamespaceMapping`, `ListItemSelection`, `ItemAttribute`, `AssociatedItemAttribute`, `ComputedItemAttribute`, `ConstructedItemAttribute`, `ListProperty`, `ComputedListProperty`, `Extends`, `ShowInAuthoringUI`, `Shared`, `Escape`.

name
This key specifies the name of the artifact that is to be defined, for example the attribute name. Names must not contain a period (`.`).

meta-data-key
Some list-rendering profiles entries support metadata that can be associated by using corresponding metadata keys. Valid values are `Type`, `Default`, `Format`, `Depends`, `ShowInAuthoringUI`.

The set of valid values for a list-rendering profile entry depend on the key that is used in the entry name. For more information, read the following topics.

CF06 "`BasicJSONSelection` list-rendering profile keys"
The following list shows the set of list-rendering profile entry keys that are available in the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework.

CF06 "`BasicJSONSelection` list-rendering profile metadata keys" on page 3294
Some item attribute and list property declarations support metadata that can be associated by using corresponding metadata keys.

`BasicJSONSelection` list-rendering profile keys:

The following list shows the set of list-rendering profile entry keys that are available in the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework.

Name
This value specifies the name of this list-rendering profile. Specifying the name is mandatory for each rendering profile. You can reference this value through the profile attribute of the `[Plugin:ListRenderingContext]` tag and through the `Extends` list-rendering profile entry of other list-rendering profiles. The `Extends` list-rendering profile entry is described later in this list. If this profile does not reference a resource bundle through the `ResourceBundleBaseName` key, this name also shows up in the **Insert a tag** dialog of the IBM Web Content Manager authoring user interface. The name must be unique among all list-rendering profiles in the system.

Example: To set the name of the profile that is identified by profile ID `xyz` to `yourCo.ProductList`, specify `xyz.name=yourCo.ProductList`.

BeanListProviderID

Specifying the ID for the DDC plug-in is mandatory for each list-rendering profile. It specifies the extension ID of the DDC plug-in that is used with this profile. The DDC plug-in also gets the calls for computing all computed item attributes. Computed item attributes are described later in this list.

Example: To use the generic JSON DDC plug-in, specify `xyz.BeanListProviderID=ibm.portal.ddc.json`.

ResourceBundleBaseName

The resource bundle base name is optional. It specifies the base name of the Java resource bundle that serves the translated strings for the name of this profile and all item attributes that are defined in this profile. The translated strings include associated, computed, and constructed item attributes. The resource bundle base name must identify a Java resource bundle that is available in the portal class path.

Example: To use the resource bundle `yourCo.Bundle` for the profile that is identified by profile ID `xyz`, you specify `xyz.ResourceBundleBaseName=yourCo.Bundle`

ShowInAuthoringUI

Valid values are `true` and `false`. If you set this attribute to `false`, the list-rendering profile does not show up in the **Select the target component for this tag** selection list of the **Insert a Tag** dialog of the Web Content Manager authoring user interface.

ListItemSelection

The list item selection defines the `BasicJSONSelection` statement that is used to break down the source JSON document into a list of individual JSON objects. Each object serves the data for a corresponding item in the resulting bean list. The DDC plug-in evaluates the `BasicJSONSelection` statement against the root element of the source JSON object. For the value, specify a valid `BasicJSONSelection` statement. Specifying the list item selection is mandatory for profiles of type `BasicJSONSelection`.

Example: To turn all entries of a JSON Array with the name `children` that are contained in a JSON member with the name `data` in a JSON feed into list items in the profile that is identified by the profile ID `xyz`, specify `xyz.ListItemSelection=data.children`.

ItemAttribute

This key declares an attribute available in all items in the list. The attribute values can then be accessed in your Web Content Manager design components by using the `[AttributeResource attributeName=""]` tag by setting `attributeName` to the name value of the corresponding `ItemAttribute` declaration. For example, to access the value of an item attribute declared with the key `xyz.ItemAttribute.myTitle`, you can use the following Web Content Manager tag `[AttributeResource attributeName="myTitle"]`. Furthermore, declared attributes are listed in the **Select attribute resource tag type** selection box of the **Insert a Tag** dialog of the Web Content Manager authoring user interface when you select the containing profile in the **Select the target component for this tag** selection box.

Important: Each profile must define at least one `ItemAttribute` named `id`. This ID is used to uniquely identify an individual item in the list.

In profiles of type `BasicJSONSelection`, the value for an item attribute declaration is a `BasicJSONSelection` statement that is used for extracting the

declared attribute value from the JSON source object. The DDC plug-in evaluates the `BasicJSONSelection` statement for each item in the lists relative to the corresponding JSON fragments that are identified by using the `ListItemSelection` specification.

Example: To extract the author email of the entries in a JSON feed, specify `xyz.ItemAttribute.authorEmail=author.email`.

`AssociatedItemAttribute`

This key declares an attribute that is available in all items in the list. You can access the attribute values accessed in your Web Content Manager design components in the same way as item attributes. They also appear in the same places in the Web Content Manager authoring user interface. The difference is that in `BasicJSONSelection` based profiles, you can use associated item attribute declarations to extract data not directly from the source JSON document. Instead, you can extract data from a linked JSON source document that is referenced by the source JSON document. To define an associated item attribute, you must build the value for such an entry by combining a source URL with a `BasicJSONSelection` statement to select the attribute value from the linked document. You specify the source URL by using the name of the attribute that selects the source URL from the current bean list. You specify the `ItemAttribute` reference that provides the document URL by using the pattern `{$item-attribute-name}`

Example: To obtain the full product description from a list of products, specify as follows:

```
xyz.ItemAttribute.detailsLink=entry.link[rel=details]
xyz.AssociatedItemAttribute.productFullDescription={$detailsLink}details.description
```

The DDC plug-in computes the values for this attribute in the following sequence:

1. The DDC plug-in evaluates the value of the `detailsLink` item attribute.
2. The DDC plug-in loads the JSON document from that URL.
3. The DDC plug-in evaluates the `BasicJSONSelection` statement `detailsLink` on that document.

`ConstructedItemAttribute`

This key declares an attribute that is available in all items in the list. If you define such an attribute, you can access the attribute values in your Web Content Manager design components in the same way as item attributes. The attribute values also appear in the same places as item attributes in the Web Content Manager authoring user interface. The difference between the two attribute types is as follows: You can build the values for constructed item attributes by combining the values from one or more other item attributes that have a static string. To define a constructed item attribute, you must provide a template string that is used to build the actual attribute values. The template string can contain arbitrary text that is mixed with item attribute references. To specify the `ItemAttribute` references, you provide the document URL by using the pattern `{$item-attribute-name}`. Example: To construct an image URL from an image location and image file name item attribute, specify as follows:

```
xyz.ItemAttribute.imageLocation=entry.image.location
xyz.ItemAttribute.imageFileName=entry.image.name
xyz.ConstructedItemAttribute.imageURL={$imageLocation}/{$imageName}
```

`ComputedItemAttribute`

This key declares an attribute that is available in all items in the list. You can then access the attribute values in your Web Content Manager design components in the same way as item attributes. They attribute values also

appear in the same places as item attributes in the Web Content Manager authoring user interface. The difference between the two types of attribute types is as follows: The values for computed item attributes are not extracted from JSON objects but computed based on the item attribute values and other context information. The DDC plug-in that is associated to the list-rendering profile through the `BeanListProviderID` key computes the attribute values. To serve such data, the DDC plug-in must implement the optional `com.ibm.portal.wcm.plr.ComputedAttributeValueProvider` Java interface. This interface is defined by the public IBM Digital Data Connector (DDC) for WebSphere Portal Express APIs. The computation is typically done based on the non-computed attribute values in combination with configuration data or other extra context information.

Note: The generic JSON DDC plug-in supports no computed item attributes.

To define a computed item attribute, you must declare on which data the attribute value computation depends:

- The computed item attribute value depends only on the other item attribute values. In this case, you must set it to value `{$lazy}`.
- The computed item attribute value depends on other item attribute values and on the current rendering context that is the current portlet request and response objects. These objects are not cached in the bean list cache, but they are recomputed for each rendering request. If the computed item attribute value depends on the current request context, you must set the value of such a computed item attribute value to the string `{$default}`.

Example: To define an attribute that represents a stateful portal resource URL that points to the current item, you can declare a computed item attribute such as this one: `xyz.ComputedItemAttribute.portalLink={default}`.

During run time, the DDC framework then requests the actual values for this attribute. It does so by starting the following method on the specific DDC plug-in plug-in that associated with this list-rendering profile:

`com.ibm.portal.wcm.plr.ComputedAttributeValueProvider.getComputedItemAttributeValue()`

`ListProperty`

This key declares a property of the list. You can then access the attribute values in your Web Content Manager design components. You do so by using the `[Plugin:ListRenderingContext action="getListProperty" key=""]` tag and by setting the `key` to the name value of the corresponding `ListProperty` declaration. For example, to access the value of a list property that is declared with the key `xyz.ListProperty.myProperty`, you use the following Web Content Manager tag: `[Plugin:ListRenderingContext action="getListProperty" key="myProperty"]` The value for a list property declaration is a `BasicJSONSelection` statement that is used for extracting the declared attribute value from the JSON source document. The plug-in evaluates the `BasicJSONSelection` statement relative to the root of the source JSON document.

Example: To get the link to the next feed page of a JSON feed, specify

`xyz.ListProperty.nextLink=links.[rel=next]`

`ComputedListProperty`

This key declares a list property of the list. You can then access the attribute values in your Web Content Manager design components in the same way as list properties. The difference between list properties and computed list properties as follows: The values for computed list properties are not extracted from JSON documents, but computed based on the item attribute values and

other context information. The DDC plug-in that is associated to the list-rendering profile through `theBeanListProviderID` key computes the property values. To serve such data, the DDC plug-in must implement the optional `com.ibm.portal.wcm.plr.ComputedAttributeValueProvider` Java interface. This interface is defined by the public APIs of Digital Data Connector APIs. The computation is usually done based on the non-computed list property values in combination with configuration data or other extra context information.

Note: The generic JSON DDC plug-in does not support computed list properties. To define a computed list property, you must declare on which of the following two types of data the property value computation depends:

- The computed item attribute value depends only on the other list property values. In this case, you must set it to value `{$lazy}`.
- The computed list property value depends on other list property values and on the current rendering context, that is the current portlet request and response objects. These objects are not cached in the bean list cache, but they are recomputed for each rendering request. If the computed list property value depends on the current request context, you must set the value of such a computed list property value to the value `{$default}`.

`Extends`

Use this key to define a derived profile. A derived profile includes the item attribute and the list property declarations of other list-rendering profiles by referencing these profiles. For more information, read *Creating list-rendering profiles*.

Related tasks:

"Creating list-rendering profiles" on page 3272
When you create a list-rendering profile, you can either create a completely new profile or create a derived profile by extending an existing profile. A derived profile includes the item attribute and list property declarations of other list-rendering profiles by referencing these profiles.

`BasicJSONSelection` list-rendering profile metadata keys:

Some item attribute and list property declarations support metadata that can be associated by using corresponding metadata keys.

You can use them in combination with `ItemAttribute`, `AssociatedItemAttribute`, `ConstructedItemAttribute`, `ComputedItemAttribute`, and `Listproperty` declarations that are listed under *List-rendering profile keys*. The following list describes the supported metadata key values:

`Default`

Use this metadata key to specify a static default value that you want to be returned if the appropriate value cannot be retrieved. Example:

```
xyz.ItemAttribute.rating=./avgRating
xyz.ItemAttribute.rating.default=0.5
```

`Depends`

To enforce a specific computation order for the non-lazy computed item attributes and list properties in the current profile, set this metadata on the `ComputedItemAttribute` and `ComputedListProperty`. Such a computation order can be required if specific computed attributes must access the value of other computed attributed values. In this case, the attribute that requires the value of another computed attributed must list the other attribute in the `Depends`

metadata key. If the attribute depends on multiple other computed attributes, you can list multiple attribute names as a comma-separated list. Example: You defined computed attributes named `link` and `proxiedLink`. You want to make sure that the value for the `link` attribute is computed before the value for the `proxiedLink` attribute. In this case, specify as follows:

```
xyz.ItemAttribute.link={computed}
xyz.ItemAttribute.proxiedLink={computed}
xyz.ItemAttribute.proxiedLink.depends=link
```

Note: You do not have to flag dependencies on non-computed attributes. Non-computed attributes are always extracted from the JSON before the computed attributes are resolved.

Escape

Use this metadata to define the escaping that you want applied when you write the values of this attribute. Specify one of the values `xml`, `json`, `javascript`, or `none`. The default value is `none`.

Shared

Use this metadata in context of `AssociatedItemAttribute` declarations to define the cache scope for the associated JSON document. If this document can be shared among multiple users, you can set this metadata to the value `true`. If the associated document can contain different data for different users, cache it per user. In this case, you can leave this metadata undefined or explicitly set it to `false`.

Note: Associated documents are cached in the `com.ibm.workplace.wcm.pzn.plr.json.DocumentCache`. Associated documents that are cached in scope `Shared` are not automatically invalidated during user login.

Processors

Use this metadata on an `ItemAttribute`, `AssociatedItemAttribute`, `ConstructedItemAttribute`, or `ComputedItemAttribute` to specify one or multiple IDs of attribute value processor plug-ins that you want to run when the attribute value is retrieved. The ID specified must reference the unique ID of attribute value processor plug-ins that are deployed to your system. If you specify multiple IDs, separate them using a comma. Example:

```
xyz.ItemAttribute.updated=entry.title
xyz.ItemAttribute.updated.Processors=com.acme.TransformText
```

For more information, read *Creating and deploying custom attribute value processor plug-ins*.

Related tasks:

CF03 "Creating and deploying custom attribute value processor plug-ins" on page 3296
You can deploy custom attribute value processor plug-ins for DDC into the WebSphere Portal application extension registry. Attribute value processor plug-ins can be used to process the value of an item attribute after the value is determined by the list rendering profile.

Creating and deploying custom Digital Data Connector plug-ins

You can deploy custom IBM Digital Data Connector (DDC) for WebSphere Portal Express plug-ins as plug-ins into the WebSphere Portal application extension registry.

About this task

The extension point ID for custom DDC plug-ins is `com.ibm.portal.wcm.plr.BeanListProvider`. Individual DDC plug-in implementations implement the `com.ibm.portal.wcm.plr.BeanListProvider` Java interface. This interface is defined in the public Digital Data Connector SPI Javadoc documentation.

Individual DDC plug-in implementations can delegate the initial bean list computation to the generic XML DDC plug-in and extend the returned bean list. They do so by modifying the bean list object that the generic XML DDC plug-in returns or by adding more computed attributes. To implement the computation logic for individual computed attributes, DDC plug-ins can use the optional `com.ibm.portal.wcm.plr.ComputedAttributeValueProvider` Java interface. If you implement this interface, the DDC framework calls the corresponding methods whenever a bean list attribute requests computed item attributes.

Related concepts:

"Application extension registry" on page 3148
WebSphere Portal Express provides an application extension registry which is equivalent to the application extension registry provided by IBM WebSphere Application Server. This registry allows any J2EE compliant application to define extension points for other applications to use, or to plug in to other extensible applications.

Creating and deploying custom attribute value processor plug-ins

CF03

You can deploy custom attribute value processor plug-ins for DDC into the WebSphere Portal application extension registry. Attribute value processor plug-ins can be used to process the value of an item attribute after the value is determined by the list rendering profile.

About this task

The extension point ID for custom attribute value processor plug-ins is `com.ibm.portal.wcm.plr.AttributeValueProcessor`.

Individual plug-in implementations implement one of the two following Java interfaces for the object factory responsible for creating the actual attribute value processor instance.

- `com.ibm.portal.wcm.plr.AttributeValueProcessorFactory`
- `com.ibm.portal.wcm.plr.AttributeValueOnRequestProcessorFactory`

The actual attribute value processor implementations must implement the `com.ibm.portal.wcm.plr.AttributeValueProcessor` Java interface.

All three interfaces are defined in the public Digital Data Connector SPI. For more information about these interfaces, see *The developer WebSphere Portal SPI documentation.*

Digital Data Connector cache tuning

To improve performance, you can tune the caches for the IBM Digital Data Connector (DDC) for WebSphere Portal Express framework.

Both social rendering and the DDC framework use the same caches to cache bean list information.

"Digital Data Connector caches"
The IBM Digital Data Connector (DDC) for WebSphere Portal Express framework provides the caches that are listed here.

Related tasks:

Using the list-rendering cache

Digital Data Connector caches

The IBM Digital Data Connector (DDC) for WebSphere Portal Express framework provides the caches that are listed here.

Both social rendering and the DDC framework use the following caches to cache bean list information:

com.ibm.workplace.wcm.pzn.plr.BeanListCache
The bean list cache caches the bean list Java objects that the Digital Data Connector plug-ins return. The DDC plug-ins control the cache key generation for the individual entries and whether the bean lists are automatically removed from the cache during user login. By default, this cache is enabled.

Note: Single entries of this cache can have a size of several MB. Therefore, the default number of cache entries for this cache is much lower than the default of other portal caches. When you use the bean list cache, closely monitor the cache and tune it as required. You might also consider the size of individual cache entries and how to influence it. For more information, read *Configuring the maximum number of items loaded from Connections*.

com.ibm.workplace.wcm.pzn.plr.xml.DocumentCache
The document cache is used by the generic XML DDC plug-in for caching the Document Object Model (DOM) objects for individual source URIs. This cache specifically caches the DOMs for associated item attributes. If an individual associated item attribute is flagged as `shared` in the list-rendering profile, the cache entries are shared between different users. Such shared documents do not get invalidated on user login. Documents that are loaded through non-shared associated item attributes are cached separately per user. These cache entries are automatically invalidated during login. By default, this cache is enabled.

com.ibm.workplace.wcm.pzn.plr.json.DocumentCache
The document cache is used by the generic JSON DDC plug-in for caching the Document Object Model (DOM) objects for individual source URIs. This cache specifically caches the DOMs for associated item attributes. If an individual associated item attribute is flagged as `shared` in the list-rendering profile, the cache entries are shared between different users. Such shared documents do not get invalidated on user login. Documents that are loaded through

non-shared associated item attributes are cached separately per user. These cache entries are automatically invalidated during login. By default, this cache is enabled.

com.ibm.workplace.wcm.pzn.plr.ListRenderingCache
The list-rendering cache caches the markup that a specific appearance component generates for a specific bean list instance. If you enable this cache, the updates in the appearance component might not become visible immediately, as updates to the corresponding IBM Web Content Manager design components do not invalidate this cache. In general, the entries in this cache are invalidated together with the corresponding bean list objects in the bean list cache that is listed earlier. As a result, it is good practice to disable this cache on authoring systems and enable it on delivery systems.

To use this cache, you must use the `ListRenderingCache` rendering plug-in to instrument the Web Content Manager design components that are involved in the markup generation for this cache. For more information, read *Using the list-rendering cache*.

You configure the caches through the WP Cache Manager Service resource environment provider in the WebSphere Integrated Solutions Console. For more information, read the *Portal service configuration* and *Setting service configuration properties*.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

"Configuring the maximum number of items loaded from Connections" on page 2123
You can define a value for the maximum number of social objects that you want the IBM Connections to return when data for a list of social objects is requested.

Related reference:

"Portal service configuration" on page 289
IBM WebSphere Portal Express comprises a framework of configuration services to accommodate the different scenarios that portals of today need to address. You can configure some of these services.

Hints and tips for Digital Data Connector

Learn about things that are useful to know when you work with the social rendering integration with IBM Digital Data Connector (DDC) for WebSphere Portal Express.

Configure access on pages that show social rendering content items only for users who also have an account in IBM Connections.
Otherwise, the user gets an error when the user opens the portal page with the social rendering content items. Examples:

- A IBM WebSphere Portal Express administrator might access a page with social rendering content items by using the user ID `wpsadmin`. In this case, an error occurs because that user ID does not exist in Connections.
- Your WebSphere Portal Express might be configured with more than one LDAP, but your Connections might be configured with only one of these LDAPs. If a user who is stored in one of the other LDAPs accesses a page with social rendering content items, an error occurs.

To avoid such errors, give access to such pages only to users who have an account in Connections.

Digital Data Connector does not support WSRP
The Digital Data Connector infrastructure does not support remote rendering through WSRP (Web Services for Remote Portlets). This means that lists rendered through the DDC cannot be provided as remote portlets.

Helper class samples for web content context

You can create the helper classes PortletWCMContextHelper, PortalWCMContextHelper, and WCMContextHelper from the sample code that is provided here to programmatically determine the current web content context. The context indicates a content item or site area that is rendered by a web content viewer.

Important: These helper classes are not provided by default with IBM Web Content Manager. Example code is provided here that you can take and use to implement the classes yourself.

The helper classes can be used by custom themes or custom portlets that need to render information that is related to the current context.

The web content viewer determines the context to be rendered by evaluating several conditions in the following order:

1. Private render parameter
2. **path-info** parameter
3. Public render parameter
4. Portlet configuration setting for the web content viewer
5. Web content association that is defined for the page

The viewer evaluates each condition in turn until it finds a valid context. As soon as the viewer finds a context, remaining conditions are not evaluated.

Private render parameters and portlet configuration settings are visible only to the web content viewer. However, **path-info** parameters, public render parameters, and content associations are visible to all portlets on a page and to portal code (for example, in a theme).

"PortletWCMContextHelper"
Use the PortletWCMContextHelper class to determine the web content context from within a portlet.

"PortalWCMContextHelper" on page 3301
Use the PortalWCMContextHelper class to determine the web content context from within portal code, such as a theme.

"WCMContextHelper" on page 3304
The WCMContextHelper helper class is an abstract class that is used by the PortalWCMContextHelper class and the PortletWCMContextHelper class.

PortletWCMContextHelper

Use the PortletWCMContextHelper class to determine the web content context from within a portlet.

Preparation

To use the PortletWCMContextHelper class, configure the portlet to receive a public render parameter and a **path-info** parameter. Update the`portlet.xml` file for the portlet, and add the following entries inside the `<portlet-app>` tag:

```
<public-render-parameter>
  <description>WCM public context</description>
  <identifier>PUBLIC_CONTEXT</identifier>
  <qname xmlns:wcm="http://www.ibm.com/xmlns/prod/datatype/content">wcm:context</qname>
</public-render-parameter>
<public-render-parameter>
  <description>Shared path-info parameter of WebSphere Portal</description>
  <identifier>PATH_INFO</identifier>
  <qname xmlns:wcm="http://www.ibm.com/xmlns/prod/websphere/portal/publicparams">wcm:path-info</qname>
</public-render-parameter>
```

In addition, update the `<portlet>` tag with the following lines:

```
<supported-public-render-parameter>PUBLIC_CONTEXT</supported-public-render-parameter>
<supported-public-render-parameter>PATH_INFO</supported-public-render-parameter>
```

Implementation

After a new instance of the PortletWCMContextHelper is created, the portlet can start the getCurrentWCMContext method, passing in the parameters **PortletRequest** and **PortletResponse**:

```
public String getCurrentWCMContext(PortletRequest, PortletResponse)
```

This method returns a string that contains the content path that defines the current web content context. To determine the content path, the getCurrentWCMContext method runs the following checks:

1. The method reads the value of the **path-info** public render parameter. The method constructs the content path from the **path-info** parameter and the content association of the current page, when these conditions are true:
   - The **path-info** parameter is present.
   - The `friendly.pathinfo.enabled` property is enabled in the portal configuration service.
2. If the **path-info** parameter is not present or if the `friendly.pathinfo.enabled` property is disabled, the method reads the public render parameter. If the public render parameter is present, the method returns the value of this parameter.
3. If no public render parameter is present, the method evaluates the current page for a default content association. The method then returns the path of the content item that is mapped to the page.

Source of PortletWCMContextHelper

```
/*****************************************************************
 * Copyright IBM Corp. 2010, 2013                                *
 *****************************************************************/

package com.ibm.portal.extension;


import java.util.Map;
import javax.naming.*;
import javax.portlet.*;
import com.ibm.portal.*;
import com.ibm.portal.content.*;
import com.ibm.portal.portlet.service.*;
import com.ibm.portal.portlet.service.model.*;
import com.ibm.portal.services.contentmapping.exceptions.*;
import com.ibm.portal.state.service.*;
import com.ibm.workplace.wcm.api.exceptions.*;

/**
 * Helper class to determine the current WCM context. This class can only be used from portlet code.
```

```
 * From portal code (e.g. theme)  please use PortalWCMContextHelper instead.
 */

public class PortletWCMContextHelper extends WCMContextHelper {

    private final NavigationSelectionModelProvider navSelectionModelProvider;

   /**
    * Initializes the PortletWCMContextHelper.
    *
    * @throws NamingException
    * @throws PortletServiceUnavailableException
    */
   public PortletWCMContextHelper() throws NamingException, PortletServiceUnavailableException {
      // this initialization needs to be done only once
      final Context ctx = new InitialContext();
      PortletServiceHome psh = (PortletServiceHome) ctx.lookup(PortletStateManagerService.JNDI_NAME);
      navSelectionModelProvider = psh.getPortletService(NavigationSelectionModelProvider.class);
   }

   /**
    * Gets the WCM context of the current page. It checks path info, public render parameter and
    * content mapping in this order. A WCM context defined as private render parameter or
    * preference of the Web Content Viewer portlet is NOT returned.
    *
    * @param request PortletRequest
    * @param response PortletResponse
    *
    * @return String representation of a content path.
    *
    * @throws ModelException
    * @throws IllegalDocumentTypeException
    * @throws DocumentRetrievalException
    * @throws DocumentIdCreationException
    * @throws OperationFailedException
    * @throws ServiceNotAvailableException
    * @throws ContentMappingDataBackendException
    */

   @SuppressWarnings("unchecked")
   public String getCurrentWCMContext(final RenderRequest request, final RenderResponse response)
         throws ModelException, ContentMappingDataBackendException, ServiceNotAvailableException,
         OperationFailedException, DocumentIdCreationException, DocumentRetrievalException, IllegalDocumentTypeException {

      String contentPath = null;
      final Map<String, String []> publicParameter = request.getPublicParameterMap();
      final ContentNode currentPage =
          getCurrentPageNode(navSelectionModelProvider.getNavigationSelectionModel(request, response));

      // check path info

      if (publicParameter.containsKey("PATH_INFO")) {
         final String [] pathInfo = publicParameter.get("PATH_INFO");
         if (pathInfo != null && pathInfo.length > 0) {
            String contentMapping = getContentMapping(currentPage);
            contentPath = assembleContentPath(contentMapping, pathInfo);
         }
      }
      if (contentPath == null) {
         // check public WCM context
         contentPath = request.getParameter("PUBLIC_CONTEXT");
         if (contentPath == null) {
            // check content mapping
            contentPath = getContentMapping(currentPage);
         }
      }
      return contentPath;
   }
}
```

PortalWCMContextHelper

Use the PortalWCMContextHelper class to determine the web content context from within portal code, such as a theme.

To use the PortalWCMContextHelper class, create an instance of the helper class and start the getCurrentWCMContext method, passing in the parameters **`HttpServletRequest`** and **`HttpServletResponse`**:

```
public String getCurrentWCMContext(HttpServletRequest, HttpServletResponse)
```

This method returns a string that contains the content path that defines the current web content context. To determine the content path, the getCurrentWCMContext method runs the following checks:

1. The method reads the value of the **path-info** public render parameter. The method constructs the content path from the **path-info** parameter and the content association of the current page, when these conditions are true:
   - The **path-info** parameter is present.
   - The `friendly.pathinfo.enabled` property is enabled in the portal configuration service.
2. If the **path-info** parameter is not present or if the `friendly.pathinfo.enabled` property is disabled, the method reads the public render parameter. If the public render parameter is present, the method returns the value of this parameter.
3. If no public render parameter is present, the method evaluates the current page for a default content association. The method then returns the path of the content item that is mapped to the page.

Source of PortalWCMContextHelper

```
/*****************************************************************
 * Copyright IBM Corp. 2010, 2011                                *
 *****************************************************************/
package com.ibm.portal.extension;
import java.util.Map;
import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.xml.namespace.QName;
import com.ibm.portal.MetaData;
import com.ibm.portal.ModelException;
import com.ibm.portal.ObjectID;
import com.ibm.portal.content.ContentMetaDataModel;
import com.ibm.portal.content.ContentModel;
import com.ibm.portal.content.ContentNode;
import com.ibm.portal.model.ContentMetaDataModelHome;
import com.ibm.portal.model.ContentModelHome;
import com.ibm.portal.services.contentmapping.exceptions.ContentMappingDataBackendException;
import com.ibm.portal.state.StateHolder;
import com.ibm.portal.state.accessors.StateAccessor;
import com.ibm.portal.state.accessors.StateAccessorFactory;
import com.ibm.portal.state.accessors.exceptions.StateNotInRequestException;
import com.ibm.portal.state.accessors.portlet.PortletAccessorFactory;
import com.ibm.portal.state.accessors.portlet.SharedStateAccessor;
import com.ibm.portal.state.exceptions.CannotInstantiateAccessorException;
import com.ibm.portal.state.exceptions.StateException;
import com.ibm.portal.state.exceptions.UnknownAccessorTypeException;
import com.ibm.portal.state.service.PortalStateManagerServiceHome;
import com.ibm.portal.state.service.StateManagerService;
import com.ibm.workplace.wcm.api.exceptions.DocumentIdCreationException;
import com.ibm.workplace.wcm.api.exceptions.DocumentRetrievalException;
import com.ibm.workplace.wcm.api.exceptions.IllegalDocumentTypeException;
import com.ibm.workplace.wcm.api.exceptions.OperationFailedException;
import com.ibm.workplace.wcm.api.exceptions.ServiceNotAvailableException;

/**
 * Helper class to determine the current WCM context. This class can only be used from portal code.
 * From portlet code please use PortletWCMContextHelper instead.
 */

public class PortalWCMContextHelper extends WCMContextHelper {

   /** QName of the shared render parameter that holds the WCM public context */
   static final QName PUBLIC_WCM_CONTEXT_PARAM_QNAME =
         new QName("http://www.ibm.com/xmlns/prod/datatype/content", "context");

   /** QName of the shared render parameter that holds the path info */
   static final QName PUBLIC_PATH_INFO_QNAME =
         new QName("http://www.ibm.com/xmlns/prod/websphere/portal/publicparams", "path-info");

   /**
    * Page metadata key that controls the sharing scope for public render
    * parameter of portlets on this page
    */
   static final String PARAM_SHARING_SCOPE_KEY = "param.sharing.scope";
   private final PortalStateManagerServiceHome stateManagerServiceHome;
   private final ContentModelHome contentModelHome;
```

```
private final ContentMetaDataModelHome contentMetaDataModelHome;
/**
 * Initializes the PortalWCMContextHelper.
 *
 * @throws NamingException
 */
public PortalWCMContextHelper() throws NamingException {
   // this initialization needs to be done only once.
   final Context ctx = new InitialContext();
   stateManagerServiceHome =
      (PortalStateManagerServiceHome) ctx.lookup(PortalStateManagerServiceHome.JNDI_NAME);
   contentModelHome = (ContentModelHome) ctx.lookup(ContentModelHome.JNDI_NAME);
   contentMetaDataModelHome = (ContentMetaDataModelHome) ctx.lookup(ContentMetaDataModelHome.JNDI_NAME);
}

/**
 * Gets the WCM context of the current page. It checks path info, public WCM context render parameter
 * and content mapping in this order. A WCM context defined as private render parameter or
 * preference of the Web Content Viewer portlet is NOT returned.
 *
 * @param request HttpServletRequest
 * @param response HttpServletResponse
 *
 * @return String representation of a content path.
 *
 * @throws StateException
 * @throws IllegalDocumentTypeException
 * @throws DocumentRetrievalException
 * @throws DocumentIdCreationException
 * @throws OperationFailedException
 * @throws ServiceNotAvailableException
 * @throws ContentMappingDataBackendException
 * @throws ModelException
 */

public String getCurrentWCMContext(final HttpServletRequest request, final HttpServletResponse response)
      throws StateException, ContentMappingDataBackendException, ServiceNotAvailableException,
      OperationFailedException, DocumentIdCreationException, DocumentRetrievalException,
      IllegalDocumentTypeException, ModelException {
   String contentPath = null;
   final StateManagerService stateManagerService =
      stateManagerServiceHome.getPortalStateManagerService(request, response);
   try {
      final StateHolder currentState = getCurrentPortalState(request, stateManagerService);
      final ContentModel<ContentNode> contentModel =
      contentModelHome.getContentModelProvider().getContentModel(request, response);
      final ContentMetaDataModel metaDataModel =
         contentMetaDataModelHome.getContentMetaDataModelProvider().getContentMetaDataModel(request, response);
      // find out public render parameter scope
      final ObjectID currentPage = getCurrentPageID(stateManagerService, currentState);
      final ContentNode page = contentModel.getLocator().findByID(currentPage);
      final MetaData metadata = metaDataModel.getMetaData(page);
      final Object scope = metadata.getValue(PARAM_SHARING_SCOPE_KEY);
      String publicRenderScope;
      if (scope != null) {
         publicRenderScope = scope.toString();
      } else {
         publicRenderScope = SharedStateAccessor.KEY_GLOBAL_PUBLIC_RENDER_PARAMETERS;
      }

     // load public render parameter
      final PortletAccessorFactory portletAccFct =
        stateManagerService.getAccessorFactory(PortletAccessorFactory.class);
      final SharedStateAccessor sharedStateAcc =
          portletAccFct.getSharedStateAccessor(publicRenderScope, currentState);
      if (sharedStateAcc != null) {
         try {
            final Map<QName, String[]> sharedRenderParams = sharedStateAcc.getParameters();
            // check path info first.
            if (sharedRenderParams.containsKey(PUBLIC_PATH_INFO_QNAME)) {
               final String[] pathInfo = sharedRenderParams.get(PUBLIC_PATH_INFO_QNAME);
               if (pathInfo != null && pathInfo.length > 0) {
                  String contentMapping = getContentMapping(stateManagerService, currentState);
                  contentPath = assembleContentPath(contentMapping, pathInfo);
               }
            }
            // if there is no path info check the public WCM context.
            if (contentPath == null && sharedRenderParams.containsKey(PUBLIC_WCM_CONTEXT_PARAM_QNAME)) {
               final String[] values = sharedRenderParams.get(PUBLIC_WCM_CONTEXT_PARAM_QNAME);
               if (values != null && values.length > 0) {
                  contentPath = values[0];
               }
            }
      } finally {
         sharedStateAcc.dispose();
      }
   }
   if (contentPath == null) {
      // check for a content mapping
      contentPath = getContentMapping(stateManagerService, currentState);
```

```
      }
      } finally {
         stateManagerService.dispose();
      }
      return contentPath;
   }

   /**
    * Gets the current portal state.
    *
    * @param request HttpServletRequest
    * @param stateManagerService StateManagerService, entry point to the portal state API
    *
    * @return the current portal state
    *
    * @throws UnknownAccessorTypeException
    * @throws CannotInstantiateAccessorException
    * @throws StateNotInRequestException
    */
   private StateHolder getCurrentPortalState(final HttpServletRequest request,
         final StateManagerService stateManagerService) throws UnknownAccessorTypeException,
         CannotInstantiateAccessorException, StateNotInRequestException {
      StateHolder result = null;
      final StateAccessorFactory stateAccFac =
         (StateAccessorFactory) stateManagerService.getAccessorFactory(StateAccessorFactory.class);
      final StateAccessor stateAcc = stateAccFac.getStateAccessor();
      try {
         result = stateAcc.getStateHolder(request);
      } finally {
         stateAcc.dispose();
      }
      return result;
   }
}
```

WCMContextHelper

The WCMContextHelper helper class is an abstract class that is used by the PortalWCMContextHelper class and the PortletWCMContextHelper class.

The WCMContextHelper class provides three methods:

getCurrentPageID
: This method uses the State API to determine the currently selected page and returns the object ID of the page.

getContentMapping
: This method uses the Content Mapping Service to retrieve the content item that is associated with the current page. The association always contains the ID of the mapped content. After you determine the content ID, the method uses the IBM Web Content Manager API to convert the ID to a content path. The method returns the content path of the mapped item.

assembleContentPath
: This method takes the **path-info** parameter, which has multiple values, and the content association of the current page. The method creates the web content context by appending all elements of the **path-info** parameter to the content mapping. The method then returns the result, which is the content path that defines the web content context.

Source of WCMContextHelper

```
/*****************************************************************
 * Copyright IBM Corp. 2010, 2013                                *
 *****************************************************************/

package com.ibm.portal.extension;

import javax.naming.*;
import com.ibm.portal.content.*;
import com.ibm.portal.navigation.*;
import com.ibm.portal.services.contentmapping.*;
import com.ibm.portal.services.contentmapping.exceptions.*;
import com.ibm.workplace.wcm.api.*;
import com.ibm.workplace.wcm.api.exceptions.*;

/**
```

```
 * Abstract class containing helper methods to determine the current WCM context.
 * This class can not be used directly. When in portal code please use PortalWCMContextHelper,
 * when in Portlet code please use PortletWCMContextHelper.
 */

public abstract class WCMContextHelper {

   private final ContentMappingInfoHome contentMappingInfoHome;
   private final WebContentService webContentService;

   /**
    * Initializes the WCMContextHelper.
    *
    * @throws NamingException
    */
   public WCMContextHelper() throws NamingException {
      // this initialization needs to be done only once.
      final Context ctx = new InitialContext();
      contentMappingInfoHome = (ContentMappingInfoHome) ctx.lookup(ContentMappingInfoHome.JNDI_NAME);
      webContentService = (WebContentService) ctx.lookup("portal:service/wcm/WebContentService");
   }

    /**
     * Gets the default content mapping of the current page.
     *
     * @param currentPage The content node of the current page
     *
     * @return String representation of the content path that is mapped to the page
     *
     * @throws ContentMappingDataBackendException
     * @throws OperationFailedException
     * @throws ServiceNotAvailableException
     * @throws DocumentIdCreationException
     * @throws IllegalDocumentTypeException
     * @throws DocumentRetrievalException
     */
    @SuppressWarnings({ "rawtypes", "unchecked" })
    protected String getContentMapping(final ContentNode currentPage)
        throws ContentMappingDataBackendException, ServiceNotAvailableException,
        OperationFailedException, DocumentIdCreationException, DocumentRetrievalException,
        IllegalDocumentTypeException {

        //get the default mapping for the current page
        final ContentMappingInfo cmInfo = contentMappingInfoHome.getContentMappingInfo(currentPage);
        final ContentMapping cm = cmInfo.getDefaultContentMapping();
        String contentPath = null;
        if (cm != null) {
          final String contentID = cm.getContentID();
          if(contentID == null) {
              contentPath = cm.getContentPath();
          } else {
              //find the content path to the ID
              final Repository repository = webContentService.getRepository();
              final Workspace workspace = repository.getWorkspace();
              try {
                 final DocumentId docID = workspace.createDocumentId(contentID);
                 contentPath = workspace.getPathById(docID, true, true);
              } finally {
                 repository.endWorkspace();
              }
          }
        }
        return contentPath;
    }

    /**
     * Gets the content node of the current page
     *
     * @param navSelectionModel The navigation selection model
     *
     * @return {@link ContentNode} of the current page
     */
    protected ContentNode getCurrentPageNode(final NavigationSelectionModel<NavigationNode> navSelectionModel) {
       ContentNode result = null;
       final NavigationNode currentNavNode = navSelectionModel.getSelectedNode();
       if(currentNavNode!= null) {
         result = currentNavNode.getContentNode();
       }
       return result;
    }

   /**
    * Assembles a proper content path that resembles 'library/site/sitearea/content'. The method
    * takes care of placing path separators where necessary. The returned content path does not
    * contain path separators as first or as last character.
    *
    * @param contentMapping
    *           The content mapping of the current page. Must not be <code>null</code>.
    * @param pathInfo
    *           The value of the path-info public render parameter for the current page. Must not be
```

```
     *           <code>null</code>.
     * @return A fully-qualified content path.
     */
    protected String assembleContentPath(final String contentMapping, final String [] pathInfo) {
       final StringBuilder result = new StringBuilder();
       // add the context mapping of the page w/o trailing forward slash
       if (contentMapping.charAt(contentMapping.length() - 1) == '/') {
          result.append(contentMapping, 0, contentMapping.length() - 1);
       } else {
          result.append(contentMapping);
       }
       // add all parts of path-info separated by forward slashes
       for (final String pathInfoFragment : pathInfo) {
          if (pathInfoFragment != null && pathInfoFragment.length() > 0) {
             // add leading forward slash before each fragment
             result.append('/');
             // add the path-info fragment
             result.append(pathInfoFragment);
          }
       }
       return result.toString();
    }
}
```

REST service for Web Content Manager

Application developers can use Representational State Transfer (REST) services to work with Web Content Manager. The REST service for Web Content Manager provides authoring access to content items and elements. The service follows the Atom Publication Protocol, and Atom feeds, and entries are accessible in XML (application/atom+xml) and JSON (application/json) format.

"Getting started with the REST service for Web Content Manager"
Before getting started with the REST service for Web Content Manager you should become familiar with how it works and how to use it.

"How to use REST with Web Content Manager items" on page 3308
Different processes are used when items are created and updated by using REST.

"REST content formats for components and elements" on page 3309
When you use REST with components or elements, use these content formats. These examples can be used as templates for your own REST solutions.

"REST Query service for web content" on page 3316
The REST service for Web Content Manager comes with a defined set of query parameters. You can also define your own query parameters in a white list. You can also predefine a query to run more complex searches, and control the allowable filters on these searches by using a white list.

"How to manage web content items by using REST" on page 3324
You can use the Web Content Manager REST Service to create, read, update, and delete the following item types.

"Reference material for the Web Content Manager REST service" on page 3385
Reference material for REST response codes, links, media types, and attachments.

Getting started with the REST service for Web Content Manager

Before getting started with the REST service for Web Content Manager you should become familiar with how it works and how to use it.

The REST service for Web Content Manager is a collection of web services that are compliant with the Atom Publishing Protocol. They provide access to web content, including versions and workflow states, through HTTP. The service is designed according to the REST (REpresentational State Transfer) architectural style.

REST services make it easy to build interactive content, which can be modified directly by your site users. Responsive, integrated editing tools can be created by embedding HTML and JavaScript in web content components, which bind to the REST service to display or update content asynchronously. (Ajax)

HTTP makes integration with remote clients easier than with a traditional API. Web Content Manager functions can be visible to remote systems without adding more server-side components, such as JSP, to access Java APIs. HTTP allows these services to work seamlessly with your infrastructure including firewall, proxy servers, and caches.

Note: Any examples in this section that contain incomplete XML, or XML without namespace declarations, use the following declarations:

- xmlns:atom="http://www.w3.org/2005/Atom"
- xmlns:app="http://www.w3.org/2007/app"
- xmlns:wcm="http://www.ibm.com/xmlns/wcm"

Service entry points

The URLs, which comprise the REST service can change from release to release, or even with minor updates. Therefore, it is recommended to never bookmark, or generate a URL unless it is for a defined entry point.

Atom publishing protocol service document

`/wps/mycontenthandler/model/service`

This service document includes the entry points for all portal REST services. When you browse for content, you must first retrieve the service document. The AtomPub service document describes the top-level collections in an APP service. These collections represent libraries and other types of content accessible through the service.

Web content queries can be stored in certain collections. This allows administrators to limit the scope and structure of queries, and bind them to specific URLs, which all authenticated users can access to retrieve the results as an Atom feed.

POC Service

If a specific content item is known, it can be accessed directly through the POC service, or the POC service can be used to look up an appropriate URL for the content. The identity of a piece of content in the REST service is represented by its POC URI. The POC URI can be found in the ID element of the Atom Entry documents, which represents the content item.

Queries

While queries can be stored within the REST service, they can also be run directly through a single location:

`/wcmrest/query`

This flexibility is subject to security controls to prevent users from inadvertently overloading production servers with complex queries.

REST Service Access Levels

To use the REST service, for Web Content Manager a client user be assigned the "user" role or higher in the *WCM REST SERVICE* virtual resource. All authenticated users are assigned the "user" role by default.

An administrator can edit the *WCM REST SERVICE* virtual resource. Click the **Administration menu** icon. Then, click **Access** > **Resource Permissions**. Then, click **Virtual Resources**.

Table 484. REST user roles

Header	Header
User	Users assigned the "user" role can: • work with web content items and run defined queries.
Editor	Users assigned the "editor" role can: • work with web content items and run defined queries. • Run custom queries through the following path: `/wcmrest/query`
Manager	Users assigned the "manager" role can: • work with web content items and run defined queries. • Run custom queries through the following path: `/wcmrest/query` • create, read, update, and delete defined queries.

How to use REST with Web Content Manager items

Different processes are used when items are created and updated by using REST.

Item types

A complete list of supported item types are documented here: "REST Item Types" on page 3397.

Content representations

Atom entry documents

Each web content item has an associated Atom entry document. The entry document provides access to the common metadata properties of the item, such as title and description, along with links to related items, and access control information.

The default media type of an entry document is `application/atom+xml`. However, `application/json` representations can also be obtained. Entry documents can be retrieved by running an `HTTP GET` to the entry resource. Entry resource URIs can be obtained from:

- The service document
- Links in other entry documents
- Links in feed documents
- Using POC to resolve an ID

Example links to entry documents:

```
<link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteArea/c6b00ee6-d628-4cbd-9e65-15c90f2093a6"/>
<link rel="parent" href="/wps/mycontenthandler/!ut/p/wcmrest/item/ae6a3632-a1b5-456a-866e-e9baab84fe29"/>
<link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/item/54a68ca2-c550-4385-966f-b0b612147547"/>
```

Media resources

In addition to the entry document, most items also have a media resource that is associated with them. The media resource exists to store the content

of the item, for example, HTML or an image. The media resource location is found in the edit-media link of an items entry document. Media resource URLs support HTTP GET and PUT.

```
<link rel="edit-media" type="text/html"
href="/wps/mycontenthandler/!ut/p/wcmrest/PresentationTemplate/02da6e9d-20ca-4c54-ae4a-f1114fa8e948"/>
```

Content Negotiation

When a resource supports multiple representations with different content types, clients can use content negotiation to request a specific representation. For example, entry documents are available in `application/atom+xml; type=entry` and application/json. There are two ways a client can specify the media type that the client can accept:

HTTP Accept header

```
Accept: application/atom+xml
```

Request parameter mime-type

```
GET /wps/mycontenthandler/!ut/p/wcmrest/SiteArea/c6b00ee6-d628-4cbd-
9e65-15c90f2093a6?mime-type=application%2Fjson HTTP/1.0
```

Note: The methods that are used to specify the accept type work for all supported media types as listed in "Supported media types" on page 3392.

Item path

The path of a requested item can be included by specifying the `options=item-path` URL parameter. For example:

```
HTTP 1.1 GET /wps/mycontenthandler/wcmrest/SiteArea/c0b72020-10b7-4197-a436-62a1d94ce03f?options=item-path

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm">
    ...
    <wcm:path>
        <wcm:pathElement>
            <wcm:title xml:lang="en-GB">rest_query_library</wcm:title>
            <wcm:name>rest_query_library</wcm:name>
            <wcm:link label="Read" rel="alternate" xml:lang="en-GB" href="/wps/mycontenthandler/wcmrest/Library/69452890-2f40-4ce6-90af-e65620a552af"/>
        </wcm:pathElement>
        <wcm:pathElement>
            <wcm:title xml:lang="en-GB">site_b</wcm:title>
            <wcm:name>site_b</wcm:name>
            <wcm:link label="Read" rel="alternate" xml:lang="en-GB" href="/wps/mycontenthandler/wcmrest/SiteArea/c44dc8e1-eea5-4262-842c-562788a34461"/>
        </wcm:pathElement>
    </wcm:path>
...
</entry>
```

Example entry

This code is an example in `application/atom+xml` format:

```
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <id>3aabfe14-cc9e-4eb5-ad06-d4fc8fd2f1df</id>
    <title>SampleHTMLComponent</title>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/LibraryHTMLComponent/3aabfe14-cc9e-4eb5-ad06-d4fc8fd2f1df"/>
    <link rel="edit-media" type="text/html" href="/wps/mycontenthandler/!ut/p/wcmrest/LibraryHTMLComponent/3aabfe14-cc9e-4eb5-ad06-d4fc8fd2f1df"/>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/item/c98d11e1-7f2a-480e-9aac-40eb1949cbda"/>
    <link rel="create-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/3aabfe14-cc9e-4eb5-ad06-d4fc8fd2f1df/create-draft"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/3aabfe14-cc9e-4eb5-ad06-d4fc8fd2f1df/change-to-draft"/>
    <link rel="versions" href="/wps/mycontenthandler/!ut/p/wcmrest/item/3aabfe14-cc9e-4eb5-ad06-d4fc8fd2f1df/versions"/>
    <link rel="add-attachment" href="/wps/mycontenthandler/!ut/p/wcmrest/LibraryHTMLComponent/3aabfe14-cc9e-4eb5-ad06-d4fc8fd2f1df/attachments"/>
    <updated>2011-05-30T02:05:44.574Z</updated>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!wtpqWRz_9ePiiVHk1Rw2cw/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!wtpqWRz_9ePiiVHk1Rw2cw/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:name>SampleHTMLComponent</wcm:name>
    <wcm:type>LibraryHTMLComponent</wcm:type>
    <wcm:state>PUBLISHED</wcm:state>
   </entry>
```

REST content formats for components and elements

When you use REST with components or elements, use these content formats. These examples can be used as templates for your own REST solutions.

Component reference component or element

XML

```
<content type="application/vnd.ibm.wcm+xml">
   <reference>/wps/mycontenthandler/wcmrest/LibraryShortTextComponent/c63aaf55-7d32-42e3-9bd
</content>
```

JSON

```
"content" :
{
   "type" : "application/vnd.ibm.wcm+xml",
   "reference" : "/wps/mycontenthandler/wcmrest/LibraryShortTextComponent/c63aaf55-7d32-42e3
}
```

Date and Time component or element

XML

```
<content type="application/vnd.ibm.wcm+xml">
   <date type="DateTime">2014-06-18T14:45:00.000Z</date>
</content>
```

JSON

```
"content" :
{
   "type" : "application/vnd.ibm.wcm+xml",
   "date" :
   {
      "type" : "DateTime",
      "value" : "Wed, 18 Jun 2014 14:45:00.000Z"
   }
}
```

File resource component or element for modifying

XML

```
<content type="application/vnd.ibm.wcm+xml">    <wcm:binaryresource type="image/jpg" fileNam
</content>
```

JSON

```
"content" :
{
   "type" : "application/vnd.ibm.wcm+xml",
   "binaryresource" :
   {
      "type" : "text/javascript",
      "fileName" : "testFile.js",
      "value" : "VGhpcyBpcyBub3QgZ3JlYXQgdGVzdCBkYXRhLi4uLg=="
   }
}
```

File resource component or element for reading

XML

```
<content type="application/vnd.ibm.wcm+xml">
   <resourceUri type="image/png">/wps/wcm/myconnect/60e94008-22a6-4e99-bdbf-864a1b12442f/Scr
</data>
```

JSON

```
"content" :
{
   "type" : "application/vnd.ibm.wcm+xml",
   "resourceUri" :
   {
      "type" : "image/png",
      "value" : "/wps/wcm/myconnect/60e94008-22a6-4e99-bdbf-864a1b12442f/Screenshot+from+201
   }
}
```

HTML component or element

XML

```
<content type="text/html">
   <![CDATA[
      <div class="lotusui30">
      <h2 style="display:block;">
         <div>
            <div class="lotusLeft">
               [EditableProperty context="current" type="content" format="div" field="titl
               [Property context="current" type="content" field="title"]
               [/EditableProperty]
           </div>
         </div>
     </h2>
     </div>
   ]]&gt;
</content>
```

JSON

```
"content" :
{
   "type" : "text/html",
   "value" : "this is some html text"
}
```

Image component or element for reading

XML

```
<content type="application/vnd.ibm.wcm+xml">
   <wcm:image xmlns="http://www.ibm.com/xmlns/wcm">
      <dimension height="16" width="16" border="0"/>
      <altText></altText>
      <tagName></tagName>
      <resourceUri type="image/gif">/wps/wcm/myconnect/5e7dd2e9-8a94-4964-a8d4-3a467be462c
   </wcm:image>
</content>
```

JSON

```
"content" :
{
   "type" : "application/vnd.ibm.wcm+xml",
   "image" :
   {
      "dimension" :
      {
         "height" : "50",
         "width" : "50",
         "border" : "1"
      },
      "altText" : "",
      "tagName" : "",
      "fileName" : "Selection_009.png",
      "resourceUri" :
      {
         "type" : "image/png",
         "value" : "/wps/wcm/myconnect/9d2a5e32-d22e-4b1d-a02d-23b54998d024/Selection_009.
      }
   }
}
```

Image component or element for updating

XML

```
<content type="application/vnd.ibm.wcm+xml">
   <wcm:image xmlns="http://www.ibm.com/xmlns/wcm">
      <dimension height="16" width="16" border="0"/>
```

```
      <altText></altText>
      <tagName></tagName>
      <wcm:binaryresource type="image/jpg" fileName="image.jpg">
         VGhpcyBpcyBub3QgZ3JlYXQgdGVzdCBkYXRhLi4uLg==&#xD;
      </wcm:binaryresource>
   </wcm:image>
</content>
```

JSON

```
"content" :
{
   "type" : "application/vnd.ibm.wcm+xml",
   "image" :
   {
      "dimension" :
      {
         "height" : "50",
         "width" : "50",
         "border" : "1"
      },
      "altText" : "",
      "tagName" : "",
      "fileName" : "Selection_009.png",
      "binaryresource" :
      {
         "type" : "text/javascript",
         "fileName" : "testFile.js",
         "value" : "VGhpcyBpcyBub3QgZ3JlYXQgdGVzdCBkYXRhLi4uLg=="
      }
   }
}
```

JSP component or element

XML

```
<content type="application/vnd.ibm.wcm+xml">
   <jsp>
      <path>/test.jsp</path>
      <errorMessage>Failed to locate the JSP file</errorMessage>
   </jsp>
</content>
```

JSON

```
"content" :
{
   "type" : "application/vnd.ibm.wcm+xml",
   "jsp" :
   {
      "path" : "/test.jsp",
      "errorMessage" : "Failed to locate the JSP file"
   }
}
```

Link component or element

XML

```
<content type="application/vnd.ibm.wcm+xml">
   <linkElement>
      <destination type="content" allowClear="false" queryString="">
         /wps/mycontenthandler/wcmrest/Content/51be861b-12dd-4846-945c-d8c2627299d6
      </destination>
      <display type="title"></display>
      <description useDestination="true"></description>
      <target>None</target>
      <additionalAttributes></additionalAttributes>
   </linkElement>
</content>
```

JSON

```
"content" :
{
   "type" : "application/vnd.ibm.wcm+xml",
   "linkElement" :
   {
      "destination" :
      {
         "type" : "content",
         "allowClear" : false,
         "queryString" : "",
         "value" : "/wps/mycontenthandler/wcmrest/Content/51be861b-12dd-4846-945c-d8c26272
      },
      "display" :
      {
         "type" : "title"
      },
      "description" :
      {
         "useDestination" : true,
         "value" : ""
      },
      "target" : "None",
      "additionalAttributes" : ""
   }
}
```

Numeric component or element

XML - floating point

```
<content type="application/vnd.ibm.wcm+xml">
   <wcm:double>1.01</wcm:double>
</content>
```

XML - whole number

```
<content type="application/vnd.ibm.wcm+xml">
   <wcm:integer>20</wcm:integer>
</content>
```

JSON - floating point

```
"content" :
{
   "type" : "application/vnd.ibm.wcm+xml",
   "double" : 12.1
}
```

JSON - whole number

```
"content" :
{
   "type" : "application/vnd.ibm.wcm+xml",
   "double" : 20
}
```

Option selection element

XML

```
<content type="application/vnd.ibm.wcm+xml">
   <optionselection>
      <displaytype>Automatic</displaytype>
      <selection>UserDefined</selection>
      <options mode="Singleselect">
         <option selected="false" id="A">A</option>
         <option selected="false" id="B">B</option>
         <option selected="false" id="C">C</option>
```

```
            <option selected="false" id="D">D</option>
         </options>
      </optionselection>
   </content>
```

JSON

```
"content" :
{
   "type" : "application/vnd.ibm.wcm+xml",
   "optionselection" :
   {
      "displaytype" : "Automatic",
      "selection" : "UserDefined",
      "options" :
      {
         "mode" : "Singleselect",
         "option" :
         [
            {
               "selected" : false,
               "id" : "A",
               "value" : "A"
            },
            {
               "selected" : false,
               "id" : "B",
               "value" : "B"
            },
            {
               "selected" : false,
               "id" : "C",
               "value" : "C"
            },
            {
               "selected" : false,
               "id" : "D",
               "value" : "D"
            }
         ]
      }
   }
}
```

Rich Text component or element

XML

```
<content type="text/html">
   <![CDATA[
      <p dir="ltr">This is some rich text</p>
   ]]&gt;
</data>
```

JSON

```
"content" :
{
   "type" : "text/html",
   "value" : "<p dir=\"ltr\">This is some rich text<\/p>\n"
}
```

Style sheet component or element

XML

```
<content type="application/vnd.ibm.wcm+xml">
   <wcm:stylesheet xmlns="http://www.ibm.com/xmlns/wcm/8.0">
      <mediaType>All</mediaType>
      <type>Persistent</type>
```

```
      <title></title>
      <resourceUri>/wps/wcm/myconnect/cec80b3d-5529-47ed-a769-f85563c7fe66/Design.css?MOD=
   </wcm:stylesheet>
</content>
```

JSON

```
"content" :
{
   "type" : "application/vnd.ibm.wcm+xml",
   "stylesheet" :
   {
      "mediaType" : "All",
      "type" : "Persistent",
      "title" : "",
      "resourceUri" : "/wps/wcm/myconnect/cec80b3d-5529-47ed-a769-f85563c7fe66/Design.css?
   }
}
```

Text or Short Text component or element

XML

```
<content type="text/plain">
   this is some text content
</content>
```

JSON

```
"data" :
{
   "type" : "text/plain",
   "value" : "this is some text content"
}
```

User selection component or element

XML

```
<content type="application/vnd.ibm.wcm+xml">
   <userSelection>
      <user>
         <distinguishedName>uid=WCMUT_Contributor_A,o=defaultWIMFileBasedRealm</distinguis
         <atom:uri>/wps/mycontenthandler/um/users/profiles/Z9eAeJHPGJP86M9EAMMG6K9PAMMG62R
         <atom:name>WCMUT_Contributor_A WCMUT_Contributor_A</atom:name>
      </user>
      <user>
         <distinguishedName>uid=WCMUT_Contributor_B,o=defaultWIMFileBasedRealm</distinguis
         <atom:uri>/wps/mycontenthandler/um/users/profiles/Z9eAeKPD83H16JHC2JM47KPOCJMG6IH
         <atom:name>WCMUT_Contributor_B WCMUT_Contributor_B</atom:name>
      </user>
   </userSelection>
</content>
```

JSON

```
"data" :
{
   "type" : "application/vnd.ibm.wcm+xml",
   "userSelection" :
   {
      "user" :
      [
         {
            "distinguishedName" : "uid=WCMUT_Contributor_A,o=defaultWIMFileBasedRealm",
            "uri" : "/wps/mycontenthandler/um/users/profiles/Z9eAeJHPGJP86M9EAMMG6K9PAMMG6
            "name" : "WCMUT_Contributor_A WCMUT_Contributor_A"
         },
         {
            "distinguishedName" : "uid=WCMUT_Contributor_B,o=defaultWIMFileBasedRealm",
            "uri" : "/wps/mycontenthandler/um/users/profiles/Z9eAeKPD83H16JHC2JM47KPOCJMG6
            "name" : "WCMUT_Contributor_B WCMUT_Contributor_B"
```

```
            }
        ]
    }
}
```

REST Query service for web content

The REST service for Web Content Manager comes with a defined set of query parameters. You can also define your own query parameters in a white list. You can also predefine a query to run more complex searches, and control the allowable filters on these searches by using a white list.

"Defined query Service"
Defined queries are stored queries that can be run and updated as required.

"Query parameters" on page 3319
The following parameters can be used with queries.

Defined query Service

Defined queries are stored queries that can be run and updated as required.

The defined queries feature allows administrators to define a query in XML format through the REST service, and bind that query to a URI. This query is used to obtain the results by sending a GET request to the bound URI.

Create

A defined query is created by sending a POST request to the following URI:

```
/DefinedQueryComponent/
```

For example:

```
HTTP/1.1 POST
http://host:port/wps/mycontenthandler/wcmrest/DefinedQueryComponent
Content-Type: application/atom+xml
<atom:entry xmlns:atom="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <atom:title>defined query title-1712115665</atom:title>
    <wcm:name>query-name</wcm:name>
    <wcm:description>defined query description</wcm:description>
</atom:entry>

201 Created
```

The URI that is bound to this query has the following format:

```
/definedquery/component-name
```

For example, the URI used to obtain the results of the previous query example is:

```
/definedquery/query-name
```

The response to a create operation contains a link relation "query-results" specifying this URI.

```
<atom:link atom:rel="query-results"
atom:href="/wps/mycontenthandler/!ut/p/wcmrest/definedquery/query-name"/>
```

Note: The only way to update this URI is by modifying the name of the component.

Read

To retrieve a list of defined queries, you send a GET request to the following URI:

`/DefinedQueryComponent/`

For example:

```
HTTP/1.1 GET
http://host:port/wps/mycontenthandler/wcmrest/DefinedQueryComponent
Accept-Type: application/atom+xml

<atom:feed xmlns:atom="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <atom:title>Custom Queries</atom:title>
    <atom:updated>2011-07-04T01:19:27.126Z</atom:updated>
    <atom:entry>
        <atom:id>wcmrest:a5d4f72f-a7b7-4576-a7d3-5a4e15c66f01</atom:id>
        <wcm:name>query-one-name</wcm:name>
        <atom:title>Query 1</atom:title>
        <atom:updated>2011-07-04T01:19:27.159Z</atom:updated>
        <atom:link atom:rel="query-results" atom:href="/wps/mycontenthandler/!ut/p/wcmrest/definedquery/query-one-name
        <atom:link atom:rel="edit-media" atom:type="application/vnd.ibm.wcm+xml" atom:href="/wps/mycontenthandler/!ut/p/wcmrest/DefinedQueryComponent/a5d4f72f-a7b7-4576-a7d3-5a4e15c66f01"/>
        <atom:link atom:rel="edit" atom:href="/wps/mycontenthandler/!ut/p/wcmrest/DefinedQueryComponent/a5d4f72f-a7b7-4576-a7d3-5a4e15c66f01"/>
    </atom:entry>
    <atom:entry>
        <atom:id>wcmrest:6276ff18-f370-45eb-89c3-053d335aba88</atom:id>
        <atom:title>defined query title-1280236937</atom:title>
        <wcm:name>query-two-name</wcm:name>
        <atom:updated>2011-07-04T01:19:27.167Z</atom:updated>
        <atom:link atom:rel="query-results" atom:href="/wps/mycontenthandler/!ut/p/wcmrest/definedquery/query-two-name"/>
        <atom:link atom:rel="edit-media" atom:type="application/vnd.ibm.wcm+xml" atom:href="/wps/mycontenthandler/!ut/p/wcmrest/DefinedQueryComponent/6276ff18-f370-45eb-89c3-053d335aba88"/>
        <atom:link atom:rel="edit" atom:href="/wps/mycontenthandler/!ut/p/wcmrest/DefinedQueryComponent/6276ff18-f370-45eb-89c3-053d335aba88"/>
    </atom:entry>
</atom:feed>
```

To retrieve the XML of a specific defined query, you send a GET request to the following URI:

`/DefinedQueryComponent/`*`item-uuid`*

query-name

For example:

```
HTTP/1.1 GET
http://host:port/wps/mycontenthandler/wcmrest/DefinedQueryComponent/6276ff18-f370-45eb-89c3-053d335aba88
Accept-Type: application/atom+xml

200 OK

<atom:entry xmlns:atom="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <atom:id>wcmrest:6276ff18-f370-45eb-89c3-053d335aba88</atom:id>
    <wcm:type>DefinedQueryComponent</wcm:type>
    <atom:title>defined query title-1460391124</atom:title>
    <wcm:name>query-name</wcm:name>
    <atom:updated>2011-07-04T01:34:01.051Z</atom:updated>
    <wcm:description>defined query description</wcm:description>
    <atom:link atom:rel="edit" atom:href="/wps/mycontenthandler/!ut/p/wcmrest/DefinedQueryComponent/6276ff18-f370-45eb-89c3-053d335aba88"/>
    <atom:link atom:rel="edit-media" atom:type="application/vnd.ibm.wcm+xml" atom:href="/wps/mycontenthandler/!ut/p/wcmrest/DefinedQueryComponent/6276ff18-f370-45eb-89c3-053d335aba88"/>
    <atom:link atom:rel="query-results" atom:href="/wps/mycontenthandler/!ut/p/wcmrest/definedquery/query-name"/>
</atom:entry>
```

To retrieve the raw data of a specific defined query, send a GET request to the edit-media link relation:

`/DefinedQueryComponent/item-id`

For example:

```
HTTP/1.1 GET
http://host:port/wps/mycontenthandler/wcmrest/DefinedQueryComponent/items-id
Accept-Type: application/vnd.ibm.wcm+xml

200 OK

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<definedQuery restrictParameters="true" page="1" pageSize="10" depth="DESCENDANTS">
    <select>
        <typeEquals>
            <type>com.ibm.workplace.wcm.api.Content</type>
        </typeEquals>
        <nameLike>
            <name>article%</name>
        </nameLike>
        <titleLike>
            <title>product%</title>
        </titleLike>
    </select>
    <allowParameters>
        <parameter>lastmodifiedbefore</parameter>
        <parameter>workflowid</parameter>
        <parameter>createdbefore</parameter>
        <parameter>authoringtemplateid</parameter>
    </allowParameters>
</definedQuery>
```

Update

To update the metadata of a query, use a PUT request that contains the new specification to the following URI:

`/DefinedQueryComponent/item-uuid`

For example:

```
HTTP/1.1 PUT
http://host:port/wps/mycontenthandler/wcmrest/DefinedQueryComponent/6276ff18-f370-45eb-89c3-053d335aba88
Content-Type: application/atom+xml
(... atom data ... )

200 OK
```

To update the raw data of a query that specifies the parameters that are used to conduct the query, use a PUT request that contains the new specification to the following URI:

`/DefinedQueryComponent/item-uuid`

For example:

```
HTTP/1.1 PUT
http://host:port/wps/mycontenthandler/wcmrest/DefinedQueryComponent/6276ff18-f370-45eb-89c3-053d335aba88
Content-Type: application/vnd.ibm.wcm+xml
(... xml data ... )

200 OK
```

Delete

To delete a defined query, send a DELETE request to the following URI:

`/DefinedQueryComponent/item-id`

For example:

```
HTTP/1.1 DELETE
http://host:port/wps/mycontenthandler/wcmrest/definedquery/6276ff18-f370-45eb-89c3-053d335aba88

200 OK
```

White List

The white list is a list of extra query parameters that can be used to refine the scope of the defined query. These parameters are the only parameters that have an effect when appended to the bound URI.

For example, if the white list includes a parameter of `name`, the following request returns the results of the defined query that have the name "hello world":

```
HTTP/1.1 GET
http://host:port/wps/mycontenthandler/wcmrest/query-name?name=hello+world

200 OK
```

Query parameters

The following parameters can be used with queries.

Table 485. Query parameters

Parameter	Details and examples of parameters that can be added to queries
`approver`	This parameter is used to query items with a specific approver. A user ID must be specified when this parameter is used. For example: `?approver=userid`
`authoringtemplateid`	This parameter is used to query items with a specific authoring template ID. For example: `?authoringtemplateid=wcmrest:18cfc80c-a490-4d75-9057-fed3db89de53`
`author`	This parameter is used to query items with a specific author. A user UID must be specified when this parameter is used. For example: `?author=uid=WCMUT_Editor_A,o=defaultWIMFileBasedRealm`
`categoryid`	This parameter is used to query items with a specific category ID. For example: `?categoryid=wcmrest:18cfc80c-a490-4d75-9057-fed3db89de53`
`createdafter`	This parameter is used to query items that are created after a specific date and time. For example: `?createdafter=2011-01-11T11:43:29.0150Z`
`createdbefore`	This parameter is used to query items that are created before a specific date and time. For example: `?createdbefore=2011-01-11T11:43:29.0150Z`
`creator`	This parameter is used to query items with a specific creator. A user UID must be specified when this parameter is used. For example: `?creator=uid=WCMUT_Editor_A,o=defaultWIMFileBasedRealm`
`dateformat`	This parameter is used to define the date format of query parameters. For example: `?dateformat=mm-dd-yyyy&createdbefore=12-31-2011` If a date format not specified, then the default format *yyyy-MM-dd'T'HH:mm:ssz* is used.

Table 485. Query parameters (continued)

Parameter	Details and examples of parameters that can be added to queries
depth	This parameter is used with the parentid and is used to define whether to search for all descendants of a parent, or just the immediate children of a parent item. For example, to query only the immediate children of an item, you add &depth=CHILDREN to the query: ?parentid=wcmrest:18cfc80c-a490-4d75-9057-fed3db89de53&depth=CHILDREN To query all descendants of an item, you add &depth=DESCENDANTS to the query: ?parentid=wcmrest:18cfc80c-a490-4d75-9057-fed3db89de53&depth=DESCENDANTS
expireafter	This parameter is used to query items that are expired after a specific date and time. For example: ?expireafter=2011-01-11T11:43:29.0150Z
expirebefore	This parameter is used to query items that were expired before a specific date and time. For example: ?expirebefore=2011-01-11T11:43:29.0150Z
id	This parameter is used to query an item with a specific ID. For example: ?id=wcmrest:18cfc80c-a490-4d75-9057-fed3db89de53
keyword	This parameter is used to query items that are profiled with a specific keyword. For example: ?keyword=keywordValue
lastmodifiedafter	This parameter is used to query items that were last modified after a specific date and time. For example: ?lastmodifiedafter=2011-01-11T11:43:29.0150Z
lastmodifiedbefore	This parameter is used to query items that were last modified before a specific date and time. For example: ?lastmodifiedbefore=2011-01-11T11:43:29.0150Z
lastmodifier	This parameter is used to query items that were last modified by a specific user. A user UID must be specified when this parameter is used. For example: ?lastmodifier=uid=WCMUT_Editor_A,o=defaultWIMFileBasedRealm
libraryid	This parameter is used to query items that are stored in a specific library. For example: ?libraryid=wcmrest:18cfc80c-a490-4d75-9057-fed3db89de53

Table 485. Query parameters (continued)

Parameter	Details and examples of parameters that can be added to queries
`namelike`	This parameter is used as a wildcard query for items with names like the specified `namelike` parameter. For example: `?namelike=nameApproxValue%` **Note:** Libraries cannot be queried by using this parameter.
`name`	This parameter is used to query an item with a specific name. For example: `?name=nameValue` **Note:** Libraries cannot be queried by using this parameter.
`owner`	This parameter is used to query items with a specific owner. A user UID must be specified when this parameter is used. For example: `?owner=uid=WCMUT_Editor_A,o=defaultWIMFileBasedRealm`
`pagesize`	This parameter is used to restrict the number of items that are returned by a query to a set number. It can be used with the `page` parameter to return specific pages of results. For example, to restrict the number of queries to be returned to 5: `?type=PresentationTemplate&pagesize=5`
`page`	This parameter is used with the `pagesize` parameter to define what set of results to display. For example, if `pagesize` is set to 5, and the `page` parameter is set to 2, then only results 6 - 10 are displayed. For example: `?type=PresentationTemplate&pagesize=5&page=2`
`parentid`	This parameter is used to query items that are the children of a specific parent item. For example: `?parentid=wcmrest:18cfc80c-a490-4d75-9057-fed3db89de53` You can use the `depth` parameter to define whether to search for all descendants of a parent, or just the immediate children of a parent item.
`projectid`	This parameter is used to query items that are linked to a specific project. For example: `?projectid=wcmrest:18cfc80c-a490-4d75-9057-fed3db89de53`

Table 485. Query parameters (continued)

Parameter	Details and examples of parameters that can be added to queries
projectstate	This parameter is used to query items that are linked to a project with a specific state. The following values can be used with this parameter: • ACTIVE • SYNDICATING • PENDING • PUBLISHING • PUBLISHED • PUBLISHED_FAILED For example, to query items that are linked to projects with a state of "active" you would use the following query: ?projectstate=ACTIVE
publishafter	This parameter is used to query items that are published after a specific date and time. For example: ?publishafter=2011-01-11T11:43:29.0150Z
publishbefore	This parameter is used to query items that were published before a specific date and time. For example: ?publishbefore=2011-01-11T11:43:29.0150Z
sort	The sort parameter is appended to queries to determine how query results are sorted. The following values can be used with the sort parameter. • author • created • modified • name • title • parents • position The values _ascending or _descending are appended to the query to determine sort order. For example, to sort a presentation template query in ascending order of creation, you would use: ?type=PresentationTemplate&sort=created_ascending To sort a presentation template query in descending order of creation, you would use: ?type=PresentationTemplate&sort=created_descending If _ascending or _descending are not specified, the results as displayed in ascending order.

Table 485. Query parameters (continued)

Parameter	Details and examples of parameters that can be added to queries
`state`	This parameter is used to query items that are in a specific state. The following values can be used with this parameter: • `DRAFT` • `PUBLISHED` • `EXPIRED` For example: `?state=PUBLISHED`
`titlelike`	This parameter is used as a wildcard query for items with titles like the specified `titlelike` parameter. For example: `?titlelike=nameApproxValue%`
`title`	This parameter is used to query an item with a specific title. For example: `?title=titleValue`
`type`	This parameter is used to query items of a specific item type. For example, to query a list of components: `?type=LibraryHTMLComponent`
`workflowid`	This parameter is used to query items that use a specific workflow. For example: `?workflowid=wcmrest:8d25860b-7a5c-4015-9cd5-bdcc60ce14bb`
`workflowstageid`	This parameter is used to query items that are currently active within a specific workflow stage. For example: `?workflowstageid=wcmrest:18cfc80c-a490-4d75-9057-fed3db89de53`

How to use multiple parameters

- Multiple instances of the same parameter type in a query can be specified only as "OR" queries, with the following exceptions:

  Only one value allowed

  Only one instance of the following parameters can be used in a single query. If multiple instances are used, only the first instance is used by the query:

  - `dateformat`
  - `depth`
  - `page`
  - `pagesize`

  "AND" queries allowed

  The following queries can be used as "AND" queries:

  - `createdafter`
  - `createdbefore`
  - `expireafter`
  - `expirebefore`

- `lastmodifiedafter`
- `lastmodifiedbefore`
- `publishafter`
- `publishbefore`

Sort values

Sort values are comma-separated. For example:

```
?sort=created_ascending,title_descending
```

How to manage web content items by using REST

You can use the Web Content Manager REST Service to create, read, update, and delete the following item types.

"How to use REST with libraries" on page 3325
You can use the Web Content Manager REST service to create, read, update, and delete libraries.

"How to use REST with components" on page 3327
You can use the Web Content Manager REST service to create, read, update, and delete some types of components.

"How to use REST with elements before version 8.5 CF03" on page 3330
You can use the Web Content Manager REST service to create, read, update, and delete some types of elements that are stored in site areas and content items. All element types are supported.

"How to use REST with elements with version 8.5 CF03 or higher" on page 3332
You can use the Web Content Manager REST service to create, read, update, and delete elements.

CF06 "How to use REST with authoring templates" on page 3337
You can use the Web Content Manager REST service to create, read, update, and delete content templates and site area templates. You can also set default values for items that are created by using these authoring templates.

"How to REST with presentation templates" on page 3346
You can use the Web Content Manager REST service to create, read, update, and delete presentation templates.

"How to use REST with content items" on page 3348
You can use the Web Content Manager REST service to create, read, update, and delete content items.

"How to use REST with site areas" on page 3350
You can use the Web Content Manager REST service to create, read, update, and delete site areas.

"How to use REST with managed pages" on page 3352
You can use the Web Content Manager REST service to read managed pages.

"How to use REST with drafts and workflows" on page 3353
You can use the REST services for Web Content Manager to create drafts, approve items in a workflow, and move items through different stages of a workflow.

"How to use REST with workflow items" on page 3355
You can use the Web Content Manager REST service to create, read, update, and delete workflow items.

"How to use REST with workflow stages" on page 3357
You can use the Web Content Manager REST service to create, read, update, and delete workflow stages.

How to use REST with libraries

You can use the Web Content Manager REST service to create, read, update, and delete libraries.

Create

A library can be created by sending a POST request to the following URI:

```
/Library
```

For example:

```
HTTP/1.1 POST /wps/mycontenthandler/wcmrest/Library
Content-Type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <wcm:name>My Library</wcm:name>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:library xmlns="http://www.ibm.com/xmlns/wcm/8.0">
            <allowDeletion>false</allowDeletion>
            <enabled>true</enabled>
            <language>en</language>
            <includeDefaultItems>true</includeDefaultItems>
        </wcm:library>
    </content>
</entry>

HTTP 201 Created
```

Update

An existing library can be updated by sending a PUT request to the following URI: `/Library/library-id`

For example:

```
HTTP/1.1 PUT /wps/mycontenthandler/wcmrest/Library/ff207ba6-ac9a-4b38-b831-99528ff5067c
Content-Type: application/atom+xml
Accept: application/atom+xml

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:ff207ba6-ac9a-4b38-b831-99528ff5067c</id>
    <title>My New Library Title</title>
    <summary/>
    <wcm:name>my library</wcm:name>
    <wcm:type>Library</wcm:type>
    <updated>2014-10-09T03:18:02.303Z</updated>
    <link rel="self" href="/wps/mycontenthandler/wcmrest/Library/ff207ba6-ac9a-4b38-b831-99528ff5067c" xml:lang="en" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/wcmrest/Library/ff207ba6-ac9a-4b38-b831-99528ff5067c" xml:lang="en" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/wcmrest/Library/ff207ba6-ac9a-4b38-b831-99528ff5067c" xml:lang="en" label="Delete"/>
    <link rel="access-control" href="/wps/mycontenthandler/ac/access:oid:Z6QReDe6JP43OS62BOCJM4C3BE2MMG62RCGJM8COPC2JM47P9D43SOC6BD03RS633" xml:lang="en" label="Access Control"/>
    <link rel="edit-media" href="/wps/mycontenthandler/wcmrest/Library/ff207ba6-ac9a-4b38-b831-99528ff5067c" type="application/vnd.ibm.wcm+xml" xml:lang="en" label="Edit Media"/>
    <link rel="preset-folders" href="/wps/mycontenthandler/wcmrest/Library/ff207ba6-ac9a-4b38-b831-99528ff5067c/preset-folders"/>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:library xmlns="http://www.ibm.com/xmlns/wcm/8.0">
            <allowDeletion>true</allowDeletion>
            <enabled>true</enabled>
            <language>en</language>
        </wcm:library>
    </content>
</entry>

HTTP 200 OK

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:ff207ba6-ac9a-4b38-b831-99528ff5067c</id>
    <title>My New Library Title</title>
    <summary/>
    <wcm:name>my library</wcm:name>
    <wcm:type>Library</wcm:type>
    <updated>2014-10-09T03:18:02.303Z</updated>
    <link rel="self" href="/wps/mycontenthandler/wcmrest/Library/ff207ba6-ac9a-4b38-b831-99528ff5067c" xml:lang="en" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/wcmrest/Library/ff207ba6-ac9a-4b38-b831-99528ff5067c" xml:lang="en" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/wcmrest/Library/ff207ba6-ac9a-4b38-b831-99528ff5067c" xml:lang="en" label="Delete"/>
    <link rel="access-control" href="/wps/mycontenthandler/ac/access:oid:Z6QReDe6JP43OS62BOCJM4C3BE2MMG62RCGJM8COPC2JM47P9D43SOC6BD03RS633" xml:lang="en" label="Access Control"/>
    <link rel="edit-media" href="/wps/mycontenthandler/wcmrest/Library/ff207ba6-ac9a-4b38-b831-99528ff5067c" type="application/vnd.ibm.wcm+xml" xml:lang="en" label="Edit Media"/>
    <link rel="preset-folders" href="/wps/mycontenthandler/wcmrest/Library/ff207ba6-ac9a-4b38-b831-99528ff5067c/preset-folders"/>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:library xmlns="http://www.ibm.com/xmlns/wcm/8.0">
            <allowDeletion>true</allowDeletion>
            <enabled>true</enabled>
            <language>en</language>
        </wcm:library>
    </content>
</entry>
```

Read

An existing library can be read by sending a GET request to the following URI: `/Library/library-id`

For example:

```
HTTP/1.1 GET /wps/mycontenthandler/wcmrest/Library/ff207ba6-ac9a-4b38-b831-99528ff5067c
Accept: application/atom+xml

HTTP 200 OK

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:ff207ba6-ac9a-4b38-b831-99528ff5067c</id>
    <title>My Library</title>
    <summary/>
    <wcm:name>my library</wcm:name>
    <wcm:type>Library</wcm:type>
    <updated>2014-10-09T03:18:02.303Z</updated>
    <link rel="self" href="/wps/mycontenthandler/wcmrest/Library/ff207ba6-ac9a-4b38-b831-99528ff5067c" xml:lang="en" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/wcmrest/Library/ff207ba6-ac9a-4b38-b831-99528ff5067c" xml:lang="en" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/wcmrest/Library/ff207ba6-ac9a-4b38-b831-99528ff5067c" xml:lang="en" label="Delete"/>
    <link rel="access-control" href="/wps/mycontenthandler/ac/access:oid:Z6QReDe6JP43OS62BOCJM4C3BE2MMG62RCGJM8COPC2JM47P9D43SOC6BD03RS633" xml:lang="en" label="Access Control"/>
    <link rel="edit-media" href="/wps/mycontenthandler/wcmrest/Library/ff207ba6-ac9a-4b38-b831-99528ff5067c" type="application/vnd.ibm.wcm+xml" xml:lang="en" label="Edit Media"/>
    <link rel="preset-folders" href="/wps/mycontenthandler/wcmrest/Library/ff207ba6-ac9a-4b38-b831-99528ff5067c/preset-folders"/>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:library xmlns="http://www.ibm.com/xmlns/wcm/8.0">
            <allowDeletion>false</allowDeletion>
            <enabled>true</enabled>
            <language>en</language>
        </wcm:library>
    </content>
</entry>
```

Delete

An existing library can be deleted by sending a DELETE request to the following URI:

```
/Library/library-id
```

For example:

```
HTTP/1.1 DELETE /wps/mycontenthandler/wcmrest/Library/ff207ba6-ac9a-4b38-b831-99528ff5067c

HTTP 200 OK
```

Asynchronous Delete

An existing library can also be deleted asynchronously by using the following URI. The response contains a 'Content-Location' header, containing a URI that can be used to monitor the progress of the delete operation:

```
/Library/library-id?synchronous=true
```

For example:

```
HTTP/1.1 DELETE /wps/mycontenthandler/wcmrest/Library/ff207ba6-ac9a-4b38-b831-99528ff5067c?synchronous=true

HTTP 202 Accepted
Content-Location: /wps/mycontenthandler/wcmrest/Library/fd18e3a0-b96b-4a66-9bcc-f3d142c7837e/wcmTask:com.ibm.workplace.wcm.services.task.DeleteTask:ced02fae-abc7-4ff4-af4d-c08804b32811/delete-status
```

How to use REST with components

You can use the Web Content Manager REST service to create, read, update, and delete some types of components.

The following component types can be used with the Web Content Manager REST service.

Table 486. Component types

Component	API type
Authoring Tools Component	LibraryAuthoringToolsComponent
Date and time component	LibraryDateComponent
File resource component	LibraryFileComponent
HTML Component	LibraryHTMLComponent
Image component	LibraryImageComponent
JSP Component	LibraryJSPComponent
Link Component	LibraryLinkComponent
List Presentation Component	LibraryListPresentationComponent
Menu Component	LibraryMenuComponent
Navigator Component	LibraryNavigatorComponent
Number component	LibraryNumericComponent
Page Navigation Component	LibraryPageNavigationComponent
Personalization Component	LibraryPersonalizationComponent
Reference Component	LibraryReferenceComponent
Rich text component	LibraryRichTextComponent
Search Component	LibrarySearchComponent
Short text component	LibraryShortTextComponent
Stylesheet Component	LibraryStyleSheetComponent
Text component	LibraryTextComponent

Table 486. Component types (continued)

Component	API type
User Selection Component	LibraryUserSelectionComponent
Username Component	LibraryUserNameComponent

Create

A component can be created by sending a POST request to the following URI with an Atom entry that represents the component:

`/<library-component-api-type>`

For example:

```
POST /wps/mycontenthandler/wcmrest/LibraryNumericComponent HTTP/1.0
Content-Type: application/atom+xml

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <title>SampleNumericComponentTitle</title>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/item/c98d11e1-7f2a-480e-9aac-40eb1949cbda"/>
    <wcm:name>SampleNumericComponentName</wcm:name>
    <wcm:description>SampleNumericComponentDescription</wcm:description>
</entry>

HTTP/1.0 201 Created
Content-type: application/atom+xml; type=entry
Content-location: /wps/mycontenthandler/!ut/p/wcmrest/LibraryNumericComponent/0d678334-69ae-4d3a-a525-91bb551e5a18

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <id>0d678334-69ae-4d3a-a525-91bb551e5a18</id>
    <title>SampleNumericComponentTitle</title>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/LibraryNumericComponent/0d678334-69ae-4d3a-a525-91bb551e5a18"/>
    <link rel="edit-media" type="text/plain" href="/wps/mycontenthandler/!ut/p/wcmrest/LibraryNumericComponent/0d678334-69ae-4d3a-a525-91bb551e5a18"/>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/item/c98d11e1-7f2a-480e-9aac-40eb1949cbda"/>
    <link rel="create-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/0d678334-69ae-4d3a-a525-91bb551e5a18/create-draft"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/0d678334-69ae-4d3a-a525-91bb551e5a18/change-to-draft"/>
    <link rel="versions" href="/wps/mycontenthandler/!ut/p/wcmrest/item/0d678334-69ae-4d3a-a525-91bb551e5a18/versions"/>
    <updated>2011-05-30T04:33:40.540Z</updated>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5U175Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5U175Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:name>SampleNumericComponentName</wcm:name>
    <wcm:description>SampleNumericComponentDescription</wcm:description>
    <wcm:type>NUMERIC</wcm:type>
    <wcm:state>PUBLISHED</wcm:state>
</entry>
```

Update

A component can be updated by sending a PUT request to the following URI with an Atom entry that includes the fields of the item that need to be changed.

`/<library-component-api-type>/<itemuuid>`

For example:

```
PUT /wps/mycontenthandler/wcmrest/LibraryNumericComponent/c98d11e1-7f2a-480e-9aac-40eb1949cbda HTTP/1.0
Content-type : application/atom+xml

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <title>SampleNumericComponentTitleUpdated</title>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/item/c98d11e1-7f2a-480e-9aac-40eb1949cbda"/>
    <wcm:name>SampleNumericComponentNameUpdated</wcm:name>
    <wcm:description>SampleNumericComponentDescriptionUpdated</wcm:description>
</entry>


HTTP/1.0 200 OK
Content-type: application/atom+xml; type=entry

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <id>0d678334-69ae-4d3a-a525-91bb551e5a18</id>
    <title>SampleNumericComponentTitleUpdated</title>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/LibraryNumericComponent/0d678334-69ae-4d3a-a525-91bb551e5a18"/>
    <link rel="edit-media" type="text/plain" href="/wps/mycontenthandler/!ut/p/wcmrest/LibraryNumericComponent/0d678334-69ae-4d3a-a525-91bb551e5a18"/>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/item/c98d11e1-7f2a-480e-9aac-40eb1949cbda"/>
    <link rel="create-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/0d678334-69ae-4d3a-a525-91bb551e5a18/create-draft"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/0d678334-69ae-4d3a-a525-91bb551e5a18/change-to-draft"/>
    <link rel="versions" href="/wps/mycontenthandler/!ut/p/wcmrest/item/0d678334-69ae-4d3a-a525-91bb551e5a18/versions"/>
    <updated>2011-05-30T04:38:49.522Z</updated>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5U175Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5U175Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:name>SampleNumericComponentNameUpdated</wcm:name>
```

```
    <wcm:description>SampleNumericComponentDescriptionUpdated</wcm:description>
    <wcm:type>NUMERIC</wcm:type>
    <wcm:state>PUBLISHED</wcm:state>
</entry>
```

Read

A component can be read by sending a GET request to the following URI:
`/<library-component-api-type>/<itemuuid>`

For example:

```
GET /wps/mycontenthandler/wcmrest/LibraryNumericComonent/0d678334-69ae-4d3a-a525-91bb551e5a18 HTTP/1.0


HTTP/1.0 200 OK
Content-type: application/atom+xml; type=entry

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <id>0d678334-69ae-4d3a-a525-91bb551e5a18</id>
    <title>SampleNumericComponentTitleUpdated</title>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/LibraryNumericComponent/0d678334-69ae-4d3a-a525-91bb551e5a18"/>
    <link rel="edit-media" type="text/plain" href="/wps/mycontenthandler/!ut/p/wcmrest/LibraryNumericComponent/0d678334-69ae-4d3a-a525-91bb551e5a18"/>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/item/c98d11e1-7f2a-480e-9aac-40eb1949cbda"/>
    <link rel="create-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/0d678334-69ae-4d3a-a525-91bb551e5a18/create-draft"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/0d678334-69ae-4d3a-a525-91bb551e5a18/change-to-draft"/>
    <link rel="versions" href="/wps/mycontenthandler/!ut/p/wcmrest/item/0d678334-69ae-4d3a-a525-91bb551e5a18/versions"/>
    <updated>2011-05-30T04:38:49.522Z</updated>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5Ul75Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5Ul75Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:name>SampleNumericComponentNameUpdated</wcm:name>
    <wcm:description>SampleNumericComponentDescriptionUpdated</wcm:description>
    <wcm:type>NUMERIC</wcm:type>
    <wcm:state>PUBLISHED</wcm:state>
</entry>
```

Delete

A component can be deleted by sending a DELETE request to the following URI:
`/<library-component-api-type>/<itemuuid>`

For example:

DELETE

```
HTTP/1.1 DELETE
http://host:port/wps/mycontenthandler/wcmrest/LibraryNumericComponent/<itemuuid>
```

Response

```
Status Code :200
Status Message : OK
```

Specifying raw data

The content of a component is accessed from the location that is specified in the HREF attribute of the edit-media link. The link also contains a TYPE attribute that contains the accepted media type for the content. For example:

```
<link rel="edit-media" type="text/plain"
href="/wps/mycontenthandler/!ut/p/wcmrest/LibraryTextComponent/0d678334-69ae-4d3a-a525-91bb551e5a18"/>
```

A complete list of media types are documented here: “Supported media types” on page 3392.

To update content of a library component PUT content in an accepted media type to the edit-media URL. For example:

```
PUT /wps/mycontenthandler/!ut/p/wcmrest/LibraryTextComponent/0d678334-69ae-4d3a-a525-91bb551e5a18 HTTP/1.0
Content-type: text/plain

This is some text to add to the component.


HTTP/1.0 200 OK
```

To retrieve content from a library component GET content from the edit-media URL. For example:

```
GET  /wps/mycontenthandler/!ut/p/wcmrest/LibraryTextComponent/0d678334-69ae-4d3a-a525-91bb551e5a18 HTTP/1.0
Accept: text/plain


HTTP/1.0 200 OK
Content-type: text/plain

This is some text to add to the component.
```

An alternative to specifying the media type in the HTTP accept header is to use the request parameter mime-type. You must URL encode the value. For example:

```
GET  /wps/mycontenthandler/!ut/p/wcmrest/LibraryTextComponent/0d678334-69ae-4d3a-a525-91bb551e5a18?mime-type=text%2Fplain HTTP/1.0


HTTP/1.0 200 OK
Content-type: text/plain

This is some text to add to the component.
```

Related concepts:

"REST content formats for components and elements" on page 3309
When you use REST with components or elements, use these content formats. These examples can be used as templates for your own REST solutions.

How to use REST with elements before version 8.5 CF03

You can use the Web Content Manager REST service to create, read, update, and delete some types of elements that are stored in site areas and content items. All element types are supported.

Create

An element can be created by sending a POST request to the following URI with an Atom entry that represents the title of the element:

```
/[Content|SiteArea]/<parent-uuid>/elements
```

Note: The type of the element to be created must be specified in the type field of the entry that is posted.

For example:

```
POST /wps/mycontenthandler/wcmrest/SiteArea/c6b00ee6-d628-4cbd-9e65-15c90f2093a6/elements HTTP/1.0
Content-type: application/atom+xml

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <title>Number Element Title</title>
    <wcm:name>numericElementName</wcm:name>
    <wcm:type>NumericComponent</wcm:type>
</entry>


HTTP/1.0 201 Created
Content-type: application/atom+xml; type=entry
Content-location: /wps/mycontenthandler/!ut/p/wcmrest/SiteArea/c6b00ee6-d628-4cbd-9e65-15c90f2093a6/elements/numericElementName

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <title>Number Element Title</title>
    <link rel="edit-media" type="text/plain" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteArea/c6b00ee6-d628-4cbd-9e65-15c90f2093a6/elements/numericElementName"/>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteArea/c6b00ee6-d628-4cbd-9e65-15c90f2093a6/elements/numericElementName"/>
    <wcm:name>numericElementName</wcm:name>
    <wcm:type>NumericComponent</wcm:type>
</entry>
```

Update

An element can be updated by sending a PUT request to the following URI with an Atom entry that includes the name and title of the element.

```
/[Content|SiteArea]/<parent-uuid>/elements/<element-name-encoded>
```

For example:

```
PUT /wps/mycontenthandler/wcmrest/SiteArea/c6b00ee6-d628-4cbd-9e65-15c90f2093a6/elements/numericComponentName HTTP/1.0
Content-type: application/atom+xml

<atom:entry xmlns:atom="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <title>numericElementTitleUpdated</title>
```

```
    <wcm:name>numericElementNameUpdated</wcm:name>
</atom:entry>


HTTP/1.0 200 OK
Content-type: application/atom+xml; type=entry

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <title>numericComponentTitleUpdated</title>
    <link rel="edit-media" type="text/plain" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteArea/c6b00ee6-d628-4cbd-9e65-15c90f2093a6/elements/numericElementNameUpdated"/>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteArea/c6b00ee6-d628-4cbd-9e65-15c90f2093a6/elements/numericElementNameUpdated"/>
    <wcm:name>numericElementNameUpdated</wcm:name>
    <wcm:type>NumericComponent</wcm:type>
</entry>
```

Read

An element can be read by sending a GET request to the following URI:
`/[Content|SiteArea]/<parent-uuid>/elements/<element-name-encoded>`

For example:

```
GET /wps/mycontenthandler/wcmrest/Content/c6b00ee6-d628-4cbd-9e65-15c90f2093a6/elements/numericElementName HTTP/1.0


HTTP/1.0 200 OK
Content-type: application/atom+xml; type=entry

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <title>numericElementTitleUpdated</title>
    <link rel="edit-media" type="application/vnd.ibm.wcm+xml" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteArea/c6b00ee6-d628-4cbd-9e65-15c90f2093a6/elements/numericElementName"/>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteArea/c6b00ee6-d628-4cbd-9e65-15c90f2093a6/elements/numericElementName"/>
    <wcm:name>numericElementName</wcm:name>
    <wcm:type>NumericComponent</wcm:type>
</entry>
```

Delete

An element can be deleted by sending a DELETE request to the following URI:
`/[Content|SiteArea]/<parent-uuid>/elements/<element-name-encoded>`

For example:

DELETE

```
HTTP/1.1 DELETE
http://host:port/wps/mycontenthandler/wcmrest/Content/<parent-uuid>/elements/<element-name-encoded>
```

Response

```
Status Code :200
Status Message : OK
```

Specifying raw data

The content of an element is accessed from the media resource that is specified in the HREF attribute of the edit-media link. The link also contains a TYPE attribute that contains the accepted media type of the content. For example:

```
<link rel="edit-media" type="text/html"
href="/wps/mycontenthandler/!ut/p/wcmrest/SiteArea/c6b00ee6-d628-4cbd-9e65-15c90f2093a6/elements/heading"/>
```

A complete list of media types are documented here: “Supported media types” on page 3392.

To update content of an element PUT content in an accepted media type to the edit-media URL. For example:

```
PUT /wps/mycontenthandler/!ut/p/wcmrest/SiteArea/c6b00ee6-d628-4cbd-9e65-15c90f2093a6/elements/heading HTTP/1.0
Content-type: text/html

<h1>Heading Text</h1>


HTTP/1.0 200 OK
```

To retrieve content from a library component GET content from the edit-media URL. For example:

```
GET  /wps/mycontenthandler/!ut/p/wcmrest/SiteArea/c6b00ee6-d628-4cbd-9e65-15c90f2093a6/elements/heading HTTP/1.0
Accept: text/html


HTTP/1.0 200 OK
Content-type: text/plain

<h1>Heading Text</h1>
```

An alternative to specifying the media type in the HTTP accept header, is to use the request parameter mime-type. You must URL encode the value. For example:

```
GET  /wps/mycontenthandler/!ut/p/wcmrest/SiteArea/c6b00ee6-d628-4cbd-9e65-15c90f2093a6/elements/heading?mime-type=text%2Fhtml HTTP/1.0


HTTP/1.0 200 OK
Content-type: text/plain

<h1>Heading Text</h1>
```

Related concepts:

"REST content formats for components and elements" on page 3309
When you use REST with components or elements, use these content formats. These examples can be used as templates for your own REST solutions.

How to use REST with elements with version 8.5 CF03 or higher

You can use the Web Content Manager REST service to create, read, update, and delete elements.

Create

You can use the Web Content Manager REST service to create, read, update, and delete all element types that belong to one of the following element container types:

- Content
- Site area
- The default content of a content template
- The default site area of a site area template

There are two ways to run these operations. The first is to address each element individually, and by using a URI and use the HTTP verbs (GET, PUT, POST, DELETE) to manipulate them. You can also update multiple elements with a PUT request.

An element can be created by sending a POST request to one of the following URIs with an Atom entry that represent the title of the element:

Content item
`/Content/content-id/elements`

Site area
`/SiteArea/site-area-id/elements`

The default content of a content template
`/ContentTemplate/template-id/Prototype/elements`

The default content of a site area template
`/SiteAreaTemplate/template-id/Prototype/elements`

Note: The type of the element to be created must be specified in the type field of the entry that is posted.

For example:

```
HTTP/1.1 POST /wps/mycontenthandler/wcmrest/ContentTemplate/12015598-24c4-40f6-9be0-68c52663c03f/Prototype/elements/

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <title>MyDateElement</title>
    <wcm:name>MyDateElement</wcm:name>
    <wcm:type>DateComponent</wcm:type>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:date type="DateTime">2014-08-05T06:01:07.152Z</wcm:date>
    </content>
</entry>

HTTP/1.1 201 Created
Content-Type: application/atom+xml
Content-Location: /wps/mycontenthandler/wcmrest/ContentTemplate/3f6311ae-d277-42d0-869c-eeb7a28ba754/Prototype/elements/MyDateElement

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <title>MyDateElement</title>
    <wcm:name>MyDateElement</wcm:name>
    <wcm:type>TextComponent</wcm:type>
    <link rel="self" href="/wps/mycontenthandler/wcmrest/ContentTemplate/3f6311ae-d277-42d0-869c-eeb7a28ba754/Prototype/elements/MyDateElement" xml:lang="en" label="Read"/>
    <link rel="edit-media" href="/wps/mycontenthandler/wcmrest/ContentTemplate/3f6311ae-d277-42d0-869c-eeb7a28ba754/Prototype/elements/MyDateElement" type="text/plain" xml:lang="en" label="Edit Media"/>
    <link rel="edit" href="/wps/mycontenthandler/wcmrest/ContentTemplate/3f6311ae-d277-42d0-869c-eeb7a28ba754/Prototype/elements/MyDateElement" xml:lang="en" label="Edit"/>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:date type="DateTime">2014-08-05T06:01:07.152Z</wcm:date>
    </content>
</entry>
```

Update

An element can be updated by sending a PUT request to one of the following URIs:

Content item
/Content/content-id/elements/element-name

Site area
/SiteArea/site-area-id/elements/element-name

The default content of a content template
/ContentTemplate/template-id/Prototype/elements/element-name

The default content of a site area template
/SiteAreaTemplate/template-id/Prototype/elements/element-name

For example:

```
HTTP/1.1 PUT /wps/mycontenthandler/wcmrest/ContentTemplate/3f6311ae-d277-42d0-869c-eeb7a28ba754/Prototype/elements/MyDateElement
Content-Type: application/atom+xml

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <title xml:lang="en">MyDateElement</title>
    <wcm:name>MyDateElement with a new name</wcm:name>
    <wcm:type>DateComponent</wcm:type>
    <link rel="self" href="/wps/mycontenthandler/wcmrest/ContentTemplate/3f6311ae-d277-42d0-869c-eeb7a28ba754/Prototype/elements/MyDateElement" xml:lang="en" label="Read"/>
    <link rel="edit-media" href="/wps/mycontenthandler/wcmrest/ContentTemplate/3f6311ae-d277-42d0-869c-eeb7a28ba754/Prototype/elements/MyDateElement" type="application/vnd.ibm.wcm+xml" xml:lang="en" label="Edit Media"/>
    <link rel="edit" href="/wps/mycontenthandler/wcmrest/ContentTemplate/3f6311ae-d277-42d0-869c-eeb7a28ba754/Prototype/elements/MyDateElement" xml:lang="en" label="Edit"/>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:date type="DateTime">2013-08-05T06:01:07.152Z</wcm:date>
    </content>
</entry>

HTTP/1.1 200 OK
Content-Type: application/atom+xml

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <title xml:lang="en">MyDateElement</title>
    <wcm:name>MyDateElement with a new name</wcm:name>
    <wcm:type>DateComponent</wcm:type>
    <link rel="self" href="/wps/mycontenthandler/wcmrest/ContentTemplate/3f6311ae-d277-42d0-869c-eeb7a28ba754/Prototype/elements/MyDateElement" xml:lang="en" label="Read"/>
    <link rel="edit-media" href="/wps/mycontenthandler/wcmrest/ContentTemplate/3f6311ae-d277-42d0-869c-eeb7a28ba754/Prototype/elements/MyDateElement" type="application/vnd.ibm.wcm+xml" xml:lang="en" label="Edit Media"/>
    <link rel="edit" href="/wps/mycontenthandler/wcmrest/ContentTemplate/3f6311ae-d277-42d0-869c-eeb7a28ba754/Prototype/elements/MyDateElement" xml:lang="en" label="Edit"/>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:date type="DateTime">2013-08-05T06:01:07.152Z</wcm:date>
    </content>
</entry>
```

Read

An element can be read by sending a GET request to one of the following URIs:

Content item
/Content/content-id/elements/element-name

Site area
/SiteArea/site-area-id/elements/element-name

The default content of a content template
/ContentTemplate/template-id/Prototype/elements/element-name

The default content of a site area template
/SiteAreaTemplate/template-id/Prototype/elements/element-name

For example:

```
HTTP/1.1 GET /wps/mycontenthandler/wcmrest/ContentTemplate/3f6311ae-d277-42d0-869c-eeb7a28ba754/Prototype/elements/MyDateElement
Accept: application/atom+xml

HTTP/1.1 200 OK
Content-Type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <title xml:lang="en">MyDateElement</title>
    <wcm:name>MyDateElement</wcm:name>
    <wcm:type>DateComponent</wcm:type>
    <link rel="self" href="/wps/mycontenthandler/wcmrest/ContentTemplate/3f6311ae-d277-42d0-869c-eeb7a28ba754/Prototype/elements/MyDateElement" xml:lang="en" label="Read"/>
    <link rel="edit-media" href="/wps/mycontenthandler/wcmrest/ContentTemplate/3f6311ae-d277-42d0-869c-eeb7a28ba754/Prototype/elements/MyDateElement" type="application/vnd.ibm.wcm+xml" xml:lang="en" label="Edit Media"/>
    <link rel="edit" href="/wps/mycontenthandler/wcmrest/ContentTemplate/3f6311ae-d277-42d0-869c-eeb7a28ba754/Prototype/elements/MyDateElement" xml:lang="en" label="Edit"/>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:date type="DateTime">2014-08-05T06:01:07.152Z</wcm:date>
    </content>
</entry>
```

Delete

An element can be deleted by sending a DELETE request to one of the following URIs:

Content item
/Content/content-id/elements/element-name

Site area
/SiteArea/site-area-id/elements/element-name

The default content of a content template
/ContentTemplate/template-id/Prototype/elements/element-name

The default content of a site area template
/SiteAreaTemplate/template-id/Prototype/elements/element-name

For example:

```
HTTP/1.1 DELETE /wps/mycontenthandler/wcmrest/ContentTemplate/3f6311ae-d277-42d0-869c-eeb7a28ba754/Prototype/elements/MyDateElement

HTTP/1.1 200 OK
```

Read feed with multiple elements

Element feeds can be obtained by issuing a GET request to the following URIs:

Content item
/Content/content-id/elements/element-name

Site area
/SiteArea/site-area-id/elements/element-name

The default content of a content template
/ContentTemplate/template-id/Prototype/elements/element-name

The default content of a site area template
/SiteAreaTemplate/template-id/Prototype/elements/element-name

For example:

```
HTTP/1.1 GET /wps/mycontenthandler/wcmrest/Content/4f9dfaa1-a823-4964-9583-8bbd69504595/elements
Accept: application/atom+xml

HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <title xml:lang="en">Elements Feed</title>
    <updated>2014-08-08T04:50:01.839Z</updated>
    <link rel="parent" href="/wps/mycontenthandler/wcmrest/Content/4f9dfaa1-a823-4964-9583-8bbd69504595" xml:lang="en" label="Parent"/>
    <entry>
        <title xml:lang="en">MyRichTextElement</title>
        <wcm:name>MyRichTextElement</wcm:name>
        <wcm:type>RichTextComponent</wcm:type>
        <link rel="alternate" href="/wps/mycontenthandler/wcmrest/Content/4f9dfaa1-a823-4964-9583-8bbd69504595/elements/MyRichTextElement" xml:lang="en" label="Read"/>
        <content type="text/html"><![CDATA[<p dir="ltr">This is some rich text</p>
]]&gt;</content>
    </entry>
    <entry>
        <title xml:lang="en">MyDateComponent</title>
        <wcm:name>MyDateComponent</wcm:name>
        <wcm:type>DateComponent</wcm:type>
        <link rel="alternate" href="/wps/mycontenthandler/wcmrest/Content/4f9dfaa1-a823-4964-9583-8bbd69504595/elements/MyDateComponent" xml:lang="en" label="Read"/>
        <content type="application/vnd.ibm.wcm+xml">
            <wcm:date type="DateTime">2014-08-06T15:00:00.000Z</wcm:date>
        </content>
    </entry>
```

```
    <entry>
        <title xml:lang="en">MyTextComponent</title>
        <wcm:name>MyTextComponent</wcm:name>
        <wcm:type>TextComponent</wcm:type>
        <link rel="alternate" href="/wps/mycontenthandler/wcmrest/Content/4f9dfaa1-a823-4964-9583-8bbd69504595/elements/MyTextComponent" xml:lang="en" label="Read"/>
        <content type="text/plain">This is some text</content>
    </entry>
</feed>
```

Read inline with multiple elements

Element within an entry can be obtained by issuing a GET request to the following URIs:

Content item
/Content/content-id/

Site area
/SiteArea/site-area-id/

The default content of a content template
/ContentTemplate/template-id/Prototype/

The default content of a site area template
/SiteAreaTemplate/template-id/Prototype/

For example:

```
HTTP/1.1 GET /wps/mycontenthandler/wcmrest/Content/4f9dfaa1-a823-4964-9583-8bbd69504595
Accept: application/atom+xml

HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:4f9dfaa1-a823-4964-9583-8bbd69504595</id>
    <title xml:lang="en">MyContentItem</title>
    <summary xml:lang="en"></summary>
    <wcm:name>MyContentItem</wcm:name>
    <wcm:type>Content</wcm:type>
    ... some content elided ...
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:content xmlns="http://www.ibm.com/xmlns/wcm/8.0">
            <elements xmlns:atom="http://www.w3.org/2005/Atom">
                <element name="MyRichTextElement">
                    <title xml:lang="en">MyRichTextElement</title>
                    <type>RichTextComponent</type>
                    <data type="text/html"><![CDATA[<p dir="ltr">This is some rich text</p>]]&gt;</data>
                </element>
                <element name="MyDateComponent">
                    <title xml:lang="en">MyDateComponent</title>
                    <type>DateComponent</type>
                    <data type="application/vnd.ibm.wcm+xml">
                        <date type="DateTime">2014-08-06T15:00:00.000Z</date>
                    </data>
                </element>
                <element name="MyTextComponent">
                    <title xml:lang="en">MyTextComponent</title>
                    <type>TextComponent</type>
                    <data type="text/plain">This is some text</data>
                </element>
            </elements>
        </wcm:content>
    </content>
</entry>
```

Update multiple elements

An item that contains elements can have elements added, removed, or updated by issuing a PUT request, specifying the elements that should be present on the item. Any elements that don't exist on the item are added. Any elements that are not specified in the request will be removed. Any existing elements will be updated.

The elements of an item can be modified by issuing a PUT request to the following URIs:

Content item
/Content/content-id/

Site area
/SiteArea/site-area-id/

The default content of a content template
/ContentTemplate/template-id/Prototype/

The default content of a site area template

/SiteAreaTemplate/template-id/Prototype/

For example:

```
HTTP/1.1 GET /wps/mycontenthandler/wcmrest/Content/4f9dfaa1-a823-4964-9583-8bbd69504595
Accept: application/atom+xml

HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:4f9dfaa1-a823-4964-9583-8bbd69504595</id>
    <title xml:lang="en">MyContentItem</title>
    <summary xml:lang="en"></summary>
    <wcm:name>MyContentItem</wcm:name>
    <wcm:type>Content</wcm:type>
    ... some content elided ...
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:content xmlns="http://www.ibm.com/xmlns/wcm/8.0">
            <elements xmlns:atom="http://www.w3.org/2005/Atom">
                <element name="MyRichTextElement">
                    <title xml:lang="en">MyRichTextElement</title>
                    <type>RichTextComponent</type>
                    <data type="text/html"><![CDATA[<p dir="ltr">This is some rich text</p>]]&gt;</data>
                </element>
                <element name="MyDateComponent">
                    <title xml:lang="en">MyDateComponent</title>
                    <type>DateComponent</type>
                    <data type="application/vnd.ibm.wcm+xml">
                        <date type="DateTime">2014-08-06T15:00:00.000Z</date>
                    </data>
                </element>
                <element name="MyTextComponent">
                    <title xml:lang="en">MyTextComponent</title>
                    <type>TextComponent</type>
                    <data type="text/plain">This is some text</data>
                </element>
            </elements>
        </wcm:content>
    </content>
</entry>
```

This response is used to perform the following modifications:

1. Remove *MyTextComponent*.
2. Modify *MyRichTextElement*.
3. Add an element named *MyNewTextElement*.

```
HTTP/1.1 PUT /wps/mycontenthandler/wcmrest/Content/4f9dfaa1-a823-4964-9583-8bbd69504595
Content-Type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:4f9dfaa1-a823-4964-9583-8bbd69504595</id>
    <title xml:lang="en">MyContentItem</title>
    <summary xml:lang="en"></summary>
    <wcm:name>MyContentItem</wcm:name>
    <wcm:type>Content</wcm:type>
    ... some content elided ...
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:content xmlns="http://www.ibm.com/xmlns/wcm/8.0">
            <elements xmlns:atom="http://www.w3.org/2005/Atom">
                <element name="MyRichTextElement">
                    <title xml:lang="en">MyRichTextElement</title>
                    <type>RichTextComponent</type>
                    <data type="text/html"><![CDATA[<p dir="ltr">This is some rich text that has been modified</p>]]&gt;</data>
                </element>
                <element name="MyDateComponent">
                    <title xml:lang="en">MyDateComponent</title>
                    <type>DateComponent</type>
                    <data type="application/vnd.ibm.wcm+xml">
                        <date type="DateTime">2014-08-06T15:00:00.000Z</date>
                    </data>
                </element>
                <element name="MyNewTextElement">
                    <title xml:lang="en">MyNewTextElement</title>
                    <type>TextComponent</type>
                    <data type="text/plain">This is the newly created text element</data>
                </element>
            </elements>
        </wcm:content>
    </content>
</entry>


HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:4f9dfaa1-a823-4964-9583-8bbd69504595</id>
    <title xml:lang="en">MyContentItem</title>
    <summary xml:lang="en"></summary>
    <wcm:name>MyContentItem</wcm:name>
    <wcm:type>Content</wcm:type>
    ... some content elided ...
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:content xmlns="http://www.ibm.com/xmlns/wcm/8.0">
            <elements xmlns:atom="http://www.w3.org/2005/Atom">
                <element name="MyRichTextElement">
                    <title xml:lang="en">MyRichTextElement</title>
                    <type>RichTextComponent</type>
                    <data type="text/html"><![CDATA[<p dir="ltr">This is some rich text that has been modified</p>]]&gt;</data>
                </element>
                <element name="MyDateComponent">
                    <title xml:lang="en">MyDateComponent</title>
                    <type>DateComponent</type>
                    <data type="application/vnd.ibm.wcm+xml">
                        <date type="DateTime">2014-08-06T15:00:00.000Z</date>
                    </data>
                </element>
                <element name="MyNewTextElement">
                    <title xml:lang="en">MyNewTextElement</title>
                    <type>TextComponent</type>
                    <data type="text/plain">This is the newly created text element</data>
```

```
            </element>
          </elements>
        </wcm:content>
    </content>
</entry>
```

How to use REST with authoring templates

You can use the Web Content Manager REST service to create, read, update, and delete content templates and site area templates. You can also set default values for items that are created by using these authoring templates.

CF06 "How to use REST with content templates"
You can use the Web Content Manager REST service to create, read, update, and delete content templates. You can also set default values for items that are created by using these content templates.

CF06 "How to use REST with site area templates" on page 3341
You can use the Web Content Manager REST service to create, read, update, and delete site area templates. You can also set default values for items that are created by using these site area templates.

How to use REST with content templates:

You can use the Web Content Manager REST service to create, read, update, and delete content templates. You can also set default values for items that are created by using these content templates.

Create

A content template can be created by sending a POST request to the following URI with an Atom entry that represents the content template:

`/ContentTemplate`

For example:

```
POST /wps/mycontenthandler/wcmrest/ContentTemplate
Content-Type : application/atom+xml

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <title>SampleContentTemplateTitle</title>
    <wcm:name>SampleContentTemplateName</wcm:name>
    <summary xml:lang="en">SampleContentTemplateDescription</summary>
    <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/91d0b849-7e9b-4053-a267-d4b84be29062" label="Library"/>
</entry>

HTTP/1.0 201 Created
Content-type: application/atom+xml; type=entry
Content-location: /wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/e15bb37f-6eba-42f4-8470-e54431b69fb3

<?xml version="1.0" encoding="UTF-8"?><entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:d19db2ce-87dc-484f-937a-203139818fbd</id>
    <title xml:lang="en">SampleContentTemplateTitle</title>
    <summary xml:lang="en">SampleContentTemplateDescription</summary>
    <wcm:name>SampleContentTemplateName</wcm:name>
    <wcm:type>ContentTemplate</wcm:type>
    <updated>2015-03-19T04:02:33.655Z</updated>
    <wcm:created>2015-03-19T04:02:33.655Z</wcm:created>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,cn=users,dc=test</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/um/users/profiles/Z9eAeO9PA3SGCJPO0JM4633DEJM46GHC4MM07LHO4JM4C2BCCJOO64JCAMHH613</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,cn=users,dc=test</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/um/users/profiles/Z9eAeO9PA3SGCJPO0JM4633DEJM46GHC4MM07LHO4JM4C2BCCJOO64JCAMHH613</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,cn=users,dc=test</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/um/users/profiles/Z9eAeO9PA3SGCJPO0JM4633DEJM46GHC4MM07LHO4JM4C2BCCJOO64JCAMHH613</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
    <wcm:creator>
        <wcm:distinguishedName>uid=wpsadmin,cn=users,dc=test</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/um/users/profiles/Z9eAeO9PA3SGCJPO0JM4633DEJM46GHC4MM07LHO4JM4C2BCCJOO64JCAMHH613</uri>
        <name>wpsadmin</name>
    </wcm:creator>
    <link rel="self" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd" label="Delete"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/d19db2ce-87dc-484f-937a-203139818fbd/change-to-draft" label="Change To Draft"/>
    <link rel="create-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/d19db2ce-87dc-484f-937a-203139818fbd/create-draft" label="Create Draft"/>
    <link rel="access-control" href="/wps/mycontenthandler/!ut/ac/access:oid:Z6QReDe4BCI3I9CIPOAMM07N1P6MMG6O1DCMM47JPD2MM86GPC2JP47O9CG3J9C43" label="Access Control"/>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/Library/91d0b849-7e9b-4053-a267-d4b84be29062" label="Library"/>
    <link rel="versions" href="/wps/mycontenthandler/!ut/p/wcmrest/item/d19db2ce-87dc-484f-937a-203139818fbd/versions" label="Versions"/>
    <link rel="elements" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd/Prototype/elements" label="Elements"/>
    <link rel="prototype" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd/Prototype" label="Prototype"/>
    <link rel="prototype-properties" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd/Prototype/properties" type="application/vnd.ibm.wcm+xml" label="Prototype Properties"/>
    <link rel="new-content" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd/new-content" label="New Content"/>
    <category scheme="wcmrest:workflowState" term="PUBLISHED" label="Published" xml:lang="en"/>
    <category scheme="wcmrest:favorite" term="false" xml:lang="en"/>
</entry>
```

Update

A content template can be updated by sending a PUT request to the following URI with an Atom entry that includes the fields on the item that need to be changed.

`/ContentTemplate/item-uuid`

For example:

```
PUT /wps/mycontenthandler/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd
Content-Type: application/atom+xml

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <title>SampleContentTemplateTitleUpdated</title>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/Library/91d0b849-7e9b-4053-a267-d4b84be29062"/>
    <wcm:name>SampleContentTemplateNameUpdated</wcm:name>
    <summary xml:lang="en">SampleContentTemplateDescriptionUpdated</summary>
</entry>

HTTP/1.0 200 OK
Content-type: application/atom+xml; type=entry

<?xml version="1.0" encoding="UTF-8"?><entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:d19db2ce-87dc-484f-937a-203139818fbd</id>
    <title xml:lang="en">SampleContentTemplateTitleUpdated</title>
    <summary xml:lang="en">SampleContentTemplateDescriptionUpdated</summary>
    <wcm:name>SampleContentTemplateNameUpdated</wcm:name>
    <wcm:type>ContentTemplate</wcm:type>
    <updated>2015-03-19T04:06:49.462Z</updated>
    <wcm:created>2015-03-19T04:02:33.655Z</wcm:created>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,cn=users,dc=test</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/um/users/profiles/Z9eAeO9PA3SGCJPO0JM4633DEJM46GHC4MM07LHO4JM4C2BCCJOO64JCAMHH613</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
    <wcm:creator>
        <wcm:distinguishedName>uid=wpsadmin,cn=users,dc=test</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/um/users/profiles/Z9eAeO9PA3SGCJPO0JM4633DEJM46GHC4MM07LHO4JM4C2BCCJOO64JCAMHH613</uri>
        <name>wpsadmin</name>
    </wcm:creator>
    <link rel="self" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd" label="Delete"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/d19db2ce-87dc-484f-937a-203139818fbd/change-to-draft" label="Change To Draft"/>
    <link rel="create-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/d19db2ce-87dc-484f-937a-203139818fbd/create-draft" label="Create Draft"/>
    <link rel="access-control" href="/wps/mycontenthandler/!ut/ac/access:oid:Z6QReDe4BCI3I9C1POAMM07N1P6MMG6O1DCMM47JPD2MM86GPC2JP47O9CG3J9C43" label="Access Control"/>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/Library/91d0b849-7e9b-4053-a267-d4b84be29062" label="Library"/>
    <link rel="versions" href="/wps/mycontenthandler/!ut/p/wcmrest/item/d19db2ce-87dc-484f-937a-203139818fbd/versions" label="Versions"/>
    <link rel="elements" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd/Prototype/elements" label="Elements"/>
    <link rel="prototype" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd/Prototype" label="Prototype"/>
    <link rel="prototype-properties" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd/Prototype/properties" type="application/vnd.ibm.wcm+xml" label="Prototype Properties"/>
    <link rel="new-content" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd/new-content" label="New Content"/>
    <category scheme="wcmrest:workflowState" term="PUBLISHED" label="Published" xml:lang="en"/>
    <category scheme="wcmrest:favorite" term="false" xml:lang="en"/>
</entry>
```

Read

A content template can be read by sending a GET request to the following URI:

`/ContentTemplate/item-uuid`

For example:

```
GET /wps/mycontenthandler/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd

HTTP/1.0 200 OK
Content-type: application/atom+xml; type=entry

<?xml version="1.0" encoding="UTF-8"?><entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:d19db2ce-87dc-484f-937a-203139818fbd</id>
    <title xml:lang="en">SampleContentTemplateTitleUpdated</title>
    <summary xml:lang="en">SampleContentTemplateDescriptionUpdated</summary>
    <wcm:name>SampleContentTemplateName</wcm:name>
    <wcm:type>ContentTemplate</wcm:type>
    <updated>2015-03-19T04:06:49.462Z</updated>
    <wcm:created>2015-03-19T04:02:33.655Z</wcm:created>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,cn=users,dc=test</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/um/users/profiles/Z9eAeO9PA3SGCJPO0JM4633DEJM46GHC4MM07LHO4JM4C2BCCJOO64JCAMHH613</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
    <wcm:creator>
        <wcm:distinguishedName>uid=wpsadmin,cn=users,dc=test</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/um/users/profiles/Z9eAeO9PA3SGCJPO0JM4633DEJM46GHC4MM07LHO4JM4C2BCCJOO64JCAMHH613</uri>
        <name>wpsadmin</name>
    </wcm:creator>
    <link rel="self" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd" label="Delete"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/d19db2ce-87dc-484f-937a-203139818fbd/change-to-draft" label="Change To Draft"/>
    <link rel="create-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/d19db2ce-87dc-484f-937a-203139818fbd/create-draft" label="Create Draft"/>
    <link rel="access-control" href="/wps/mycontenthandler/!ut/ac/access:oid:Z6QReDe4BCI3I9C1POAMM07N1P6MMG6O1DCMM47JPD2MM86GPC2JP47O9CG3J9C43" label="Access Control"/>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/Library/91d0b849-7e9b-4053-a267-d4b84be29062" label="Library"/>
    <link rel="versions" href="/wps/mycontenthandler/!ut/p/wcmrest/item/d19db2ce-87dc-484f-937a-203139818fbd/versions" label="Versions"/>
    <link rel="elements" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd/Prototype/elements" label="Elements"/>
    <link rel="prototype" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd/Prototype" label="Prototype"/>
    <link rel="prototype-properties" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd/Prototype/properties" type="application/vnd.ibm.wcm+xml" label="Prototype Properties"/>
    <link rel="new-content" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd/new-content" label="New Content"/>
    <category scheme="wcmrest:workflowState" term="PUBLISHED" label="Published" xml:lang="en"/>
    <category scheme="wcmrest:favorite" term="false" xml:lang="en"/>
</entry>
```

Delete

A content template can be deleted by sending a DELETE request to the following URI:

`/ContentTemplate/item-uuid`

For example:

```
DELETE /wps/mycontenthandler/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd

Response:
Status Code :200
Status Message : OK
```

CF06 "How to set default content values for content templates by using REST"
You can update and read default content values for content items that are created by using a content template.

CF06 "How to set default properties for content templates by using REST" on page 3340
You can update and read the default properties of content items that are created by using a content template.

How to set default content values for content templates by using REST:

You can update and read default content values for content items that are created by using a content template.

These values include:

- Elements
- Default workflow
- Presentation template override

Update

You can update the default content values of a content template by sending a PUT request to the following URI:

`/ContentTemplate/item-uuid/Prototype`

For example:

```
PUT /wps/mycontenthandler/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd/Prototype
Content-Type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?><entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:8986d098-4b75-43c0-b912-37c905960dd1</id>
    <wcm:type>Content</wcm:type>
    <link rel="workflow" href="/wps/mycontenthandler/!ut/p/wcmrest/Workflow/27b4254a-3762-42e3-8099-997f394874d4" label="Workflow"/>
    <link rel="presentation-override" href="/wps/mycontenthandler/!ut/p/wcmrest/PresentationTemplate/832f2d12-829c-41eb-a808-63393a3f77ce" label="Presentation Override"/>
</entry>


HTTP/1.0 200 OK
Content-type: application/atom+xml; type=entry

<?xml version="1.0" encoding="UTF-8"?><entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:8986d098-4b75-43c0-b912-37c905960dd1</id>
    <title/>
    <summary/>
    <wcm:type>Content</wcm:type>
    <updated>2015-03-19T04:06:49.489Z</updated>
    <wcm:created>2015-03-19T04:02:33.670Z</wcm:created>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,cn=users,dc=test</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/um/users/profiles/Z9eAeO9PA3SGCJPO0JM4633DEJM46GHC4MM07LHO4JM4C2BCCJOO64JCAMHH613</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
    <wcm:creator>
        <wcm:distinguishedName>uid=wpsadmin,cn=users,dc=test</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/um/users/profiles/Z9eAeO9PA3SGCJPO0JM4633DEJM46GHC4MM07LHO4JM4C2BCCJOO64JCAMHH613</uri>
        <name>wpsadmin</name>
    </wcm:creator>
    <link rel="self" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd/Prototype" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd/Prototype" label="Edit"/>
    <link rel="workflow-stage" href="/wps/mycontenthandler/!ut/p/wcmrest/WorkflowStage/b241a6b7-49ea-4a33-ad26-4ca1cc927620" label="Workflow Stage"/>
    <link rel="workflow" href="/wps/mycontenthandler/!ut/p/wcmrest/Workflow/27b4254a-3762-42e3-8099-997f394874d4" label="Workflow"/>
    <link rel="edit-media" href="/wps/mycontenthandler/!ut/p/wcmrest/Content/8986d098-4b75-43c0-b912-37c905960dd1" type="application/vnd.ibm.wcm+xml" label="Edit Media"/>
    <link rel="presentation-override" href="/wps/mycontenthandler/!ut/p/wcmrest/PresentationTemplate/832f2d12-829c-41eb-a808-63393a3f77ce" label="Presentation Override"/>
    <category scheme="wcmrest:workflowState" term="DRAFT" label="Draft"/>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:content xmlns="http://www.ibm.com/xmlns/wcm/8.0">
```

```
            <elements xmlns:atom="http://www.w3.org/2005/Atom"/>
        </wcm:content>
    </content>
</entry>
```

Read

The default content values of a content template can be read by sending a GET request to the following URI:

`/ContentTemplate/item-uuid/Prototype`

For example:

```
GET /wps/mycontenthandler/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd/Prototype

HTTP/1.0 200 OK
Content-type: application/atom+xml; type=entry

<?xml version="1.0" encoding="UTF-8"?><entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:8986d098-4b75-43c0-b912-37c905960dd1</id>
    <title/>
    <summary/>
    <wcm:type>Content</wcm:type>
    <updated>2015-03-19T04:06:49.489Z</updated>
    <wcm:created>2015-03-19T04:02:33.670Z</wcm:created>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,cn=users,dc=test</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/um/users/profiles/Z9eAeO9PA3SGCJPO0JM4633DEJM46GHC4MM07LHO4JM4C2BCCJOO64JCAMHH613</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
    <wcm:creator>
        <wcm:distinguishedName>uid=wpsadmin,cn=users,dc=test</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/um/users/profiles/Z9eAeO9PA3SGCJPO0JM4633DEJM46GHC4MM07LHO4JM4C2BCCJOO64JCAMHH613</uri>
        <name>wpsadmin</name>
    </wcm:creator>
    <link rel="self" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd/Prototype" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd/Prototype" label="Edit"/>
    <link rel="edit-media" href="/wps/mycontenthandler/!ut/p/wcmrest/Content/8986d098-4b75-43c0-b912-37c905960dd1" type="application/vnd.ibm.wcm+xml" label="Edit Media"/>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:content xmlns="http://www.ibm.com/xmlns/wcm/8.0">
            <elements xmlns:atom="http://www.w3.org/2005/Atom"/>
        </wcm:content>
    </content>
</entry>
```

How to set default properties for content templates by using REST:

You can update and read the default properties of content items that are created by using a content template.

These properties include:

- Default presentation template
- Selected locations
- Create content under new site area
- Single site area option
- Enable or disable workflow

Update

You can update the default properties of a content template by sending a PUT request to the following URI:

`/ContentTemplate/item-uuid/Prototype/properties`

For example:

```
PUT /wps/mycontenthandler/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd/Prototype/properties
Content-Type: application/vnd.ibm.wcm+xml

<?xml version="1.0" encoding="UTF-8"?>
<content-properties xmlns="http://www.ibm.com/xmlns/wcm/8.0" xmlns:atom="http://www.w3.org/2005/Atom">
    <workflow-control>
        <option name="ENABLE_WORKFLOW" enabled="false"/>
        <option name="HIDE_WORKFLOW_SECTION" enabled="true"/>
    </workflow-control>
    <createNewParent>true</createNewParent>
    <placement>FIRST_CHILD</placement>
    <location-options allowedLocation="ALL_AVAILABLE" contentLink="NONE"/>
    <link rel="default-presentation" href="/wps/mycontenthandler/wcmrest/PresentationTemplate/37d77b82-c3fb-4ee8-ba88-3ce0a2c1443f" label="Default Presentation"/>
</content-properties>


HTTP/1.0 200 OK
Content-type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<content-properties xmlns="http://www.ibm.com/xmlns/wcm/8.0" xmlns:atom="http://www.w3.org/2005/Atom">
    <workflow-control>
        <option name="ENABLE_WORKFLOW" enabled="false"/>
```

```
        <option name="HIDE_WORKFLOW_SECTION" enabled="true"/>
    </workflow-control>
    <link rel="default-presentation" href="/wps/mycontenthandler/!ut/p/wcmrest/PresentationTemplate/37d77b82-c3fb-4ee8-ba88-3ce0a2c1443f" type="application/vnd.ibm.wcm+xml" label="Default Presentation"/>
    <createNewParent>true</createNewParent>
    <placement>FIRST_CHILD</placement>
    <location-options allowedLocation="ALL_AVAILABLE" contentLink="NONE"/>
</content-properties>
```

Read

The default properties of a content template can be read by sending a GET request to the following URI:

`/ContentTemplate/item-uuid/Prototype/properties`

For example:

```
GET /wps/mycontenthandler/wcmrest/ContentTemplate/d19db2ce-87dc-484f-937a-203139818fbd/Prototype/properties

HTTP/1.0 200 OK
Content-type: application/vnd.ibm.wcm+xml

<?xml version="1.0" encoding="UTF-8"?>
<content-properties xmlns="http://www.ibm.com/xmlns/wcm/8.0" xmlns:atom="http://www.w3.org/2005/Atom">
    <workflow-control>
        <option name="ENABLE_WORKFLOW" enabled="true"/>
        <option name="HIDE_WORKFLOW_SECTION" enabled="false"/>
    </workflow-control>
    <createNewParent>false</createNewParent>
    <placement>LAST_CHILD</placement>
    <location-options allowedLocation="ALL_AVAILABLE" contentLink="NONE"/>
</content-properties>
```

Format of properties values

Workflow options:

```
<workflow-control>
        <option name="ENABLE_WORKFLOW" enabled="false"/>
        <option name="HIDE_WORKFLOW_SECTION" enabled="true"/>
</workflow-control>
```

Default presentation template:

```
<link rel="default-presentation"
href="/wps/mycontenthandler/wcmrest/PresentationTemplate/<item-uuid>"
label="Default Presentation"/>
```

Create content under a new site area:

```
<createNewParent>true</createNewParent>
```

Valid values: true, false.

Default placement of new item:

```
<placement>FIRST_CHILD</placement>
```

Valid values: FIRST_CHILD, LAST_CHILD.

Selected locations:

When all available locations are allowed:

```
<location-options allowedLocation="ALL_AVAILABLE" contentLink="NONE"/>
```

When only selected locations are allowed:

```
<location-options allowedLocation="SELECTED" contentLink="NONE">
            <location href="/wps/mycontenthandler/!ut/p/wcmrest/SiteArea/<item-uuid>"/>
            <location href="/wps/mycontenthandler/!ut/p/wcmrest/SiteArea/<item-uuid>"/>
</location-options>
```

Valid values for `allowedLocation`: ALL_AVAILABLE, SELECTED.

Valid values for `contentLink`: SINGLE, MULTIPLE, NONE.

How to use REST with site area templates:

You can use the Web Content Manager REST service to create, read, update, and delete site area templates. You can also set default values for items that are created by using these site area templates.

Create

A site area template can be created by sending a POST request to the following URI with an Atom entry that represents the site area template:

`/SiteAreaTemplate`

For example:

```
POST /wps/mycontenthandler/wcmrest/SiteAreaTemplate
Content-Type : application/atom+xml

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <title>SampleSiteAreaTemplateTitle</title>
    <wcm:name>SampleSiteAreaTemplateName</wcm:name>
    <summary xml:lang="en">SampleSiteAreaTemplateDescription</summary>
    <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/91d0b849-7e9b-4053-a267-d4b84be29062" label="Library"/>
</entry>

HTTP/1.0 201 Created
Content-type: application/atom+xml; type=entry
Content-location:/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:7f1055db-29ac-43ea-88b8-c6b23019b6b9</id>
    <title xml:lang="en">SampleSiteAreaTemplateTitle</title>
    <summary xml:lang="en">SampleSiteAreaTemplateDescription</summary>
    <wcm:name>SampleSiteAreaTemplateName</wcm:name>
    <wcm:type>SiteAreaTemplate</wcm:type>
    <updated>2015-03-19T11:39:52.405Z</updated>
    <wcm:created>2015-03-19T11:39:52.405Z</wcm:created>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,cn=users,dc=test</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/um/users/profiles/Z9eAeO9PA3SGCJPO0JM4633DEJM46GHC4MM07LHO4JM4C2BCCJOO64JCAMHH613</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,cn=users,dc=test</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/um/users/profiles/Z9eAeO9PA3SGCJPO0JM4633DEJM46GHC4MM07LHO4JM4C2BCCJOO64JCAMHH613</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,cn=users,dc=test</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/um/users/profiles/Z9eAeO9PA3SGCJPO0JM4633DEJM46GHC4MM07LHO4JM4C2BCCJOO64JCAMHH613</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
    <wcm:creator>
        <wcm:distinguishedName>uid=wpsadmin,cn=users,dc=test</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/um/users/profiles/Z9eAeO9PA3SGCJPO0JM4633DEJM46GHC4MM07LHO4JM4C2BCCJOO64JCAMHH613</uri>
        <name>wpsadmin</name>
    </wcm:creator>
    <link rel="self" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9" label="Delete"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/7f1055db-29ac-43ea-88b8-c6b23019b6b9/change-to-draft" label="Change To Draft"/>
    <link rel="create-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/7f1055db-29ac-43ea-88b8-c6b23019b6b9/create-draft" label="Create Draft"/>
    <link rel="access-control" href="/wps/mycontenthandler/!ut/p/ac/access:oid:Z6QReDeNHP23OK6L1P4MM86P9O6MMG6J9P2MM07OHOGJMCCMHO4JPO6H9E46R8CP1" label="Access Control"/>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/Library/91d0b849-7e9b-4053-a267-d4b84be29062" label="Library"/>
    <link rel="versions" href="/wps/mycontenthandler/!ut/p/wcmrest/item/7f1055db-29ac-43ea-88b8-c6b23019b6b9/versions" label="Versions"/>
    <link rel="elements" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9/Prototype/elements" label="Elements"/>
    <link rel="prototype" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9/Prototype" label="Prototype"/>
    <link rel="prototype-properties" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9/Prototype/properties" type="application/vnd.ibm.wcm+xml" label="Prototype Properties"/>
    <link rel="new-sitearea" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9/new-sitearea" label="New Sitearea"/>
    <category scheme="wcmrest:workflowState" term="PUBLISHED" label="Published" xml:lang="en"/>
    <category scheme="wcmrest:favorite" term="false" xml:lang="en"/>
</entry>
```

Update

A site area template can be updated by sending a PUT request to the following URI with an Atom entry that includes the fields on the item that need to be changed.

`/SiteAreaTemplate/item-uuid`

For example:

```
PUT /wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9
Content-Type: application/atom+xml

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <title>SampleSiteAreaTemplateTitleUpdated</title>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/Library/91d0b849-7e9b-4053-a267-d4b84be29062"/>
    <wcm:name>SampleSiteAreaTemplateNameUpdated</wcm:name>
    <summary xml:lang="en">SampleSiteAreaTemplateDescriptionUpdated</summary>
</entry>

HTTP/1.0 200 OK
Content-type: application/atom+xml; type=entry

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:7f1055db-29ac-43ea-88b8-c6b23019b6b9</id>
    <title xml:lang="en">SampleSiteAreaTemplateTitleUpdated</title>
    <summary xml:lang="en">SampleSiteAreaTemplateDescriptionUpdated</summary>
    <wcm:name>SampleSiteAreaTemplateNameUpdated</wcm:name>
    <wcm:type>SiteAreaTemplate</wcm:type>
    <updated>2015-03-19T11:46:42.869Z</updated>
    <wcm:created>2015-03-19T11:39:52.405Z</wcm:created>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,cn=users,dc=test</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/um/users/profiles/Z9eAeO9PA3SGCJPO0JM4633DEJM46GHC4MM07LHO4JM4C2BCCJOO64JCAMHH613</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
    <wcm:creator>
        <wcm:distinguishedName>uid=wpsadmin,cn=users,dc=test</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/um/users/profiles/Z9eAeO9PA3SGCJPO0JM4633DEJM46GHC4MM07LHO4JM4C2BCCJOO64JCAMHH613</uri>
```

```
        <name>wpsadmin</name>
    </wcm:creator>
    <link rel="self" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9" label="Delete"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/7f1055db-29ac-43ea-88b8-c6b23019b6b9/change-to-draft" label="Change To Draft"/>
    <link rel="create-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/7f1055db-29ac-43ea-88b8-c6b23019b6b9/create-draft" label="Create Draft"/>
    <link rel="access-control" href="/wps/mycontenthandler/!ut/p/ac/access:oid:Z6QReDeNHP23OK6L1P4MM86P9O6MMG6J9P2MM07OHOGJMCCMHO4JP06H9E46R8CP1" label="Access Control"/>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/Library/91d0b849-7e9b-4053-a267-d4b84be29062" label="Library"/>
    <link rel="versions" href="/wps/mycontenthandler/!ut/p/wcmrest/item/7f1055db-29ac-43ea-88b8-c6b23019b6b9/versions" label="Versions"/>
    <link rel="elements" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9/Prototype/elements" label="Elements"/>
    <link rel="prototype" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9/Prototype" label="Prototype"/>
    <link rel="prototype-properties" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9/Prototype/properties" type="application/vnd.ibm.wcm+xml" label="Prototype Properties"/>
    <link rel="new-sitearea" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9/new-sitearea" label="New Sitearea"/>
    <category scheme="wcmrest:workflowState" term="PUBLISHED" label="Published" xml:lang="en"/>
    <category scheme="wcmrest:favorite" term="false" xml:lang="en"/>
</entry>
```

Read

A site area template can be read by sending a GET request to the following URI:

`/SiteAreaTemplate/item-uuid`

For example:

```
GET /wps/mycontenthandler/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9

HTTP/1.0 200 OK
Content-type: application/atom+xml; type=entry

<?xml version="1.0" encoding="UTF-8"?><entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:7f1055db-29ac-43ea-88b8-c6b23019b6b9</id>
    <title xml:lang="en">SampleSiteAreaTemplateTitleUpdated</title>
    <summary xml:lang="en">SampleSiteAreaTemplateDescriptionUpdated</summary>
    <wcm:name>SampleSiteAreaTemplateUpdated</wcm:name>
    <wcm:type>SiteAreaTemplate</wcm:type>
    <updated>2015-03-19T11:46:42.869Z</updated>
    <wcm:created>2015-03-19T11:39:52.405Z</wcm:created>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,cn=users,dc=test</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/um/users/profiles/Z9eAeO9PA3SGCJPO0JM4633DEJM46GHC4MM07LHO4JM4C2BCCJOO64JCAMHH613</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
    <wcm:creator>
        <wcm:distinguishedName>uid=wpsadmin,cn=users,dc=test</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/um/users/profiles/Z9eAeO9PA3SGCJPO0JM4633DEJM46GHC4MM07LHO4JM4C2BCCJOO64JCAMHH613</uri>
        <name>wpsadmin</name>
    </wcm:creator>
    <link rel="self" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9" label="Delete"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/7f1055db-29ac-43ea-88b8-c6b23019b6b9/change-to-draft" label="Change To Draft"/>
    <link rel="create-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/7f1055db-29ac-43ea-88b8-c6b23019b6b9/create-draft" label="Create Draft"/>
    <link rel="access-control" href="/wps/mycontenthandler/!ut/p/ac/access:oid:Z6QReDeNHP23OK6L1P4MM86P9O6MMG6J9P2MM07OHOGJMCCMHO4JP06H9E46R8CP1" label="Access Control"/>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/Library/91d0b849-7e9b-4053-a267-d4b84be29062" label="Library"/>
    <link rel="versions" href="/wps/mycontenthandler/!ut/p/wcmrest/item/7f1055db-29ac-43ea-88b8-c6b23019b6b9/versions" label="Versions"/>
    <link rel="elements" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9/Prototype/elements" label="Elements"/>
    <link rel="prototype" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9/Prototype" label="Prototype"/>
    <link rel="prototype-properties" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9/Prototype/properties" type="application/vnd.ibm.wcm+xml" label="Prototype Properties"/>
    <link rel="new-sitearea" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9/new-sitearea" label="New Sitearea"/>
    <category scheme="wcmrest:workflowState" term="PUBLISHED" label="Published" xml:lang="en"/>
    <category scheme="wcmrest:favorite" term="false" xml:lang="en"/>
</entry>
```

Delete

A site area template can be deleted by sending a DELETE request to the following URI:

`/SiteAreaTemplate/item-uuid`

For example:

```
DELETE /wps/mycontenthandler/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9

Response:
Status Code :200
Status Message : OK
```

CF06 “How to set default site area values for site area templates by using REST”
You can update and read default site area values for site areas that are created by using a site area template.

CF06 “How to set default properties for site area templates by using REST” on page 3345
You can update and read the default properties of site areas that are created by using a site area template.

How to set default site area values for site area templates by using REST:

You can update and read default site area values for site areas that are created by using a site area template.

These values include:

- Elements
- Setting workflow
- Child template mappings
- Rendering behavior
- Presentation override
- Child default content

Update

You can update the default site area values a of site area template by sending a PUT request to the following URI:

`/SiteAreaTemplate/item-uuid/Prototype`

For example:

```
PUT /wps/mycontenthandler/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9/Prototype
Content-Type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <link rel="workflow" href="/wps/mycontenthandler/!ut/wcmrest/Workflow/27b4254a-3762-42e3-8099-997f394874d4" label="Workflow"/>
    <category scheme="wcmrest:renderingBehaviour" term="RENDER_DEFAULT_CONTENT_AS_CHILD" label="Render the default content as a child of this site area"/>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:siteArea xmlns="http://www.ibm.com/xmlns/wcm/8.0">
            <elements xmlns:atom="http://www.w3.org/2005/Atom"/>
            <templateMap>
                <templateMapping authoringTemplate="/wps/mycontenthandler/!ut/wcmrest/item/db6364ac-e29d-467a-9396-12ab82df6a31" presentationTemplate="/wps/mycontenthandler/!ut/wcmrest/PresentationTemplate/832f2d12-829c-41eb-a808-6
                <templateMapping authoringTemplate="/wps/mycontenthandler/!ut/wcmrest/item/e15bb37f-6eba-42f4-8470-e54431b69fb3" presentationTemplate="/wps/mycontenthandler/!ut/wcmrest/PresentationTemplate/37d77b82-c3fb-4ee8-ba88-3
            </templateMap>
        </wcm:siteArea>
    </content>
</entry>


HTTP/1.0 200 OK
Content-type: application/atom+xml; type=entry

<?xml version="1.0" encoding="UTF-8"?><entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:a2fd2e1d-d8da-44de-9ce3-17c8ab614208</id>
    <title/>
    <summary/>
    <wcm:name></wcm:name>
    <wcm:type>SiteArea</wcm:type>
    <updated>2015-03-19T12:09:02.600Z</updated>
    <wcm:created>2015-03-19T11:39:52.507Z</wcm:created>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,cn=users,dc=test</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/um/users/profiles/Z9eAeO9PA3SGCJPO0JM4633DEJM46GHC4MM07LHO4JM4C2BCCJOO64JCAMHH613</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
    <link rel="self" href="/wps/mycontenthandler/!ut/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9/Prototype" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/!ut/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9/Prototype" label="Edit"/>
    <link rel="workflow-stage" href="/wps/mycontenthandler/!ut/wcmrest/WorkflowStage/b241a6b7-49ea-4a33-ad26-4ca1cc927620" label="Workflow Stage"/>
    <link rel="workflow" href="/wps/mycontenthandler/!ut/wcmrest/Workflow/27b4254a-3762-42e3-8099-997f394874d4" label="Workflow"/>
    <link rel="edit-media" href="/wps/mycontenthandler/!ut/wcmrest/SiteArea/a2fd2e1d-d8da-44de-9ce3-17c8ab614208" type="application/vnd.ibm.wcm+xml" label="Edit Media"/>
    <category scheme="wcmrest:workflowState" term="DRAFT" label="Draft"/>
    <category scheme="wcmrest:renderingBehaviour" term="RENDER_DEFAULT_CONTENT_AS_CHILD" label="Render the default content as a child of this site area"/>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:siteArea xmlns="http://www.ibm.com/xmlns/wcm/8.0">
            <elements xmlns:atom="http://www.w3.org/2005/Atom"/>
            <templateMap>
                <templateMapping authoringTemplate="/wps/mycontenthandler/!ut/wcmrest/ContentTemplate/db6364ac-e29d-467a-9396-12ab82df6a31" presentationTemplate="/wps/mycontenthandler/!ut/wcmrest/PresentationTemplate/832f2d12-829c-
                <templateMapping authoringTemplate="/wps/mycontenthandler/!ut/wcmrest/ContentTemplate/e15bb37f-6eba-42f4-8470-e54431b69fb3" presentationTemplate="/wps/mycontenthandler/!ut/wcmrest/PresentationTemplate/37d77b82-c3fb-
            </templateMap>
        </wcm:siteArea>
    </content>
</entry>
```

Read

The default site area values of a site area template can be read by sending a GET request to the following URI:

`/SiteAreaTemplate/item-uuid/Prototype`

For example:

```
GET /wps/mycontenthandler/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9/Prototype

HTTP/1.0 200 OK
Content-type: application/atom+xml; type=entry

<?xml version="1.0" encoding="UTF-8"?><entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:a2fd2e1d-d8da-44de-9ce3-17c8ab614208</id>
    <title/>
    <summary/>
    <wcm:type>SiteArea</wcm:type>
    <updated>2015-03-19T11:46:42.999Z</updated>
    <wcm:created>2015-03-19T11:39:52.507Z</wcm:created>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,cn=users,dc=test</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/um/users/profiles/Z9eAeO9PA3SGCJPO0JM4633DEJM46GHC4MM07LHO4JM4C2BCCJOO64JCAMHH613</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
```

```
    <link rel="self" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9/Prototype" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9/Prototype" label="Edit"/>
    <link rel="edit-media" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteArea/a2fd2e1d-d8da-44de-9ce3-17c8ab614208" type="application/vnd.ibm.wcm+xml" label="Edit Media"/>
    <category scheme="wcmrest:renderingBehaviour" term="RENDER_DEFAULT_CONTENT_AS_CHILD" label="Render the default content as a child of this site area"/>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:siteArea xmlns="http://www.ibm.com/xmlns/wcm/8.0">
            <elements xmlns:atom="http://www.w3.org/2005/Atom"/>
        </wcm:siteArea>
    </content>
</entry>
```

Dependencies of Presentation Override and Default Content on Rendering Behavior

The availability of the **Presentation Override** and **Default Content** fields depend on the value of rendering behavior field.

Table 487. Presentation Override and Default Content on Rendering Behavior

Rendering Behavior	Presentation Override	Default Content
RENDER_DEFAULT_CONTENT_AS_CHILD	No.	Yes.
REDIRECT_TO_DEFAULT_CONTENT	No.	Yes.
RENDER_SITE_AREA_DIRECTLY	Yes.	No.
RENDER_FIRST_CHILD_CONTENT	No.	No.
RENDER_FIRST_CHILD	No.	No.

How to set default properties for site area templates by using REST:

You can update and read the default properties of site areas that are created by using a site area template.

These properties include:

- Default presentation template
- Enable or disable workflow

Update

You can update the default properties of a site area template by sending a PUT request to the following URI:

`/SiteAreaTemplate/item-uuid/Prototype/properties`

For example:

```
PUT /wps/mycontenthandler/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9/Prototype/properties
Content-Type: application/vnd.ibm.wcm+xml

<?xml version="1.0" encoding="UTF-8"?>
<sitearea-properties xmlns="http://www.ibm.com/xmlns/wcm/8.0" xmlns:atom="http://www.w3.org/2005/Atom">
    <workflow-control>
        <option name="ENABLE_WORKFLOW" enabled="true"/>
        <option name="HIDE_WORKFLOW_SECTION" enabled="true"/>
    </workflow-control>
    <link rel="default-presentation" href="/wps/mycontenthandler/wcmrest/PresentationTemplate/37d77b82-c3fb-4ee8-ba88-3ce0a2c1443f"  label="Default Presentation"/>
</sitearea-properties>


HTTP/1.0 200 OK
Content-type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<sitearea-properties xmlns="http://www.ibm.com/xmlns/wcm/8.0">
    <workflow-control>
        <option name="ENABLE_WORKFLOW" enabled="true"/>
        <option name="HIDE_WORKFLOW_SECTION" enabled="true"/>
    </workflow-control>
    <link rel="default-presentation" href="/wps/mycontenthandler/!ut/p/wcmrest/PresentationTemplate/37d77b82-c3fb-4ee8-ba88-3ce0a2c1443f" type="application/vnd.ibm.wcm+xml"  label="Default Presentation"/>
</sitearea-properties>
```

Read

The default properties of a site area template can be read by sending a GET request to the following URI:

`/SiteAreaTemplate/item-uuid/Prototype/properties`

For example:

```
GET /wps/mycontenthandler/wcmrest/SiteAreaTemplate/7f1055db-29ac-43ea-88b8-c6b23019b6b9/Prototype/properties

HTTP/1.0 200 OK
Content-type: application/vnd.ibm.wcm+xml

<?xml version="1.0" encoding="UTF-8"?>
<sitearea-properties xmlns="http://www.ibm.com/xmlns/wcm/8.0">
    <workflow-control>
        <option name="ENABLE_WORKFLOW" enabled="false"/>
        <option name="HIDE_WORKFLOW_SECTION" enabled="false"/>
    </workflow-control>
</sitearea-properties>
```

How to REST with presentation templates

You can use the Web Content Manager REST service to create, read, update, and delete presentation templates.

Create

A presentation template can be created by sending a POST request to the following URI with an Atom entry that represents the presentation template:

`/PresentationTemplate`

For example:

```
POST /wps/mycontenthandler/wcmrest/PresentationTemplate HTTP/1.0 POST
Content-Type : application/atom+xml

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <title>SamplePresentationTemplateTitle</title>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/item/c98d11e1-7f2a-480e-9aac-40eb1949cbda"/>
    <wcm:name>SamplePresentationTemplateTitleName</wcm:name>
    <wcm:description>SamplePresentationTemplateDescription</wcm:description>
</entry>


HTTP/1.0 201 Created
Content-type: application/atom+xml; type=entry
Content-location: /wps/mycontenthandler/!ut/p/wcmrest/PresentationTemplate/02da6e9d-20ca-4c54-ae4a-f1114fa8e948

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <id>02da6e9d-20ca-4c54-ae4a-f1114fa8e948</id>
    <title>SamplePresentationTemplateTitle</title>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/PresentationTemplate/02da6e9d-20ca-4c54-ae4a-f1114fa8e948"/>
    <link rel="edit-media" type="text/html" href="/wps/mycontenthandler/!ut/p/wcmrest/PresentationTemplate/02da6e9d-20ca-4c54-ae4a-f1114fa8e948"/>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/item/c98d11e1-7f2a-480e-9aac-40eb1949cbda"/>
    <link rel="create-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/02da6e9d-20ca-4c54-ae4a-f1114fa8e948/create-draft"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/02da6e9d-20ca-4c54-ae4a-f1114fa8e948/change-to-draft"/>
    <link rel="versions" href="/wps/mycontenthandler/!ut/p/wcmrest/item/02da6e9d-20ca-4c54-ae4a-f1114fa8e948/versions"/>
    <link rel="add-attachment" href="/wps/mycontenthandler/!ut/p/wcmrest/PresentationTemplate/02da6e9d-20ca-4c54-ae4a-f1114fa8e948/attachments"/>
    <updated>2011-05-30T04:46:34.811Z</updated>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5Ul75Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5Ul75Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:name>SamplePresentationTemplateTitleName</wcm:name>
    <wcm:description>SamplePresentationTemplateDescription</wcm:description>
    <wcm:type>PresentationTemplate</wcm:type>
    <wcm:state>PUBLISHED</wcm:state>
</entry>
```

Update

A presentation template can be updated by sending a PUT request to the following URI with an Atom entry that includes the fields on the item that need to be changed.

`/PresentationTemplate/<itemuuid>`

For example:

```
PUT /wps/mycontenthandler/wcmrest/PresentationTemplate/02da6e9d-20ca-4c54-ae4a-f1114fa8e948 HTTP/1.0
Content-Type: application/atom+xml

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <title>SamplePresentationTemplateTitleUpdated</title>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/item/c98d11e1-7f2a-480e-9aac-40eb1949cbda"/>
    <wcm:name>SamplePresentationTemplateTitleNameUpdated</wcm:name>
    <wcm:description>SamplePresentationTemplateDescriptionUpdated</wcm:description>
</entry>

HTTP/1.0 200 OK
Content-type: application/atom+xml; type=entry

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <id>02da6e9d-20ca-4c54-ae4a-f1114fa8e948</id>
    <title>SamplePresentationTemplateTitleUpdated</title>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/PresentationTemplate/02da6e9d-20ca-4c54-ae4a-f1114fa8e948"/>
    <link rel="edit-media" type="text/html" href="/wps/mycontenthandler/!ut/p/wcmrest/PresentationTemplate/02da6e9d-20ca-4c54-ae4a-f1114fa8e948"/>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/item/c98d11e1-7f2a-480e-9aac-40eb1949cbda"/>
    <link rel="create-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/02da6e9d-20ca-4c54-ae4a-f1114fa8e948/create-draft"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/02da6e9d-20ca-4c54-ae4a-f1114fa8e948/change-to-draft"/>
```

```
    <link rel="versions" href="/wps/mycontenthandler/!ut/p/wcmrest/item/02da6e9d-20ca-4c54-ae4a-f1114fa8e948/versions"/>
    <link rel="add-attachment" href="/wps/mycontenthandler/!ut/p/wcmrest/PresentationTemplate/02da6e9d-20ca-4c54-ae4a-f1114fa8e948/attachments"/>
    <updated>2011-05-30T04:51:39.260Z</updated>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5Ul75Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5Ul75Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:name>SamplePresentationTemplateTitleNameUpdated</wcm:name>
    <wcm:description>SamplePresentationTemplateDescriptionUpdated</wcm:description>
    <wcm:type>PresentationTemplate</wcm:type>
    <wcm:state>PUBLISHED</wcm:state>
</entry>
```

Read

A presentation template can be read by sending a GET request to the following URI:

`/PresentationTemplate/<itemuuid>`

For example:

```
GET /wps/mycontenthandler/wcmrest/PresentationTemplate/02da6e9d-20ca-4c54-ae4a-f1114fa8e948 HTTP/1.0


HTTP/1.0 200 OK
Content-type: application/atom+xml; type=entry

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <id>02da6e9d-20ca-4c54-ae4a-f1114fa8e948</id>
    <title>SamplePresentationTemplateTitleUpdated</title>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/PresentationTemplate/02da6e9d-20ca-4c54-ae4a-f1114fa8e948"/>
    <link rel="edit-media" type="text/html" href="/wps/mycontenthandler/!ut/p/wcmrest/PresentationTemplate/02da6e9d-20ca-4c54-ae4a-f1114fa8e948"/>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/item/c98d11e1-7f2a-480e-9aac-40eb1949cbda"/>
    <link rel="create-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/02da6e9d-20ca-4c54-ae4a-f1114fa8e948/create-draft"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/02da6e9d-20ca-4c54-ae4a-f1114fa8e948/change-to-draft"/>
    <link rel="versions" href="/wps/mycontenthandler/!ut/p/wcmrest/item/02da6e9d-20ca-4c54-ae4a-f1114fa8e948/versions"/>
    <link rel="add-attachment" href="/wps/mycontenthandler/!ut/p/wcmrest/PresentationTemplate/02da6e9d-20ca-4c54-ae4a-f1114fa8e948/attachments"/>
    <updated>2011-05-30T04:51:39.260Z</updated>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5Ul75Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <email></email>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5Ul75Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <email></email>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:name>SamplePresentationTemplateTitleNameUpdated</wcm:name>
    <wcm:description>SamplePresentationTemplateDescriptionUpdated</wcm:description>
    <wcm:type>PresentationTemplate</wcm:type>
    <wcm:state>PUBLISHED</wcm:state>
</entry>
```

Delete

A presentation template can be deleted by sending a DELETE request to the following URI:

`/PresentationTemplate/<itemuuid>`

For example:

DELETE

```
HTTP/1.1 DELETE
http://host:port/wps/mycontenthandler/wcmrest/PresentationTemplate/<itemuuid>
```

Response:

```
Status Code :200
Status Message : OK
```

Specifying raw data

The content of a presentation template is accessed from the media resource that is specified in the HREF attribute of the edit-media link. The link also contains a TYPE attribute that contains the accepted media type for presentation templates, which is text/html. For example:

```
<link rel="edit-media" type="text/html"
href="/wps/mycontenthandler/!ut/p/wcmrest/PresentationTemplate/02da6e9d-20ca-4c54-ae4a-f1114fa8e948"/>
```

To update content of a presentation template PUT content of type text/html type to the edit-media URL. For example:

```
PUT /wps/mycontenthandler/!ut/p/wcmrest/PresentationTemplate/02da6e9d-20ca-4c54-ae4a-f1114fa8e948 HTTP/1.0
Content-type: text/html

<html>
    <body>
        [Component name="story"]
    </body>
</html>

HTTP/1.0 200 OK
```

To retrieve content from a library component GET content from the edit-media URL. For example:

```
GET  /wps/mycontenthandler/!ut/p/wcmrest/PresentationTemplate/02da6e9d-20ca-4c54-ae4a-f1114fa8e948 HTTP/1.0
Accept: text/html


HTTP/1.0 200 OK
Content-type: text/html

<html>
    <body>
        [Component name="story"]
    </body>
</html>
```

How to use REST with content items

You can use the Web Content Manager REST service to create, read, update, and delete content items.

Create

A content item can be created by sending a POST request to the following URI with an Atom entry used to represent the content item:

`/Content`

- A library or parent link relation must be used to define the location of the hierarchical item that is being created.
- An authoring template must be specified to set what authoring template to use when the item is created.

For example:

POST

```
HTTP/1.1 POST
http://host:port/wps/mycontenthandler/wcmrest/Content/
Content-Type: application/atom+xml
  <atom:entry xmlns:atom="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
   <wcm:name>Content Name</wcm:name>
   <atom:title>Content Title</atom:title>
   <atom:link atom:rel="parent" atom:href="/wps/mycontenthandler/wcmrest/item/49f4ed95-a99f-434c-a415-77c341fa4893"/>
   <atom:link atom:rel="workflow" atom:href="/wps/mycontenthandler/wcmrest/item/abae799b-4cca-47ae-aad8-b3d8204deefb"/>
   <atom:link atom:rel="content-template" atom:href="/wps/mycontenthandler/wcmrest/item/588127d0-a4f8-44b5-87a4-5fe3f7bd3da7"/>
  </atom:entry>
```

Response:

```
201 Created
```

Create from a skeleton

A "skeleton" representation of a content item that is created from a content template can be obtained to aid in the creation of content items. This can be obtained by using a GET request to the following URI. When the skeleton is obtained and completed a POST request can be made by using this data to create the item.

`/ContentTemplate/`*`template-uuid`*`/new-content`

For example:

```
HTTP/1.1 GET http://host:port/wps/mycontenthandler/wcmrest/ContentTemplate/b7b8b3fb-8fa1-4eb3-915e-ce7514f7067f/new-content

Response
200 OK
```

```
<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm">
    <id>wcmrest:6bab48c6-1f24-454e-9bab-ae1be4cf3a0a</id>
    <title lang="en"></title>
    <summary lang="en"></summary>
    <wcm:name></wcm:name>
    <wcm:type>Content</wcm:type>
    <updated>2012-01-31T03:28:08.118Z</updated>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!7K1PhYjxBw0jzCDqHCwg2w/um/users/profiles/Z9eAeHPCAJG963RD2MMG6P9O6MMG66BD6MM47IHP4MMS6M1DAJQ4C1BCAMID653</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!7K1PhYjxBw0jzCDqHCwg2w/um/users/profiles/Z9eAeHPCAJG963RD2MMG6P9O6MMG66BD6MM47IHP4MMS6M1DAJQ4C1BCAMID653</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <link label="Library" rel="library" href="/wps/mycontenthandler/!ut/p/digest!PQo5Yhy68oeppWcEz2sddA/wcmrest/item/a423287f-b0ce-4ee3-9c95-aa0939382228" lang="en"/>
    <link label="Content Template" rel="content-template" href="/wps/mycontenthandler/!ut/p/digest!PQo5Yhy68oeppWcEz2sddA/wcmrest/Content/b7b8b3fb-8fa1-4eb3-915e-ce7514f7067f" lang="en"/>
    <content type="application/vnd.ibm.wcm+xml">
        <content xmlns="http://www.ibm.com/xmlns/wcm">
            <elements>
                <element name="Body">
                    <title lang="en">Body</title>
                    <type>RichTextComponent</type>
                    <data type="text/html"></data>
                </element>
            </elements>
        </content>
    </content>
</entry>
```

Update

A content item can be updated by sending a PUT request to the following URI with an Atom entry that specifies the fields on the item that need to be changed.

`/Content/`*`item-uuid`*

For example:

PUT

```
HTTP/1.1 PUT
http://host:port/wps/mycontenthandler/wcmrest/Content/abae799b-4cca-47ae-aad8-b3d8204deefb
Content-Type: application/atom+xml
<atom:entry xmlns:atom="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <wcm:name>Updated Content Name</wcm:name>
    <atom:title>Updated Content Title</atom:title>
</atom:entry>
```

Response:

200 OK

Read

A content item can be read by sending a GET request to the following URI:

`/Content/`*`item-uuid`*

For example:

GET

```
HTTP/1.1 GET
http://host:port/wps/mycontenthandler/wcmrest/Content/fa2bfd32-7b2f-4394-a5ab-2e150c5ed8aa

<atom:entry xmlns:atom="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <atom:id>fa2bfd32-7b2f-4394-a5ab-2e150c5ed8aa</atom:id>
    <wcm:name>content name145805586</wcm:name>
    <atom:title>content title145805586</atom:title>
    <wcm:type>Content</wcm:type>
    <atom:updated>2011-04-27T04:06:32.643Z</atom:updated>
    <atom:link atom:rel="edit" atom:href="/wps/mycontenthandler/wcmrest/Content/fa2bfd32-7b2f-4394-a5ab-2e150c5ed8aa"/>
    <atom:link atom:rel="library" atom:href="/wps/mycontenthandler/wcmrest/item/957a67f2-9d70-469f-9d43-f63f78508e48"/>
    <atom:link atom:rel="parent" atom:href="/wps/mycontenthandler/wcmrest/item/49f4ed95-a99f-434c-a415-77c341fa4893"/>
    <atom:link atom:rel="workflow-stage" atom:href="/wps/mycontenthandler/wcmrest/item/f659d3af-7d45-4fc0-ad37-86e407caf2b6"/>
    <atom:link atom:rel="workflow" atom:href="/wps/mycontenthandler/wcmrest/item/abae799b-4cca-47ae-aad8-b3d8204deefb"/>
    <atom:link atom:rel="versions" atom:href="/wps/mycontenthandler/wcmrest/item/fa2bfd32-7b2f-4394-a5ab-2e150c5ed8aa/versions"/>
    <atom:link atom:rel="content-template" atom:href="/wps/mycontenthandler/wcmrest/item/588127d0-a4f8-44b5-87a4-5fe3f7bd3da7"/>
    <atom:link atom:rel="elements" atom:href="/wps/mycontenthandler/wcmrest/Content/fa2bfd32-7b2f-4394-a5ab-2e150c5ed8aa/elements"/>
</atom:entry>
```

Response:

200 OK

Delete

A content item can be deleted by sending a DELETE request to the following URI:
`/Content/item-uuid`

For example:

DELETE

```
HTTP/1.1 DELETE
http://host:port/wps/mycontenthandler/wcmrest/Content/fa2bfd32-7b2f-4394-a5ab-2e150c5ed8aa/
```

Response:

```
200 OK
```

How to use REST with site areas

You can use the Web Content Manager REST service to create, read, update, and delete site areas.

Create

A site area can be created by sending a POST request to the following URI with an Atom entry that represents the site area:
`/SiteArea`

- A library or parent link relation must be specified. This tells the REST service the location of the hierarchical item being created.
- An authoring template must be specified. This tells the REST service what authoring template to use when creating the item.
- Template mappings can also be specified.

For example:

```
POST /wps/mycontenthandler/wcmrest/SiteArea HTTP/1.0
Content-type : application/atom+xml

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <title>SampleSiteAreaTitle</title>
    <link rel="parent" href="/wps/mycontenthandler/!ut/p/wcmrest/item/ae6a3632-a1b5-456a-866e-e9baab84fe29"/>
    <wcm:name>SampleSiteAreaName</wcm:name>
    <wcm:description>SampleSiteAreaDescription</wcm:description>
</entry>


HTTP/1.0 201 Created
Content-type : application/atom+xml; type=entry
Content-location: /wps/mycontenthandler/!ut/p/wcmrest/SiteArea/18001a8c-2117-45d2-be1c-baea28a41769

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <id>18001a8c-2117-45d2-be1c-baea28a41769</id>
    <title>SampleSiteAreaTitle</title>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteArea/18001a8c-2117-45d2-be1c-baea28a41769"/>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/item/54a68ca2-c550-4385-966f-b0b612147547"/>
    <link rel="parent" href="/wps/mycontenthandler/!ut/p/wcmrest/item/ae6a3632-a1b5-456a-866e-e9baab84fe29"/>
    <link rel="create-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/18001a8c-2117-45d2-be1c-baea28a41769/create-draft"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/18001a8c-2117-45d2-be1c-baea28a41769/change-to-draft"/>
    <link rel="versions" href="/wps/mycontenthandler/!ut/p/wcmrest/item/18001a8c-2117-45d2-be1c-baea28a41769/versions"/>
    <link rel="default-content" href="/wps/mycontenthandler/!ut/p/wcmrest/Content/null"/>
    <link rel="elements" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteArea/18001a8c-2117-45d2-be1c-baea28a41769/elements"/>
    <updated>2011-05-30T06:01:56.330Z</updated>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5Ul75Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5Ul75Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:name>SampleSiteAreaName</wcm:name>
    <wcm:description>SampleSiteAreaDescription</wcm:description>
    <wcm:type>SiteArea</wcm:type>
    <wcm:state>PUBLISHED</wcm:state>
  <content type="application/vnd.ibm.wcm+xml">
        <wcm:siteArea xmlns="http://www.ibm.com/xmlns/wcm">
            <elements xmlns:atom="http://www.w3.org/2005/Atom"/>
            <templateMap>
                <templateMapping authoringTemplate="/wps/mycontenthandler/!ut/p/digest!One_9iWYnKogRRVx7BtBTA/wcmrest/ContentTemplate/d1dab663-4488-45e4-b63e-2f9339d50b57" presentationTemplate="/wps/mycontenthandler/!ut/p/digest!Or
            </templateMap>
        </wcm:siteArea>
</entry>
```

Create from a skeleton

A "skeleton" representation of a site area that is created from a site area template can be obtained to aid in the creation of site areas. This can be obtained by sending a GET request to the following URI. When the skeleton is obtained and completed a POST request can be made by using this data to create the item.

`/SiteAreaTemplate/`*`template-uuid`*`/new-sitearea`

For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm">
    <id>wcmrest:38188c20-44e4-4447-8a54-91d47ecfcc13</id>
    <wcm:name></wcm:name>
    <wcm:type>SiteArea</wcm:type>
    <updated>2012-01-31T03:33:17.826Z</updated>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!7K1PhYjxBw0jzCDqHCwg2w/um/users/profiles/Z9eAeHPCAJG963RD2MMG6P9O6MMG66BD6MM47IHP4MMS6M1DAJQ4C1BCAMID653</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!7K1PhYjxBw0jzCDqHCwg2w/um/users/profiles/Z9eAeHPCAJG963RD2MMG6P9O6MMG66BD6MM47IHP4MMS6M1DAJQ4C1BCAMID653</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <link label="Sitearea Template" rel="sitearea-template" href="/wps/mycontenthandler/!ut/p/digest!PQo5Yhy68oeppWcEz2sddA/wcmrest/SiteArea/72456e6a-198c-47f3-8611-659b0ec6624c" lang="en"/>
    <content type="application/vnd.ibm.wcm+xml">
        <siteArea xmlns="http://www.ibm.com/xmlns/wcm">
            <elements>
                <element name="summary">
                    <title lang="en-US">Summary</title>
                    <type>TextComponent</type>
                    <data type="text/plain"><![CDATA[Text inside the element]]&gt;</data>
                </element>
                <element name="body">
                    <title lang="en-US">Body</title>
                    <type>HTMLComponent</type>
                    <data type="text/html"><![CDATA[<p>
Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
]]&gt;</data>
                </element>
            </elements>
        </siteArea>
    </content>
</entry>
```

Update

A site area can be updated by sending a PUT request to the following URI with an Atom entry that includes the fields on the item that need to be changed.

`/SiteArea/`*`item-uuid`*

For example:

```
PUT /wps/mycontenthandler/wcmrest/SiteArea/18001a8c-2117-45d2-be1c-baea28a41769 HTTP/1.0
Content-Type : application/atom+xml

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <title>SampleSiteAreaTitleUpdated</title>
    <wcm:name>SampleSiteAreaNameUpdated</wcm:name>
    <wcm:description>SampleSiteAreaDescriptionUpdated</wcm:description>
</entry>


HTTP/1.0 200 OK
Content-type: application/atom+xml; type=entry

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <id>18001a8c-2117-45d2-be1c-baea28a41769</id>
    <title>SampleSiteAreaTitleUpdated</title>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteArea/18001a8c-2117-45d2-be1c-baea28a41769"/>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/item/54a68ca2-c550-4385-966f-b0b612147547"/>
    <link rel="parent" href="/wps/mycontenthandler/!ut/p/wcmrest/item/ae6a3632-a1b5-456a-866e-e9baab84fe29"/>
    <link rel="create-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/18001a8c-2117-45d2-be1c-baea28a41769/create-draft"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/18001a8c-2117-45d2-be1c-baea28a41769/change-to-draft"/>
    <link rel="versions" href="/wps/mycontenthandler/!ut/p/wcmrest/item/18001a8c-2117-45d2-be1c-baea28a41769/versions"/>
    <link rel="default-content" href="/wps/mycontenthandler/!ut/p/wcmrest/Content/null"/>
    <link rel="elements" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteArea/18001a8c-2117-45d2-be1c-baea28a41769/elements"/>
    <updated>2011-05-30T06:04:25.741Z</updated>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5Ul75Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5Ul75Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:name>SampleSiteAreaNameUpdated</wcm:name>
    <wcm:description>SampleSiteAreaDescriptionUpdated</wcm:description>
    <wcm:type>SiteArea</wcm:type>
    <wcm:state>PUBLISHED</wcm:state>
</entry>
```

Read

A site area can be read by sending a GET request to the following URI: /SiteArea/*item-uuid*

For example:

```
GET /wps/mycontenthandler/wcmrest/SiteArea/18001a8c-2117-45d2-be1c-baea28a41769 HTTP/1.0


HTTP/1.0 200 OK
Content-type: application/atom+xml; type=entry

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <id>18001a8c-2117-45d2-be1c-baea28a41769</id>
    <title>SampleSiteAreaTitleUpdated</title>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteArea/18001a8c-2117-45d2-be1c-baea28a41769"/>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/item/54a68ca2-c550-4385-966f-b0b612147547"/>
    <link rel="parent" href="/wps/mycontenthandler/!ut/p/wcmrest/item/ae6a3632-a1b5-456a-866e-e9baab84fe29"/>
    <link rel="create-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/18001a8c-2117-45d2-be1c-baea28a41769/create-draft"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/18001a8c-2117-45d2-be1c-baea28a41769/change-to-draft"/>
    <link rel="versions" href="/wps/mycontenthandler/!ut/p/wcmrest/item/18001a8c-2117-45d2-be1c-baea28a41769/versions"/>
    <link rel="default-content" href="/wps/mycontenthandler/!ut/p/wcmrest/Content/null"/>
    <link rel="elements" href="/wps/mycontenthandler/!ut/p/wcmrest/SiteArea/18001a8c-2117-45d2-be1c-baea28a41769/elements"/>
    <updated>2011-05-30T06:04:25.741Z</updated>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5Ul75Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5Ul75Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:name>SampleSiteAreaNameUpdated</wcm:name>
    <wcm:description>SampleSiteAreaDescriptionUpdated</wcm:description>
    <wcm:type>SiteArea</wcm:type>
    <wcm:state>PUBLISHED</wcm:state>
</entry>
```

Delete

A site area can be deleted by sending a DELETE request to the following URI: /SiteArea/*item-uuid*

For example:

```
DELETE /wps/mycontenthandler/wcmrest/SiteArea/18001a8c-2117-45d2-be1c-baea28a41769 HTTP/1.0


HTTP/1.0 200 OK
```

How to use REST with managed pages

You can use the Web Content Manager REST service to read managed pages.

Read

A managed page can be read by sending a GET request to the following URI: /PortalPage/<itemuuid>

For example:

```
GET /wps/mycontenthandler/wcmrest/PortalPage/6c523449-e919-41a8-a8d0-f8b1ea207af1 HTTP/1.0
HTTP/1.0 200 OK
Content-type: application/atom+xml; type=entry
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <atom:id>wcmrest:6c523449-e919-41a8-a8d0-f8b1ea207af1</atom:id>
    <atom:title>RestPortalPage1</atom:title>
   <atom:link atom:rel="edit" atom:href="/wps/mycontenthandler/!ut/p/wcmrest/PortalPage/6c523449-e919-41a8-a8d0-f8b1ea207af1"/>
    <atom:link atom:rel="library" atom:href="/wps/mycontenthandler/!ut/p/wcmrest/item/d07f0a12-3801-465e-bc20-eaec2cecf5cb"/>
   <atom:link atom:rel="workflow-stage" atom:href="/wps/mycontenthandler/!ut/p/wcmrest/item/c5cb286b-b31b-4b9a-b8c9-9d0cb4311ae3"/>
    <atom:link atom:rel="workflow" atom:href="/wps/mycontenthandler/!ut/p/wcmrest/item/2165c8e4-1fbe-4f72-9ffb-4a43888bfa87"/>
    <atom:link atom:rel="create-draft" atom:href="/wps/mycontenthandler/!ut/p/wcmrest/item/6c523449-e919-41a8-a8d0-f8b1ea207af1/create-draft"/>
   <atom:link atom:rel="next-stage" atom:href="/wps/mycontenthandler/!ut/p/wcmrest/item/6c523449-e919-41a8-a8d0-f8b1ea207af1/next-stage"/>
    <atom:link atom:rel="previous-stage" atom:href="/wps/mycontenthandler/!ut/p/wcmrest/item/6c523449-e919-41a8-a8d0-f8b1ea207af1/previous-stage"/>
    <atom:link atom:rel="restart" atom:href="/wps/mycontenthandler/!ut/p/wcmrest/item/6c523449-e919-41a8-a8d0-f8b1ea207af1/restart"/>
   <atom:link atom:rel="versions" atom:href="/wps/mycontenthandler/!ut/p/wcmrest/item/6c523449-e919-41a8-a8d0-f8b1ea207af1/versions"/>
   <atom:link atom:rel="elements" atom:href="/wps/mycontenthandler/!ut/p/wcmrest/PortalPage/6c523449-e919-41a8-a8d0-f8b1ea207af1/elements"/>
    <atom:updated>2011-06-30T11:37:27.921Z</atom:updated>
    <wcm:name>RestPortalPage1</wcm:name>
    <wcm:description>RestPortalPage1 Description</wcm:description>
    <wcm:type>PortalPage</wcm:type>
   <atom:category atom:scheme="wcmrest:workflowState">
        <atom:term>PUBLISHED</atom:term>
        <atom:label>None</atom:label>
    </atom:category>
    <atom:author>
        <atom:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</atom:distinguishedName>
        <atom:uri>/um/users/profiles/uid=wpsadmin,o=defaultWIMFileBasedRealm</atom:uri>
        <atom:name>wpsadmin</atom:name>
    </atom:author>
    <atom:owner>
        <atom:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</atom:distinguishedName>
```

```
        <atom:uri>/um/users/profiles/uid=wpsadmin,o=defaultWIMFileBasedRealm</atom:uri>
        <atom:name>wpsadmin</atom:name>
    </atom:owner>
</atom:entry>
```

How to use REST with drafts and workflows

You can use the REST services for Web Content Manager to create drafts, approve items in a workflow, and move items through different stages of a workflow.

Creating a draft of an item that does not use a workflow

You can create a draft of items that do not use workflows.

To do that, specify explicitly the draft workflow state in the request entry data. For example, to create a link component as draft:

```
HTTP/1.1 POST

http://host:port/wps/mycontenthandler/wcmrest/LibraryLinkComponent
<atom:entry xmlns:atom="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <wcm:name>linkDraft_E</wcm:name>
    <atom:link atom:rel="library" atom:href="/wps/mycontenthandler/!ut/p/wcmrest/item/d07f0a12-3801-465e-bc20-eaec2cecf5cb"/>
    <category label="Draft" scheme="wcmrest:workflowState" term="DRAFT"/>
</atom:entry>

201 Created
```

Creating a draft in a workflow

You can use the REST service to create a new draft item. This is equivalent to using the **Create Draft** button in the authoring portlet.

A new draft of an item can be created by sending a POST request to the following URI:

`/item/`*`item-uuid`*`/create-draft`

For example:

```
HTTP/1.1 POST
http://host:port/wps/mycontenthandler/wcmrest/item/item-uuid/create-draft

201 Created
```

Moving an item to the next stage

You can use the REST service to move an item to the next stage of a workflow. This is equivalent to using the **Next Stage** button in the authoring portlet.

An item can be moved to the next stage by sending a POST request to the following URI:

`/item/`*`item-uuid`*`/next-stage`

For example:

```
HTTP/1.1 POST
http://host:port/wps/mycontenthandler/wcmrest/item/<item-uuid>/next-stage

201 Created
```

Moving an item to the previous stage

You can use the REST service to move an item to the previous stage of a workflow. This is equivalent to using the **Previous Stage** button in the authoring portlet.

An item can be moved to the previous stage by sending a POST request to the following URI:

`/item/item-uuid/previous-stage`

For example:

```
HTTP/1.1 POST
http://host:port/wps/mycontenthandler/wcmrest/item/item-uuid/previous-stage

201 Created
```

Approving an item within a workflow

You can use the REST service to move an item to the next stage of a workflow by approving it. This is equivalent to using the **Approve** button in the authoring portlet.

An item can be approved by sending a POST request to the following URI:

`/item/item-uuid/approve`

For example:

```
HTTP/1.1 POST
http://host:port/wps/mycontenthandler/wcmrest/item/item-uuid/approve

201 Created
```

Rejecting an item within a workflow

You can use the REST service to reject an item in a workflow. This is equivalent to using the **Reject** button in the authoring portlet.

An item can be rejected by sending a POST request to the following URI:

`/item/item-uuid/reject`

For example:

```
HTTP/1.1 POST
http://host:port/wps/mycontenthandler/wcmrest/item/item-uuid/reject

201 Created
```

Restarting a workflow

You can use the REST service to restart a workflow. This is equivalent to using the **Restart** button in the authoring portlet.

An item can be restarted by sending a POST request to the following URI:

`/item/item-uuid/restart`

For example:

```
HTTP/1.1 POST
http://host:port/wps/mycontenthandler/wcmrest/item/item-uuid/restart

201 Created
```

Workflow Comments

To ensure that comments are added to workflow stages that require comments, a GET request must be issued to the associated link. This will return an HTML form indicating whether a comment is required.

To add the comment you either submit the form, or perform an HTTP Post request with a content type of `text/plain`. For example:

```
HTTP GET /wps/mycontenthandler/wcmrest/item/84e35979-d7c4-429a-b2ab-eb79abc5debc/next-stage
 Accept: text/html

 <html>
<b>Next Stage</b><br><br>
<form action="/wps/mycontenthandler/!ut/p/digest!hp5aUN2TU2D-7ziKYRo2-g/wcmrest/item/84e35979-d7c4
   <label for="comment"><b>*</b>Enter Comment</label>
   <input id="comment" name="comment" type="text" required/><br>
   <input type="submit" value="Submit"></input>
</form>
</html>
```

How to use REST with workflow items

You can use the Web Content Manager REST service to create, read, update, and delete workflow items.

Create

A workflow item can be created by sending a POST request to the following URI with an Atom entry that represents the workflow item:

`/Workflow`

Note: A library link is required to specify the location to create the workflow.

For example:

```
HTTP/1.1 POST /wps/mycontenthandler/wcmrest/Workflow
Content-Type: application/atom+xml

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <wcm:name>MyWorkflow</wcm:name>
    <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/be3ca1cd-f482-4715-9972-8e683fd0be85"/>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:workflow xmlns="http://www.ibm.com/xmlns/wcm/8.0" xmlns:atom="http://www.w3.org/2005/Atom">
            <option name="REQUIRE_COMMENT_ON_APPROVAL" enabled="true"/>
            <option name="ALLOW_VALIDATION_FAILURES" enabled="true"/>
            <stages>
                <stage href="/wps/mycontenthandler/wcmrest/WorkflowStage/434ec120-aaa1-4ad0-a7fd-62a060206fc9"/>
                <stage href="/wps/mycontenthandler/wcmrest/WorkflowStage/9fca54fb-fcaf-4e99-bc03-8f496510fd4e"/>
                <stage href="/wps/mycontenthandler/wcmrest/WorkflowStage/4d0ff502-0cf4-4a94-9757-f37675202f44"/>
                <reject href="/wps/mycontenthandler/wcmrest/WorkflowStage/4d0ff502-0cf4-4a94-9757-f37675202f44"/>
                <project-exit href="/wps/mycontenthandler/wcmrest/WorkflowStage/9fca54fb-fcaf-4e99-bc03-8f496510fd4e"/>
            </stages>
            <draftCreation>ALLOW_EXCLUSIVE_DRAFTS_ONLY</draftCreation>
        </wcm:workflow>
    </content>
</entry>

HTTP/1.1 201 Created
Content-Type: application/atom+xml
Content-Location: /wps/mycontenthandler/wcmrest/Workflow/b9c05b29-66ac-48fa-9ed7-43e3912f3c54

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:b9c05b29-66ac-48fa-9ed7-43e3912f3c54</id>
    <title xml:lang="en">MyWorkflow</title>
    <summary xml:lang="en"></summary>
    <wcm:name>MyWorkflow</wcm:name>
    <wcm:type>Workflow</wcm:type>
    <updated>2014-06-03T02:03:24.406Z</updated>
    <wcm:created>2014-06-03T02:03:24.406Z</wcm:created>
    ... some content elided...
    <link rel="self" href="/wps/mycontenthandler/wcmrest/Workflow/b9c05b29-66ac-48fa-9ed7-43e3912f3c54" xml:lang="en" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/wcmrest/Workflow/b9c05b29-66ac-48fa-9ed7-43e3912f3c54" xml:lang="en" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/wcmrest/Workflow/b9c05b29-66ac-48fa-9ed7-43e3912f3c54" xml:lang="en" label="Delete"/>
    <link rel="create-draft" href="/wps/mycontenthandler/wcmrest/item/b9c05b29-66ac-48fa-9ed7-43e3912f3c54/create-draft" xml:lang="en" label="Create Draft"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/wcmrest/item/b9c05b29-66ac-48fa-9ed7-43e3912f3c54/change-to-draft" xml:lang="en" label="Change To Draft"/>
    <link rel="access-control" href="/wps/mycontenthandler/ac/access:oid:Z6QReDe2BE66OK62JCIJMO6M9O6MMG6OHP2MM4753PEJMG6J9P6JS46IHP6JHL6K1" xml:lang="en" label="Access Control"/>
    <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/be3ca1cd-f482-4715-9972-8e683fd0be85" xml:lang="en" label="Library"/>
    <link rel="versions" href="/wps/mycontenthandler/wcmrest/item/b9c05b29-66ac-48fa-9ed7-43e3912f3c54/versions" xml:lang="en" label="Versions"/>
    <link rel="edit-media" href="/wps/mycontenthandler/wcmrest/Workflow/b9c05b29-66ac-48fa-9ed7-43e3912f3c54" type="application/vnd.ibm.wcm+xml" xml:lang="en" label="Edit Media"/>
    <category scheme="wcmrest:workflowState" term="PUBLISHED" label="Published" xml:lang="en"/>
    <category scheme="wcmrest:favorite" term="false" xml:lang="en"/>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:workflow xmlns="http://www.ibm.com/xmlns/wcm/8.0" xmlns:atom="http://www.w3.org/2005/Atom">
            <option name="REQUIRE_COMMENT_ON_APPROVAL" enabled="true"/>
            <option name="ALLOW_VALIDATION_FAILURES" enabled="true"/>
            <option name="ALLOW_WORKFLOW_IN_PROJECTS" enabled="false"/>
            <stages>
                <stage href="/wps/mycontenthandler/wcmrest/WorkflowStage/434ec120-aaa1-4ad0-a7fd-62a060206fc9"/>
                <stage href="/wps/mycontenthandler/wcmrest/WorkflowStage/9fca54fb-fcaf-4e99-bc03-8f496510fd4e"/>
```

```
            <stage href="/wps/mycontenthandler/wcmrest/WorkflowStage/4d0ff502-0cf4-4a94-9757-f37675202f44"/>
            <reject href="/wps/mycontenthandler/wcmrest/WorkflowStage/4d0ff502-0cf4-4a94-9757-f37675202f44"/>
            <project-exit href="/wps/mycontenthandler/wcmrest/WorkflowStage/9fca54fb-fcaf-4e99-bc03-8f496510fd4e"/>
        </stages>
        <draftCreation>ALLOW_EXCLUSIVE_DRAFTS_ONLY</draftCreation>
    </wcm:workflow>
</content>
</entry>
```

Update

A workflow item can be updated by sending a PUT request to the following URI with an Atom entry that includes the fields on the item that need to be changed.

`/Workflow/itemuuid`

For example:

```
HTTP/1.1 PUT /wps/mycontenthandler/wcmrest/Workflow/b9c05b29-66ac-48fa-9ed7-43e3912f3c54
Content-Type: application/atom+xml

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <wcm:name>MyWorkflow With A New Name</wcm:name>
    ... some content elided ...
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:workflow xmlns="http://www.ibm.com/xmlns/wcm/8.0" xmlns:atom="http://www.w3.org/2005/Atom">
            <stages>
                <stage href="/wps/mycontenthandler/wcmrest/WorkflowStage/9fdaec77-e342-43a1-beb7-1c7ab21ee35f"/>
                <stage href="/wps/mycontenthandler/wcmrest/WorkflowStage/8eba94a5-dda2-46c4-a9f9-bba5ebac05a2"/>
                <stage href="/wps/mycontenthandler/wcmrest/WorkflowStage/9fbbe443-63f5-49ad-905d-aa54102f377a"/>
            </stages>
            <draftCreation>ALLOW_MULTIPLE_DRAFTS</draftCreation>
        </wcm:workflow>
    </content>
</entry>

HTTP/1.1 200 OK
Content-Type: application/atom+xml

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <wcm:name>MyWorkflow With A New Name</wcm:name>
    ... some content elided ...
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:workflow xmlns="http://www.ibm.com/xmlns/wcm/8.0" xmlns:atom="http://www.w3.org/2005/Atom">
            <stages>
                <stage href="/wps/mycontenthandler/wcmrest/WorkflowStage/9fdaec77-e342-43a1-beb7-1c7ab21ee35f"/>
                <stage href="/wps/mycontenthandler/wcmrest/WorkflowStage/8eba94a5-dda2-46c4-a9f9-bba5ebac05a2"/>
                <stage href="/wps/mycontenthandler/wcmrest/WorkflowStage/9fbbe443-63f5-49ad-905d-aa54102f377a"/>
            </stages>
            <draftCreation>ALLOW_MULTIPLE_DRAFTS</draftCreation>
        </wcm:workflow>
    </content>
</entry>
```

Read

A workflow item can be read by sending a GET request to the following URI:

`/Workflow/itemuuid`

For example:

```
HTTP/1.1 GET /wps/mycontenthandler/wcmrest/Workflow/b9c05b29-66ac-48fa-9ed7-43e3912f3c54
Accept: application/atom+xml

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:b9c05b29-66ac-48fa-9ed7-43e3912f3c54</id>
    <title xml:lang="en">Workflow Create Test</title>
    <summary xml:lang="en"></summary>
    <wcm:name>Workflow Create Test</wcm:name>
    <wcm:type>Workflow</wcm:type>
    <updated>2014-06-03T02:03:24.406Z</updated>
    <wcm:created>2014-06-03T02:03:24.406Z</wcm:created>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAeI1E83S86KPDCMMCC1JP4MMG6G9P6JM8CMPD6MMCCOHOE3Q0713D23S06O1</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAeI1E83S86KPDCMMCC1JP4MMG6G9P6JM8CMPD6MMCCOHOE3Q0713D23S06O1</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAeI1E83S86KPDCMMCC1JP4MMG6G9P6JM8CMPD6MMCCOHOE3Q0713D23S06O1</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
    <wcm:creator>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAeI1E83S86KPDCMMCC1JP4MMG6G9P6JM8CMPD6MMCCOHOE3Q0713D23S06O1</uri>
        <name>wpsadmin</name>
    </wcm:creator>
    <link rel="self" href="/wps/mycontenthandler/wcmrest/Workflow/b9c05b29-66ac-48fa-9ed7-43e3912f3c54" xml:lang="en" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/wcmrest/Workflow/b9c05b29-66ac-48fa-9ed7-43e3912f3c54" xml:lang="en" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/wcmrest/Workflow/b9c05b29-66ac-48fa-9ed7-43e3912f3c54" xml:lang="en" label="Delete"/>
    <link rel="create-draft" href="/wps/mycontenthandler/wcmrest/item/b9c05b29-66ac-48fa-9ed7-43e3912f3c54/create-draft" xml:lang="en" label="Create Draft"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/wcmrest/item/b9c05b29-66ac-48fa-9ed7-43e3912f3c54/change-to-draft" xml:lang="en" label="Change To Draft"/>
    <link rel="access-control" href="/wps/mycontenthandler/ac/access:oid:Z6QReDeD2BE66OK62JCIJMO6M9O6MMG6OHP2MM4753PEJMG6J9P6JS46IHP6JHL6K1" xml:lang="en" label="Access Control"/>
    <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/be3ca1cd-f482-4715-9972-8e683fd0be85" xml:lang="en" label="Library"/>
    <link rel="versions" href="/wps/mycontenthandler/wcmrest/item/b9c05b29-66ac-48fa-9ed7-43e3912f3c54/versions" xml:lang="en" label="Versions"/>
    <link rel="edit-media" href="/wps/mycontenthandler/wcmrest/Workflow/b9c05b29-66ac-48fa-9ed7-43e3912f3c54" type="application/vnd.ibm.wcm+xml" xml:lang="en" label="Edit Media"/>
    <category scheme="wcmrest:workflowState" term="PUBLISHED" label="Published" xml:lang="en"/>
    <category scheme="wcmrest:favorite" term="false" xml:lang="en"/>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:workflow xmlns="http://www.ibm.com/xmlns/wcm/8.0" xmlns:atom="http://www.w3.org/2005/Atom">
            <option name="REQUIRE_COMMENT_ON_APPROVAL" enabled="true"/>
            <option name="ALLOW_VALIDATION_FAILURES" enabled="true"/>
            <option name="ALLOW_WORKFLOW_IN_PROJECTS" enabled="false"/>
```

```
            <stages>
                <stage href="/wps/mycontenthandler/wcmrest/WorkflowStage/434ec120-aaa1-4ad0-a7fd-62a060206fc9"/>
                <stage href="/wps/mycontenthandler/wcmrest/WorkflowStage/9fca54fb-fcaf-4e99-bc03-8f496510fd4e"/>
                <stage href="/wps/mycontenthandler/wcmrest/WorkflowStage/4d0ff502-0cf4-4a94-9757-f37675202f44"/>
                <reject href="/wps/mycontenthandler/wcmrest/WorkflowStage/4d0ff502-0cf4-4a94-9757-f37675202f44"/>
                <project-exit href="/wps/mycontenthandler/wcmrest/WorkflowStage/9fca54fb-fcaf-4e99-bc03-8f496510fd4e"/>
            </stages>
            <draftCreation>ALLOW_EXCLUSIVE_DRAFTS_ONLY</draftCreation>
        </wcm:workflow>
    </content>
</entry>
```

Delete

A workflow item can be deleted by sending a DELETE request to the following URI:

```
/Workflow/itemuuid
```

For example:

```
HTTP/1.1 DELETE /wps/mycontenthandler/wcmrest/Workflow/b9c05b29-66ac-48fa-9ed7-43e3912f3c54

HTTP/1.1 200 OK
```

How to use REST with workflow stages

You can use the Web Content Manager REST service to create, read, update, and delete workflow stages.

Create

A workflow stage is created by sending a POST request to the following URI with an Atom entry that represents the workflow item:

```
/WorkflowStage
```

For example:

```
HTTP/1.1 POST /wps/mycontenthandler/wcmrest/WorkflowStage
Content-Type: application/atom+xml

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <wcm:name>WorkflowStage Create With Access Control Test</wcm:name>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/digest!_Nw2dNJ9qG0BrJuC97Bijw/wcmrest/Library/3e22085a-2ebb-423d-89c2-53d3c4aa593f"/>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:workflowStage xmlns="http://www.ibm.com/xmlns/wcm/8.0" xmlns:atom="http://www.w3.org/2005/Atom">
            <actions>
                <entering href="/wps/mycontenthandler/wcmrest/item/95b0e3d2-ae2b-431a-bd39-31f5c14f5726"/>
                <entering href="/wps/mycontenthandler/wcmrest/item/24fef197-3b88-45aa-88ef-c49623d95195"/>
                <exiting href="/wps/mycontenthandler/wcmrest/item/24fef197-3b88-45aa-88ef-c49623d95195"/>
                <exiting href="/wps/mycontenthandler/wcmrest/item/95b0e3d2-ae2b-431a-bd39-31f5c14f5726"/>
            </actions>
            <jointApproval enabled="true">
                <approver>
                    <distinguishedName>uid=WCMUT_Editor_A,o=defaultWIMFileBasedRealm</distinguishedName>
                    <atom:name>uid=WCMUT_Editor_A,o=defaultWIMFileBasedRealm</atom:name>
                </approver>
                <approver>
                    <distinguishedName>uid=WCMUT_Editor_B,o=defaultWIMFileBasedRealm</distinguishedName>
                    <atom:name>uid=WCMUT_Editor_B,o=defaultWIMFileBasedRealm</atom:name>
                </approver>
            </jointApproval>
            <workflowDefinedAccess>
                <role name="User">
                    <member>
                        <distinguishedName>uid=WCMUT_Editor_A,o=defaultWIMFileBasedRealm</distinguishedName>
                        <atom:name>uid=WCMUT_Editor_A,o=defaultWIMFileBasedRealm</atom:name>
                    </member>
                    <member>
                        <distinguishedName>uid=WCMUT_Editor_B,o=defaultWIMFileBasedRealm</distinguishedName>
                        <atom:name>uid=WCMUT_Editor_B,o=defaultWIMFileBasedRealm</atom:name>
                    </member>
                </role>
                <role name="Manager">
                    <member>
                        <distinguishedName>uid=WCMUT_Editor_A,o=defaultWIMFileBasedRealm</distinguishedName>
                        <atom:name>uid=WCMUT_Editor_A,o=defaultWIMFileBasedRealm</atom:name>
                    </member>
                    <member>
                        <distinguishedName>uid=WCMUT_Editor_B,o=defaultWIMFileBasedRealm</distinguishedName>
                        <atom:name>uid=WCMUT_Editor_B,o=defaultWIMFileBasedRealm</atom:name>
                    </member>
                </role>
            </workflowDefinedAccess>
        </wcm:workflowStage>
    </content>
</entry>



HTTP/1.1 201 Created
Content-Type: application/atom+xml
Content-Location: /wps/mycontenthandler/wcmrest/WorkflowStage/141793eb-afa7-428e-bb7b-070a25ee3d7c

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    ... some content elided ...
    <wcm:name>WorkflowStage Create With Access Control Test</wcm:name>
    ... some content elided ...
    <link rel="library" href="/wps/mycontenthandler/!ut/p/digest!_Nw2dNJ9qG0BrJuC97Bijw/wcmrest/Library/3e22085a-2ebb-423d-89c2-53d3c4aa593f"/>
```

```
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:workflowStage xmlns="http://www.ibm.com/xmlns/wcm/8.0" xmlns:atom="http://www.w3.org/2005/Atom">
            <actions>
                <entering href="/wps/mycontenthandler/wcmrest/item/95b0e3d2-ae2b-431a-bd39-31f5c14f5726"/>
                <entering href="/wps/mycontenthandler/wcmrest/item/24fef197-3b88-45aa-88ef-c49623d95195"/>
                <exiting href="/wps/mycontenthandler/wcmrest/item/24fef197-3b88-45aa-88ef-c49623d95195"/>
                <exiting href="/wps/mycontenthandler/wcmrest/item/95b0e3d2-ae2b-431a-bd39-31f5c14f5726"/>
            </actions>
            <jointApproval enabled="true">
                <approver>
                    <distinguishedName>uid=WCMUT_Editor_A,o=defaultWIMFileBasedRealm</distinguishedName>
                    <atom:name>uid=WCMUT_Editor_A,o=defaultWIMFileBasedRealm</atom:name>
                </approver>
                <approver>
                    <distinguishedName>uid=WCMUT_Editor_B,o=defaultWIMFileBasedRealm</distinguishedName>
                    <atom:name>uid=WCMUT_Editor_B,o=defaultWIMFileBasedRealm</atom:name>
                </approver>
            </jointApproval>
            <workflowDefinedAccess>
                <role name="User">
                    <member>
                        <distinguishedName>uid=WCMUT_Editor_A,o=defaultWIMFileBasedRealm</distinguishedName>
                        <atom:name>uid=WCMUT_Editor_A,o=defaultWIMFileBasedRealm</atom:name>
                    </member>
                    <member>
                        <distinguishedName>uid=WCMUT_Editor_B,o=defaultWIMFileBasedRealm</distinguishedName>
                        <atom:name>uid=WCMUT_Editor_B,o=defaultWIMFileBasedRealm</atom:name>
                    </member>
                </role>
                <role name="Manager">
                    <member>
                        <distinguishedName>uid=WCMUT_Editor_A,o=defaultWIMFileBasedRealm</distinguishedName>
                        <atom:name>uid=WCMUT_Editor_A,o=defaultWIMFileBasedRealm</atom:name>
                    </member>
                    <member>
                        <distinguishedName>uid=WCMUT_Editor_B,o=defaultWIMFileBasedRealm</distinguishedName>
                        <atom:name>uid=WCMUT_Editor_B,o=defaultWIMFileBasedRealm</atom:name>
                    </member>
                </role>
            </workflowDefinedAccess>
        </wcm:workflowStage>
    </content>
</entry>
```

Update

A workflow stage is updated by sending a PUT request to the following URI with an Atom entry that includes the fields on the item that need to be changed.

`/WorkflowStage/workflow-stage-id`

For example:

```
HTTP/1.1 PUT /wps/mycontenthandler/wcmrest/WorkflowStage/141793eb-afa7-428e-bb7b-070a25ee3d7c
Content-Type: application/atom+xml

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    ... some content elided ...
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:workflowStage xmlns="http://www.ibm.com/xmlns/wcm/8.0" xmlns:atom="http://www.w3.org/2005/Atom">
            <option name="ENABLE_PREVIOUS_STAGE_FOR_REVIEWERS" enabled="false"/>
            <option name="REQUIRE_COMMENT_ON_APPROVAL" enabled="false"/>
            <actions>
                <entering href="/wps/mycontenthandler/wcmrest/item/a9198c77-b5fa-4205-aa1a-e12d4a7ad832"/>
            </actions>
            <jointApproval enabled="false"/>
            <workflowDefinedAccess>
                <role name="User" inheritance="true" propagation="true"/>
                <role name="PrivilegedUser" inheritance="true" propagation="true"/>
                <role name="MarkupEditor" inheritance="true" propagation="true"/>
                <role name="Reviewer" inheritance="true" propagation="true"/>
                <role name="DraftCreator" inheritance="true" propagation="true"/>
                <role name="Contributor" inheritance="true" propagation="true"/>
                <role name="Editor" inheritance="true" propagation="true"/>
                <role name="Manager" inheritance="true" propagation="true"/>
                <role name="SecurityAdmin" inheritance="true" propagation="true"/>
                <role name="Admin" inheritance="true" propagation="true"/>
            </workflowDefinedAccess>
        </wcm:workflowStage>
    </content>
</entry>

HTTP/1.1 200 OK
Content-Type: application/atom+xml
```

Read

A workflow stage is read by sending a GET request to the following URI:

`/WorkflowStage/workflow-stage-id`

For example:

```
HTTP/1.1 GET /wps/mycontenthandler/wcmrest/WorkflowStage/7b40f5b6-bf52-4b9e-8062-b0755aaf1f80
Accept: application/atom+xml

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:7b40f5b6-bf52-4b9e-8062-b0755aaf1f80</id>
    <title xml:lang="en">Publish Stage</title>
    <wcm:displayTitle xml:lang="en">Publish Stage</wcm:displayTitle>
    <wcm:titleTextProviderName>com.ibm.wps.plugins.WebResourcesTextProvider</wcm:titleTextProviderName>
    <wcm:titleTextProviderKey>OOB_PUBLISH_WORKFLOW_STAGE</wcm:titleTextProviderKey>
    <summary xml:lang="en"></summary>
    <wcm:name>Publish Stage</wcm:name>
    <wcm:type>WorkflowStage</wcm:type>
    <updated>2014-04-24T01:31:59.685Z</updated>
    <wcm:created>2010-02-01T02:04:00.686Z</wcm:created>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
```

```
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAeI1E83S86KPDCMMCC1JP4MMG6G9P6JM8CMPD6MMCCOHOE3Q0713D23S06O1</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
    <wcm:creator>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAeI1E83S86KPDCMMCC1JP4MMG6G9P6JM8CMPD6MMCCOHOE3Q0713D23S06O1</uri>
        <name>wpsadmin</name>
    </wcm:creator>
    <link rel="self" href="/wps/mycontenthandler/wcmrest/WorkflowStage/7b40f5b6-bf52-4b9e-8062-b0755aaf1f80" xml:lang="en" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/wcmrest/WorkflowStage/7b40f5b6-bf52-4b9e-8062-b0755aaf1f80" xml:lang="en" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/wcmrest/WorkflowStage/7b40f5b6-bf52-4b9e-8062-b0755aaf1f80" xml:lang="en" label="Delete"/>
    <link rel="create-draft" href="/wps/mycontenthandler/wcmrest/item/7b40f5b6-bf52-4b9e-8062-b0755aaf1f80/create-draft" xml:lang="en" label="Create Draft"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/wcmrest/item/7b40f5b6-bf52-4b9e-8062-b0755aaf1f80/change-to-draft" xml:lang="en" label="Change To Draft"/>
    <link rel="access-control" href="/wps/mycontenthandler/ac/access:oid:Z6QReDeNHO83OOCLHOCJM8C6BD4JMG62BEAMM07GHD4JM8CGPDAJQ4C1JP23JJ17G1" xml:lang="en" label="Access Control"/>
    <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/f68ddb0c-c06b-43a9-84fd-d43552980e46" xml:lang="en" label="Library"/>
    <link rel="versions" href="/wps/mycontenthandler/wcmrest/item/7b40f5b6-bf52-4b9e-8062-b0755aaf1f80/versions" xml:lang="en" label="Versions"/>
    <link rel="edit-media" href="/wps/mycontenthandler/wcmrest/WorkflowStage/7b40f5b6-bf52-4b9e-8062-b0755aaf1f80" type="application/vnd.ibm.wcm+xml" xml:lang="en" label="Edit Media"/>
    <category scheme="wcmrest:workflowState" term="PUBLISHED" label="Published" xml:lang="en"/>
    <category scheme="wcmrest:favorite" term="false" xml:lang="en"/>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:workflowStage xmlns="http://www.ibm.com/xmlns/wcm/8.0" xmlns:atom="http://www.w3.org/2005/Atom">
            <option name="ENABLE_PREVIOUS_STAGE_FOR_REVIEWERS" enabled="false"/>
            <option name="REQUIRE_COMMENT_ON_APPROVAL" enabled="false"/>
            <actions>
                <entering href="/wps/mycontenthandler/wcmrest/item/b30c40a2-8962-4393-b277-4a64f7a149af"/>
            </actions>
            <jointApproval enabled="false"/>
            <workflowDefinedAccess>
                <role name="User" inheritance="true" propagation="true">
                    <member>
                        <distinguishedName>all_auth_portal_users</distinguishedName>
                        <atom:uri>/wps/mycontenthandler/um/groups/profiles/Z8eAe13RO6G4CL3TGMIPDKBQ6MGHE53P02OTDI3T26M14LRSA6PDE</atom:uri>
                        <atom:name>[all authenticated portal users]</atom:name>
                    </member>
                    <member>
                        <distinguishedName>anonymous_user</distinguishedName>
                        <atom:uri>/wps/mycontenthandler/um/users/profiles/Z9eAe1JRU6N5FDRRANP14GRR47Q5CC38ANPLCI3</atom:uri>
                        <atom:name>[anonymous portal user]</atom:name>
                    </member>
                </role>
                <role name="PrivilegedUser" inheritance="true" propagation="true"/>
                <role name="MarkupEditor" inheritance="true" propagation="true"/>
                <role name="Reviewer" inheritance="true" propagation="true"/>
                <role name="DraftCreator" inheritance="true" propagation="true"/>
                <role name="Contributor" inheritance="true" propagation="true"/>
                <role name="Editor" inheritance="true" propagation="true">
                    <member>
                        <distinguishedName>author_user</distinguishedName>
                        <atom:name>[authors]</atom:name>
                    </member>
                </role>
                <role name="Manager" inheritance="true" propagation="true"/>
                <role name="SecurityAdmin" inheritance="true" propagation="true"/>
                <role name="Admin" inheritance="true" propagation="true"/>
            </workflowDefinedAccess>
        </wcm:workflowStage>
    </content>
</entry>
```

Delete

A workflow item can be deleted by sending a DELETE request to the following URI:

```
/WorkflowStage/workflow-stage-id
```

For example:

```
HTTP/1.1 DELETE /wps/mycontenthandler/wcmrest/WorkflowStage/b9c05b29-66ac-48fa-9ed7-43e3912f3c54

HTTP/1.1 200 OK
```

How to use REST with workflow actions

You can use the Web Content Manager REST service to create, read, update, and delete all workflow action types.

"How to use REST with workflow publish, expire, or version actions" on page 3360
The properties of publish actions, expire actions and version actions are identical. Only the URI used to run the actions are different.

"How to use REST with scheduled move workflow actions" on page 3362
You can use the Web Content Manager REST service to create, read, update, and delete scheduled move workflow actions.

"How to use REST with email workflow actions" on page 3364
You can use the Web Content Manager REST service to create, read, update, and delete email workflow actions.

"How to use REST with custom workflow actions" on page 3366
You can use the Web Content Manager REST service to create, read, update, and delete custom workflow actions.

How to use REST with workflow publish, expire, or version actions:

The properties of publish actions, expire actions and version actions are identical. Only the URI used to run the actions are different.

Create

A publish action can be created by sending a POST request to the following URI with an Atom entry that represents the action:

`/PublishAction`

An expire action can be created by sending a POST request to the following URI with an Atom entry that represents the action:

`/ExpireAction`

A version action can be created by sending a POST request to the following URI with an Atom entry that represents the action:

`/VersionAction`

In this example, the type `PublishAction` can be replaced with `ExpireAction` or `VersionAction` when needed.

```
HTTP/1.1 POST /wps/mycontenthandler/wcmrest/PublishAction
Content-Type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <title>My Publish Action</title>
    <summary>This action publishes content to the live site</summary>
    <wcm:name>My Publish Action</wcm:name>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/digest!muAe8T8GIzS4EJeiF9a_sw/wcmrest/Library/03da7ddc-1bc9-47cd-ab69-b06f23a3f284" />
</entry>


HTTP/1.1 201 Created
Content-Type: application/atom+xml
Content-Location: /wps/mycontenthandler/wcmrest/PublishAction/9c1d8e5b-0e0d-467b-8f1d-58ed550f4236

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:9c1d8e5b-0e0d-467b-8f1d-58ed550f4236</id>
    <title xml:lang="en">My Publish Action</title>
    <summary xml:lang="en">This action publishes content to the live site</summary>
    <wcm:name>My Publish Action</wcm:name>
    <wcm:type>PublishAction</wcm:type>
    <updated>2014-06-24T02:19:42.060Z</updated>
    <wcm:created>2014-06-24T02:19:42.060Z</wcm:created>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
    <wcm:creator>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:creator>
    <link rel="self" href="/wps/mycontenthandler/wcmrest/PublishAction/9c1d8e5b-0e0d-467b-8f1d-58ed550f4236" xml:lang="en" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/wcmrest/PublishAction/9c1d8e5b-0e0d-467b-8f1d-58ed550f4236" xml:lang="en" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/wcmrest/PublishAction/9c1d8e5b-0e0d-467b-8f1d-58ed550f4236" xml:lang="en" label="Delete"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/wcmrest/item/9c1d8e5b-0e0d-467b-8f1d-58ed550f4236/change-to-draft" xml:lang="en" label="Change To Draft"/>
    <link rel="create-draft" href="/wps/mycontenthandler/wcmrest/item/9c1d8e5b-0e0d-467b-8f1d-58ed550f4236/create-draft" xml:lang="en" label="Create Draft"/>
    <link rel="access-control" href="/wps/mycontenthandler/ac/access:oid:Z6QReDePPO231175BD4MM0653C8MMG6MPD4MM076BC8MMK6O9P8MQK6GHP83PC6M1" xml:lang="en" label="Access Control"/>
    <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/03da7ddc-1bc9-47cd-ab69-b06f23a3f284" xml:lang="en" label="Library"/>
    <link rel="versions" href="/wps/mycontenthandler/wcmrest/item/9c1d8e5b-0e0d-467b-8f1d-58ed550f4236/versions" xml:lang="en" label="Versions"/>
    <category scheme="wcmrest:workflowState" term="PUBLISHED" label="Published" xml:lang="en"/>
    <category scheme="wcmrest:favorite" term="false" xml:lang="en"/>
</entry>
```

Update

A publish action can be updated by sending a PUT request to the following URI with an Atom entry that includes the fields on the item that need to be changed.

`/PublishAction/action-id`

An expire action can be updated by sending a PUT request to the following URI with an Atom entry that includes the fields on the item that need to be changed.

/ExpireAction/action-id

A version action can be updated by sending a PUT request to the following URI with an Atom entry that includes the fields on the item that need to be changed.
/VersionAction/action-id

In this example, the type PublishAction can be replaced with ExpireAction or VersionAction when needed.

```
HTTP/1.1 PUT /wps/mycontenthandler/wcmrest/WorkflowStage/141793eb-afa7-428e-bb7b-070a25ee3d7c
Content-Type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:9c1d8e5b-0e0d-467b-8f1d-58ed550f4236</id>
    <title xml:lang="en">My Publish Action</title>
    <summary xml:lang="en">This action publishes content to the live site</summary>
    <wcm:name>My Publish Action - now with a different name</wcm:name>
    <wcm:type>PublishAction</wcm:type>
    <updated>2014-06-24T02:19:42.060Z</updated>
    <wcm:created>2014-06-24T02:19:42.060Z</wcm:created>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
    <wcm:creator>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:creator>
    <link rel="self" href="/wps/mycontenthandler/wcmrest/PublishAction/9c1d8e5b-0e0d-467b-8f1d-58ed550f4236" xml:lang="en" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/wcmrest/PublishAction/9c1d8e5b-0e0d-467b-8f1d-58ed550f4236" xml:lang="en" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/wcmrest/PublishAction/9c1d8e5b-0e0d-467b-8f1d-58ed550f4236" xml:lang="en" label="Delete"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/wcmrest/item/9c1d8e5b-0e0d-467b-8f1d-58ed550f4236/change-to-draft" xml:lang="en" label="Change To Draft"/>
    <link rel="create-draft" href="/wps/mycontenthandler/wcmrest/item/9c1d8e5b-0e0d-467b-8f1d-58ed550f4236/create-draft" xml:lang="en" label="Create Draft"/>
    <link rel="access-control" href="/wps/mycontenthandler/ac/access:oid:Z6QReDePPO23I175BD4MM0653C8MMG6MPD4MM076BC8MMK6O9P8MQK6GHP83PC6M1" xml:lang="en" label="Access Control"/>
    <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/03da7ddc-1bc9-47cd-ab69-b06f23a3f284" xml:lang="en" label="Library"/>
    <link rel="versions" href="/wps/mycontenthandler/wcmrest/item/9c1d8e5b-0e0d-467b-8f1d-58ed550f4236/versions" xml:lang="en" label="Versions"/>
    <category scheme="wcmrest:workflowState" term="PUBLISHED" label="Published" xml:lang="en"/>
    <category scheme="wcmrest:favorite" term="false" xml:lang="en"/>
</entry>

HTTP/1.1 200 OK
```

Read

A publish action can be read by sending a GET request to the following URI:
/PublishAction/action-id

An expire action can be read by sending a GET request to the following URI:
/ExpireAction/action-id

A version action can be read by sending a GET request to the following URI:
/VersionAction/action-id

In this example, the type PublishAction can be replaced with ExpireAction or VersionAction when needed.

```
HTTP/1.1 GET /wps/mycontenthandler/wcmrest/PublishAction/9c1d8e5b-0e0d-467b-8f1d-58ed550f4236
Accept: application/atom+xml

HTTP/1.1 200 OK
Content-Type: application/atom+xml
<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:9c1d8e5b-0e0d-467b-8f1d-58ed550f4236</id>
    <title xml:lang="en">My Publish Action</title>
    <summary xml:lang="en">This action publishes content to the live site</summary>
    <wcm:name>My Publish Action</wcm:name>
    <wcm:type>PublishAction</wcm:type>
    <updated>2014-06-24T02:19:42.060Z</updated>
    <wcm:created>2014-06-24T02:19:42.060Z</wcm:created>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
```

```
    <wcm:creator>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:creator>
    <link rel="self" href="/wps/mycontenthandler/wcmrest/PublishAction/9c1d8e5b-0e0d-467b-8f1d-58ed550f4236" xml:lang="en" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/wcmrest/PublishAction/9c1d8e5b-0e0d-467b-8f1d-58ed550f4236" xml:lang="en" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/wcmrest/PublishAction/9c1d8e5b-0e0d-467b-8f1d-58ed550f4236" xml:lang="en" label="Delete"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/wcmrest/item/9c1d8e5b-0e0d-467b-8f1d-58ed550f4236/change-to-draft" xml:lang="en" label="Change To Draft"/>
    <link rel="create-draft" href="/wps/mycontenthandler/wcmrest/item/9c1d8e5b-0e0d-467b-8f1d-58ed550f4236/create-draft" xml:lang="en" label="Create Draft"/>
    <link rel="access-control" href="/wps/mycontenthandler/ac/access:oid:Z6QReDePPO23I175BD4MM0653C8MMG6MPD4MM076BC8MMK6O9P8MQK6GHP83PC6M1" xml:lang="en" label="Access Control"/>
    <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/03da7ddc-1bc9-47cd-ab69-b06f23a3f284" xml:lang="en" label="Library"/>
    <link rel="versions" href="/wps/mycontenthandler/wcmrest/item/9c1d8e5b-0e0d-467b-8f1d-58ed550f4236/versions" xml:lang="en" label="Versions"/>
    <category scheme="wcmrest:workflowState" term="PUBLISHED" label="Published" xml:lang="en"/>
    <category scheme="wcmrest:favorite" term="false" xml:lang="en"/>
</entry>
```

Delete

A publish action can be deleted by sending a DELETE request to the following URI:

`/PublishAction/action-id`

A expire action can be deleted by sending a DELETE request to the following URI:

`/ExpireAction/action-id`

A version action can be deleted by sending a DELETE request to the following URI:

`/VersionAction/action-id`

In this example, the type `PublishAction` can be replaced with `ExpireAction` or `VersionAction` when needed.

```
HTTP/1.1 DELETE /wps/mycontenthandler/wcmrest/PublishAction/7b40f5b6-bf52-4b9e-8062-b0755aaf1f80

HTTP/1.1 200 OK
```

How to use REST with scheduled move workflow actions:

You can use the Web Content Manager REST service to create, read, update, and delete scheduled move workflow actions.

Create

A scheduled move action can be created by sending a POST request to the following URI with an Atom entry that represents the action:

`/ScheduledMoveAction`

For example:

```
HTTP/1.1 POST /wps/mycontenthandler/wcmrest/ScheduledMoveAction
Content-Type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <wcm:name>My Scheduled Move Action</wcm:name>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:scheduledMoveAction xmlns="http://www.ibm.com/xmlns/wcm/8.0">
            <dateType>LIVE_DATE</type>
            <offset unit="DAY" amount="3" direction="AFTER">
                <time-of-day>15:42:12</time-of-day>
            </offset>
        </wcm:scheduledMoveAction>
    </content>
</entry>

HTTP/1.1 201 Created

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <wcm:name>My Scheduled Move Action</wcm:name>
    ... some content elided ...
    <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/624ca7c0-86ac-4e17-9167-25004c84aeca" xml:lang="en" label="Library"/>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:scheduledMoveAction xmlns="http://www.ibm.com/xmlns/wcm/8.0">
            <dateType>LIVE_DATE</type>
            <offset unit="DAY" amount="3" direction="AFTER">
                <time-of-day>15:42:12</time-of-day>
            </offset>
        </wcm:scheduledMoveAction>
    </content>
</entry>
```

Update

A scheduled move action can be updated by sending a PUT request to the following URI:

`/ScheduledMoveAction/action-id`

For example:

```
HTTP/1.1 PUT /wps/mycontenthandler/wcmrest/ScheduledMoveAction/abc4c24a-3540-4ae3-8ba6-f2f82a977046
Content-Type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    ... some content elided ...
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:scheduledMoveAction xmlns="http://www.ibm.com/xmlns/wcm/8.0">
            <dateType>LIVE_DATE</type>
            <offset unit="HOUR" amount="1" direction="BEFORE">
                <time-of-day>15:42:12</time-of-day>
            </offset>
        </wcm:scheduledMoveAction>
    </content>
</entry>

HTTP/1.1 200 OK
```

Read

A scheduled move action can be read by sending a GET request to the following URI:

`/ScheduledMoveAction/action-id`

For example:

```
HTTP/1.1 GET /wps/mycontenthandler/wcmrest/ScheduledMoveAction/abc4c24a-3540-4ae3-8ba6-f2f82a977046
Accept: application/atom+xml

HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:abc4c24a-3540-4ae3-8ba6-f2f82a977046</id>
    <title xml:lang="en">My Scheduled Move Action</title>
    <summary xml:lang="en"></summary>
    <wcm:name>My Scheduled Move Action</wcm:name>
    <wcm:type>ScheduledMoveAction</wcm:type>
    <updated>2014-06-24T04:51:00.159Z</updated>
    <wcm:created>2014-06-24T04:51:00.159Z</wcm:created>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
    <wcm:creator>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:creator>
    <link rel="self" href="/wps/mycontenthandler/wcmrest/ScheduledMoveAction/abc4c24a-3540-4ae3-8ba6-f2f82a977046" xml:lang="en" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/wcmrest/ScheduledMoveAction/abc4c24a-3540-4ae3-8ba6-f2f82a977046" xml:lang="en" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/wcmrest/ScheduledMoveAction/abc4c24a-3540-4ae3-8ba6-f2f82a977046" xml:lang="en" label="Delete"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/wcmrest/item/abc4c24a-3540-4ae3-8ba6-f2f82a977046/change-to-draft" xml:lang="en" label="Change To Draft"/>
    <link rel="create-draft" href="/wps/mycontenthandler/wcmrest/item/abc4c24a-3540-4ae3-8ba6-f2f82a977046/create-draft" xml:lang="en" label="Create Draft"/>
    <link rel="access-control" href="/wps/mycontenthandler/ac/access:oid:Z6QReDelJO66QCC1D2MMC6L1D0JMG61BP6JM072BOCJMOCIHPG3P4CPPDE3OG6M1" xml:lang="en" label="Access Control"/>
    <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/624ca7c0-86ac-4e17-9167-25004c84aeca" xml:lang="en" label="Library"/>
    <link rel="versions" href="/wps/mycontenthandler/wcmrest/item/abc4c24a-3540-4ae3-8ba6-f2f82a977046/versions" xml:lang="en" label="Versions"/>
    <link rel="edit-media" href="/wps/mycontenthandler/wcmrest/ScheduledMoveAction/abc4c24a-3540-4ae3-8ba6-f2f82a977046" type="application/vnd.ibm.wcm+xml" xml:lang="en" label="Edit Media"/>
    <category scheme="wcmrest:workflowState" term="PUBLISHED" label="Published" xml:lang="en"/>
    <category scheme="wcmrest:favorite" term="false" xml:lang="en"/>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:scheduledMoveAction xmlns="http://www.ibm.com/xmlns/wcm/8.0">
            <dateType>LIVE_DATE</type>
            <offset unit="DAY" amount="3" direction="AFTER">
                <time-of-day>15:42:12</time-of-day>
            </offset>
        </wcm:scheduledMoveAction>
    </content>
</entry>
```

Delete

A scheduled move action can be deleted by sending a DELETE request to the following URI:

`/ScheduledMoveAction/action-id`

For example:

```
HTTP/1.1 DELETE /wps/mycontenthandler/wcmrest/ScheduledMoveAction/abc4c24a-3540-4ae3-8ba6-f2f82a9770
```

```
HTTP/1.1 200 OK
```

How to use REST with email workflow actions:

You can use the Web Content Manager REST service to create, read, update, and delete email workflow actions.

Create

An email action can be created by sending a POST request to the following URI:
`/EmailAction`

For example:

```
HTTP/1.1 POST /wps/mycontenthandler/wcmrest/EmailAction
Content-Type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <wcm:name>My Email Action</wcm:name>
    <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/1f5955c8-38e7-41f9-9029-5c4c0b151976" xml:lang="en" label="Library"/>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:emailAction xmlns="http://www.ibm.com/xmlns/wcm/8.0" xmlns:atom="http://www.w3.org/2005/Atom">
            <dateType>GENERAL_DATE_ONE</dateType>
            <offset unit="MONTH" amount="1" direction="AFTER">
                <time-of-day>11:53:00</time-of-day>
            </offset>
            <recipients>
                <stageApprovers>true</stageApprovers>
                <authors of-item="true" of-referencing-items="true"/>
                <owners of-item="true" of-referencing-items="true"/>
                <additional>
                    <distinguishedName>uid=WCMUT_Editor_A,o=defaultWIMFileBasedRealm</distinguishedName>
                    <atom:name>uid=WCMUT_Editor_A,o=defaultWIMFileBasedRealm</atom:name>
                </additional>
            </recipients>
            <emailText>this is some email text</emailText>
        </wcm:emailAction>
    </content>
</entry>

HTTP/1.1 201 Created

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    ... some content elided ...
    <wcm:name>My Email Action</wcm:name>
    ... some content elided ...
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:emailAction xmlns="http://www.ibm.com/xmlns/wcm/8.0" xmlns:atom="http://www.w3.org/2005/Atom">
            <dateType>GENERAL_DATE_ONE</dateType>
            <offset unit="MONTH" amount="1" direction="AFTER">
                <time-of-day>11:53:00</time-of-day>
            </offset>
            <recipients>
                <stageApprovers>true</stageApprovers>
                <authors of-item="true" of-referencing-items="true"/>
                <owners of-item="true" of-referencing-items="true"/>
                <additional>
                    <distinguishedName>uid=WCMUT_Editor_A,o=defaultWIMFileBasedRealm</distinguishedName>
                    <atom:uri>/wps/mycontenthandler/!ut/p/digest!TPAVlKazgS66elu1F8nIzA/um/users/profiles/Z9eAe5RO8JOCCKHP4MMOCNPOAMMG6M9O6JM07I9CCMMC64BP8JOOCK1C4MGLCI1</atom:uri>
                    <atom:name>WCMUT_Editor_A WCMUT_Editor_A</atom:name>
                </additional>
            </recipients>
            <emailText>this is some email text</emailText>
        </wcm:emailAction>
    </content>
</entry>
```

Update

An email action can be updated by sending a PUT request to the following URI:
`/EmailAction/action-id`

For example:

```
HTTP/1.1 PUT /wps/mycontenthandler/wcmrest/EmailAction/abc4c24a-3540-4ae3-8ba6-f2f82a977046
Content-Type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    ... some content elided ...
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:emailAction xmlns="http://www.ibm.com/xmlns/wcm/8.0" xmlns:atom="http://www.w3.org/2005/Atom">
            <dateType>DATE_ENTERED</dateType>
            <recipients>
                <stageApprovers>false</stageApprovers>
                <authors of-item="false" of-referencing-items="false"/>
                <owners of-item="false" of-referencing-items="false"/>
                <additional>
                    <distinguishedName>uid=WCMUT_Contributor_A,o=defaultWIMFileBasedRealm</distinguishedName>
                    <atom:uri>/wps/mycontenthandler/um/users/profiles/Z9eAeJHPGJP86M9EAMMG6K9PAMMG62RDGJM4C1JP0JM06IHP4JPK61RDCMIP6K1</atom:uri>
                    <atom:name>WCMUT_Contributor_A WCMUT_Contributor_A</atom:name>
                </additional>
                <additional>
                    <distinguishedName>uid=WCMUT_Contributor_B,o=defaultWIMFileBasedRealm</distinguishedName>
                    <atom:uri>/wps/mycontenthandler/um/users/profiles/Z9eAeKPD83H16JHC2JM47KPOCJMG6IHC6MM8C3JCIJMOC4BEAMR06N9E8MQKC63</atom:uri>
```

```
                <atom:name>WCMUT_Contributor_B WCMUT_Contributor_B</atom:name>
            </additional>
        </recipients>
    </wcm:emailAction>
</content>
</entry>

HTTP/1.1 200 OK
```

Read

An email action can be read by sending a GET request to the following URI: `/EmailAction/action-id`

For example:

```
HTTP/1.1 GET /wps/mycontenthandler/wcmrest/EmailAction/64c1a38f-54e9-4852-b1e0-a52dbf40d494
Accept: application/atom+xml

HTTP/1.1 200 OK


<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:64c1a38f-54e9-4852-b1e0-a52dbf40d494</id>
    <title xml:lang="en">My Email Action</title>
    <summary xml:lang="en"></summary>
    <wcm:name>My Email Action</wcm:name>
    <wcm:type>EmailAction</wcm:type>
    <updated>2014-06-24T01:54:38.593Z</updated>
    <wcm:created>2014-06-24T01:54:38.593Z</wcm:created>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
    <wcm:creator>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:creator>
    <link rel="self" href="/wps/mycontenthandler/wcmrest/EmailAction/64c1a38f-54e9-4852-b1e0-a52dbf40d494" xml:lang="en" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/wcmrest/EmailAction/64c1a38f-54e9-4852-b1e0-a52dbf40d494" xml:lang="en" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/wcmrest/EmailAction/64c1a38f-54e9-4852-b1e0-a52dbf40d494" xml:lang="en" label="Delete"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/wcmrest/item/64c1a38f-54e9-4852-b1e0-a52dbf40d494/change-to-draft" xml:lang="en" label="Change To Draft"/>
    <link rel="create-draft" href="/wps/mycontenthandler/wcmrest/item/64c1a38f-54e9-4852-b1e0-a52dbf40d494/create-draft" xml:lang="en" label="Create Draft"/>
    <link rel="access-control" href="/wps/mycontenthandler/ac/access:oid:Z6QReDeM1D6MO4CJ1ECMMK6K9P1JMG6O9D4JM8CH9P0JM4CLHC86HPCK1C86Q47K1" xml:lang="en" label="Access Control"/>
    <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/790b8ca6-5d89-4f37-9052-783dd580f860" xml:lang="en" label="Library"/>
    <link rel="versions" href="/wps/mycontenthandler/wcmrest/item/64c1a38f-54e9-4852-b1e0-a52dbf40d494/versions" xml:lang="en" label="Versions"/>
    <link rel="edit-media" href="/wps/mycontenthandler/wcmrest/EmailAction/64c1a38f-54e9-4852-b1e0-a52dbf40d494" type="application/vnd.ibm.wcm+xml" xml:lang="en" label="Edit Media"/>
    <category scheme="wcmrest:workflowState" term="PUBLISHED" label="Published" xml:lang="en"/>
    <category scheme="wcmrest:favorite" term="false" xml:lang="en"/>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:emailAction xmlns="http://www.ibm.com/xmlns/wcm/8.0" xmlns:atom="http://www.w3.org/2005/Atom">
            <dateType>GENERAL_DATE_ONE</dateType>
            <offset unit="MONTH" amount="1" direction="AFTER">
                <time-of-day>11:53:00</time-of-day>
            </offset>
            <recipients>
                <stageApprovers>true</stageApprovers>
                <authors of-item="true" of-referencing-items="true"/>
                <owners of-item="true" of-referencing-items="true"/>
                <additional>
                    <distinguishedName>uid=WCMUT_Contributor_A,o=defaultWIMFileBasedRealm</distinguishedName>
                    <atom:uri>/wps/mycontenthandler/um/users/profiles/Z9eAeJHPGJP86M9EAMMG6K9PAMMG62RDGJM4C1JP0JM06IHP4JPK61RDCMIP6K1</atom:uri>
                    <atom:name>WCMUT_Contributor_A WCMUT_Contributor_A</atom:name>
                </additional>
                <additional>
                    <distinguishedName>uid=WCMUT_Contributor_B,o=defaultWIMFileBasedRealm</distinguishedName>
                    <atom:uri>/wps/mycontenthandler/um/users/profiles/Z9eAeKPD83H16JHC2JM47KPOCJMG61HC6MM8C3JCIJMOC4BEAMR06N9E8MQKC63</atom:uri>
                    <atom:name>WCMUT_Contributor_B WCMUT_Contributor_B</atom:name>
                </additional>
            </recipients>
        </wcm:emailAction>
    </content>
</entry>
```

Delete

An email action can be deleted by sending a DELETE request to the following URI: `/EmailAction/action-id`

For example:

```
HTTP/1.1 DELETE /wps/mycontenthandler/wcmrest/ScheduledMoveAction/abc4c24a-3540-4ae3-8ba6-f2f82a97

HTTP/1.1 200 OK
```

How to use REST with custom workflow actions:

You can use the Web Content Manager REST service to create, read, update, and delete custom workflow actions.

Locating Installed Actions

To create a custom workflow action, you need to first identify the installed action that is run by that workflow action. There are two ways to do locate installed actions.

1. List all the installed custom workflow actions by sending a GET request to the following URI:

   Note: Each entry has a link with the "alternate" relation. This link can be used to specify the action to use when you create a custom workflow action.

   `/CustomWorkflowAction/available-actions`

   For example:

```
HTTP/1.1 GET /wps/mycontenthandler/wcmrest/CustomWorkflowAction/available-actions

HTTP 200 OK

<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:CustomWorkflowAction/available-actions</id>
    <title xml:lang="en">All Custom Workflow Actions</title>
    <updated>2014-06-26T01:17:46.022Z</updated>
    <entry>
        <title xml:lang="en">ML Localize</title>
        <summary xml:lang="en">Notifies the base locale owner of changes to localized copies and performs automatic localization of modified base locale documents</summary>
        <wcm:name>com.ibm.workplace.wcm.ml.workflowactions.LocalizeMLCustomWorkflowAction</wcm:name>
        <link rel="alternate" href="/wps/mycontenthandler/wcmrest/CustomWorkflowActionFactory/MLCustomWorkflowActionFactory/actions/com.ibm.workplace.wcm.ml.workflowactions.LocalizeMLCustomWorkflowAction" label="Read"/>
    </entry>
    <entry>
        <title xml:lang="en">ML Regionalize</title>
        <summary xml:lang="en">
           Notifies the base locale owner of changes to regionalized copies and performs
           automatic regionalization of modified base locale documents
        </summary>
        <wcm:name>com.ibm.workplace.wcm.ml.workflowactions.RegionalizeMLCustomWorkflowAction</wcm:name>
        <link rel="alternate" href="/wps/mycontenthandler/wcmrest/CustomWorkflowActionFactory/MLCustomWorkflowActionFactory/actions/com.ibm.workplace.wcm.ml.workflowactions.RegionalizeMLCustomWorkflowAction" label="Read"/>
    </entry>
    <entry>
        <title xml:lang="en">Legacy ML Sync Publish Implementation</title>
        <summary xml:lang="en">
           Ensures that ML versions of the same document are published at the same time.
           This is the legacy implementation of synchronized publishing, you are recommended to use the new Projects-based
           synchronized publishing instead
        </summary>
        <wcm:name>com.ibm.workplace.wcm.ml.workflowactions.SyncPublishMLCustomWorkflowAction</wcm:name>
        <link rel="alternate" href="/wps/mycontenthandler/wcmrest/CustomWorkflowActionFactory/MLCustomWorkflowActionFactory/actions/com.ibm.workplace.wcm.ml.workflowactions.SyncPublishMLCustomWorkflowAction" label="Read"/>
    </entry>
    <entry>
        <title xml:lang="en">ML Sync Expire</title>
        <summary xml:lang="en">
           Ensures that ML versions of the same document are expired at the same time
        </summary>
        <wcm:name>com.ibm.workplace.wcm.ml.workflowactions.SyncExpireMLCustomWorkflowAction</wcm:name>
        <link rel="alternate" href="/wps/mycontenthandler/wcmrest/CustomWorkflowActionFactory/MLCustomWorkflowActionFactory/actions/com.ibm.workplace.wcm.ml.workflowactions.SyncExpireMLCustomWorkflowAction" label="Read"/>
    </entry>
    <entry>
        <title xml:lang="en">ML Sync Delete</title>
        <summary xml:lang="en">Ensures that ML versions of the same document are deleted at the same time</summary>
        <wcm:name>com.ibm.workplace.wcm.ml.workflowactions.SyncDeleteMLCustomWorkflowAction</wcm:name>
        <link rel="alternate" href="/wps/mycontenthandler/wcmrest/CustomWorkflowActionFactory/MLCustomWorkflowActionFactory/actions/com.ibm.workplace.wcm.ml.workflowactions.SyncDeleteMLCustomWorkflowAction" label="Read"/>
    </entry>
    <entry>
        <title xml:lang="en">Update ML Conf File Cache</title>
        <summary xml:lang="en">Update the ML Configuration File cache when a Configuration File is updated</summary>
        <wcm:name>com.ibm.workplace.wcm.ml.workflowactions.UpdateMLConfFileCacheCustomWorkflowAction</wcm:name>
        <link rel="alternate" href="/wps/mycontenthandler/wcmrest/CustomWorkflowActionFactory/MLCustomWorkflowActionFactory/actions/com.ibm.workplace.wcm.ml.workflowactions.UpdateMLConfFileCacheCustomWorkflowAction" label="Read"/>
    </entry>
    <entry>
        <title xml:lang="en">ML Workflow Switcher</title>
        <summary xml:lang="en">Used to switch a document back to its original workflow (as assigned by the ML Workflow engine)</summary>
        <wcm:name>com.ibm.workplace.wcm.ml.workflowactions.WorkflowSwitcherWorkflowAction</wcm:name>
        <link rel="alternate" href="/wps/mycontenthandler/wcmrest/CustomWorkflowActionFactory/MLCustomWorkflowActionFactory/actions/com.ibm.workplace.wcm.ml.workflowactions.WorkflowSwitcherWorkflowAction" label="Read"/>
    </entry>
    <entry>
        <title xml:lang="en">ML Next Stage</title>
        <summary xml:lang="en">Used to move a document to the next stage in the workflow</summary>
        <wcm:name>com.ibm.workplace.wcm.ml.workflowactions.NextStageWorkflowAction</wcm:name>
        <link rel="alternate" href="/wps/mycontenthandler/wcmrest/CustomWorkflowActionFactory/MLCustomWorkflowActionFactory/actions/com.ibm.workplace.wcm.ml.workflowactions.NextStageWorkflowAction" label="Read"/>
    </entry>
</feed>
```

2. List all the installed custom workflow action factories, and in turn the custom actions that are handled by that factory.

   The list of available custom workflow action factories can be obtained by sending a GET request to the following URI. Each entry in the returned feed provides a description of the factory, and a link to all the custom actions associated with that factory that uses the "actions" link relation.

   `/CustomWorkflowActionFactory`

For example:

```
HTTP/1.1 GET /wps/mycontenthandler/wcmrest/CustomWorkflowActionFactory

HTTP 200 OK

<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:CustomWorkflowActionFactory</id>
    <title xml:lang="en">Custom Workflow Action Factories</title>
    <updated>2014-06-26T01:26:38.340Z</updated>
    <entry>
        <id>wcmrest:MLCustomWorkflowActionFactory</id>
        <title xml:lang="en">ML Custom Workflow Action Factory</title>
        <wcm:name>MLCustomWorkflowActionFactory</wcm:name>
        <link rel="actions" href="/wps/mycontenthandler/wcmrest/CustomWorkflowActionFactory/MLCustomWorkflowActionFactory/actions"/>
        <content/>
    </entry>
</feed>
```

The list of actions that are associated with a custom workflow action factory are obtained by sending a GET request to the following URI.

Note: Each entry has a link with the "alternate" relation. This link is used to specify the action when you create a custom workflow action.

`/CustomWorkflowActionFactory/name-of-the-factory/actions`

For example:

```
HTTP/1.1 GET /wps/mycontenthandler/wcmrest/CustomWorkflowActionFactory/MLCustomWorkflowActionFactory/actions

HTTP 200 OK

<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:CustomWorkflowActionFactory/MLCustomWorkflowActionFactory/actions</id>
    <title xml:lang="en">ML Custom Workflow Action Factory</title>
    <updated>2014-06-26T01:30:16.491Z</updated>
    <entry>
        <title xml:lang="en">ML Localize</title>
        <summary xml:lang="en">
           Notifies the base locale owner of changes to localized copies and performs
           automatic localization of modified base locale documents
        </summary>
        <wcm:name>com.ibm.workplace.wcm.ml.workflowactions.LocalizeMLCustomWorkflowAction</wcm:name>
        <link rel="alternate" href="/wps/mycontenthandler/wcmrest/CustomWorkflowActionFactory/MLCustomWorkflowActionFactory/actions/com.ibm.workplace.wcm.ml.workflowactions.LocalizeMLCustomWorkflowAction" label="Read"/>
    </entry>
    ... some entries omitted ...
</feed>
```

Create

Custom workflow actions can be created by sending a POST request to the following URI:

`/CustomWorkflowAction`

For example:

```
HTTP/1.1 POST /wps/mycontenthandler/wcmrest/CustomWorkflowAction

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <title>MLS Next Stage Workflow Action</title>
    <wcm:name>MLS Next Stage Workflow Action</wcm:name>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/digest!muAe8T8GIzS4EJeiF9a_sw/wcmrest/Library/03da7ddc-1bc9-47cd-ab69-b06f23a3f284" />
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:customAction xmlns="http://www.ibm.com/xmlns/wcm/8.0">
            <action href="/wps/mycontenthandler/wcmrest/CustomWorkflowActionFactory/MLCustomWorkflowActionFactory/actions/com.ibm.workplace.wcm.ml.workflowactions.NextStageWorkflowAction"/>
        </wcm:customAction>
    </content>
</entry>

HTTP/1.1 201 Created

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:33b378f9-78be-4457-a5ec-20c9f2848000</id>
    <title xml:lang="en">MLS Next Stage Workflow Action</title>
    <summary xml:lang="en"></summary>
    <wcm:name>MLS Next Stage Workflow Action</wcm:name>
    <wcm:type>CustomWorkflowAction</wcm:type>
    ... some content elided ...
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:customAction xmlns="http://www.ibm.com/xmlns/wcm/8.0" xmlns:atom="http://www.w3.org/2005/Atom">
            <dateType>CUSTOM_ACTION_DATE</dateType>
            <action href="/wps/mycontenthandler/wcmrest/CustomWorkflowActionFactory/MLCustomWorkflowActionFactory/actions/com.ibm.workplace.wcm.ml.workflowactions.NextStageWorkflowAction"/>
        </wcm:customAction>
    </content>
</entry>
```

Update

A custom workflow action can be updated by sending a PUT request to the following URI:

`/CustomWorkflowAction/action-id`

For example:

```
HTTP/1.1 PUT /wps/mycontenthandler/wcmrest/CustomWorkflowAction/abc4c24a-3540-4ae3-8ba6-f2f82a977046
Content-Type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    ... some content elided ...
     <content type="application/vnd.ibm.wcm+xml">
        <wcm:customAction xmlns="http://www.ibm.com/xmlns/wcm/8.0" xmlns:atom="http://www.w3.org/2005/Atom">
            <dateType>SPECIFIED_DATE</dateType>
            <date>2014-06-26T01:48:00.000Z</date>
            <action href="/wps/mycontenthandler/wcmrest/CustomWorkflowActionFactory/MLCustomWorkflowActionFactory/actions/com.ibm.workplace.wcm.ml.workflowactions.NextStageWorkflowAction"/>
        </wcm:customAction>
    </content>
</entry>

HTTP/1.1 200 OK
```

Read

Custom workflow actions can be read by sending a GET request to the following URI:

`/CustomWorkflowAction/action-id`

For example:

```
HTTP/1.1 GET /wps/mycontenthandler/wcmrest/CustomWorkflowAction/33b378f9-78be-4457-a5ec-20c9f2848000
Accept: application/atom+xml

HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:33b378f9-78be-4457-a5ec-20c9f2848000</id>
    <title xml:lang="en">My Test Action</title>
    <summary xml:lang="en"></summary>
    <wcm:name>My Test Action</wcm:name>
    <wcm:type>CustomWorkflowAction</wcm:type>
    <updated>2014-06-26T01:48:54.966Z</updated>
    <wcm:created>2014-06-26T01:48:54.965Z</wcm:created>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAeI1E83S86KPDCMMCC1JP4MMG6G9P6JM8CMPD6MMCCOHOE3Q0713D23S06O1</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAeI1E83S86KPDCMMCC1JP4MMG6G9P6JM8CMPD6MMCCOHOE3Q0713D23S06O1</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAeI1E83S86KPDCMMCC1JP4MMG6G9P6JM8CMPD6MMCCOHOE3Q0713D23S06O1</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
    <wcm:creator>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAeI1E83S86KPDCMMCC1JP4MMG6G9P6JM8CMPD6MMCCOHOE3Q0713D23S06O1</uri>
        <name>wpsadmin</name>
    </wcm:creator>
    <link rel="self" href="/wps/mycontenthandler/wcmrest/CustomWorkflowAction/33b378f9-78be-4457-a5ec-20c9f2848000" xml:lang="en" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/wcmrest/CustomWorkflowAction/33b378f9-78be-4457-a5ec-20c9f2848000" xml:lang="en" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/wcmrest/CustomWorkflowAction/33b378f9-78be-4457-a5ec-20c9f2848000" xml:lang="en" label="Delete"/>
    <link rel="create-draft" href="/wps/mycontenthandler/wcmrest/item/33b378f9-78be-4457-a5ec-20c9f2848000/create-draft" xml:lang="en" label="Create Draft"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/wcmrest/item/33b378f9-78be-4457-a5ec-20c9f2848000/change-to-draft" xml:lang="en" label="Change To Draft"/>
    <link rel="access-control" href="/wps/mycontenthandler/ac/access:oid:Z6QReDeJPC4MPS6OHPIJMS6OHOAMMG6K9DEJM4CL9P6MM86GPOI3J96O1DG3O06G1" xml:lang="en" label="Access Control"/>
    <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/f68ddb0c-c06b-43a9-84fd-d43552980e46" xml:lang="en" label="Library"/>
    <link rel="versions" href="/wps/mycontenthandler/wcmrest/item/33b378f9-78be-4457-a5ec-20c9f2848000/versions" xml:lang="en" label="Versions"/>
    <link rel="edit-media" href="/wps/mycontenthandler/wcmrest/CustomWorkflowAction/33b378f9-78be-4457-a5ec-20c9f2848000" type="application/vnd.ibm.wcm+xml" xml:lang="en" label="Edit Media"/>
    <category scheme="wcmrest:workflowState" term="PUBLISHED" label="Published" xml:lang="en"/>
    <category scheme="wcmrest:favorite" term="false" xml:lang="en"/>
    <content type="application/vnd.ibm.wcm+xml">
        <wcm:customAction xmlns="http://www.ibm.com/xmlns/wcm/8.0" xmlns:atom="http://www.w3.org/2005/Atom">
            <dateType>CUSTOM_ACTION_DATE</dateType>
            <action href="/wps/mycontenthandler/wcmrest/CustomWorkflowActionFactory/MLCustomWorkflowActionFactory/actions/com.ibm.workplace.wcm.ml.workflowactions.NextStageWorkflowAction"/>
        </wcm:customAction>
    </content>
</entry>
```

Delete

A custom action can be deleted by sending a DELETE request to the following URI:

`/CustomWorkflowAction/action-id`

For example:

```
HTTP/1.1 DELETE /wps/mycontenthandler/wcmrest/CustomWorkflowAction/abc4c24a-3540-4ae3-8ba6-f2f82a977(

HTTP/1.1 200 OK
```

How to use REST with projects

You can use the REST services for Web Content Manager to create and work with projects.

Creating a project

To create a project, send a POST request that contains the appropriate data to the following URI.

`/project/`

For example:

```
POST /wps/mycontenthandler/wcmrest/Project HTTP/1.0
Content-type : application/atom+xml

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <title>SampleProjectTitle</title>
    <wcm:name>SampleProjectName</wcm:name>
    <wcm:description>SampleProjectDescription</wcm:description>
</entry>


HTTP/1.0 Created
Content-type: application/atom+xml; type=entry
Content-location: /wps/mycontenthandler/!ut/p/wcmrest/Project/80d503aa-fec5-477c-a8b2-372897982af

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <id>80d503aa-fec5-477c-a8b2-372897982afe</id>
    <title>SampleProjectTitle</title>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/Project/80d503aa-fec5-477c-a8b2-372897982afe"/>
    <link rel="add-item" href="/wps/mycontenthandler/!ut/p/wcmrest/Project/80d503aa-fec5-477c-a8b2-372897982afe/additem"/>
    <link rel="remove-item" href="/wps/mycontenthandler/!ut/p/wcmrest/Project/80d503aa-fec5-477c-a8b2-372897982afe/removeitem"/>
    <updated>2011-05-30T06:15:29.952Z</updated>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5Ul75Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5Ul75Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:name>SampleProjectName</wcm:name>
    <wcm:description>SampleProjectDescription</wcm:description>
    <wcm:type>Project</wcm:type>
</entry>
```

Reading a project

To read a project, send a GET request that contains the appropriate data to the following URI.

`/project/item-uuid`

For example:

```
GET /wps/mycontenthandler/wcmrest/Project/80d503aa-fec5-477c-a8b2-372897982afe HTTP/1.1


HTTP/1.0 200 OK
Content-type: application/atom+xml; type=entry

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <id>80d503aa-fec5-477c-a8b2-372897982afe</id>
    <title>SampleProjectTitle</title>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/Project/80d503aa-fec5-477c-a8b2-372897982afe"/>
    <link rel="add-item" href="/wps/mycontenthandler/!ut/p/wcmrest/Project/80d503aa-fec5-477c-a8b2-372897982afe/additem"/>
    <link rel="remove-item" href="/wps/mycontenthandler/!ut/p/wcmrest/Project/80d503aa-fec5-477c-a8b2-372897982afe/removeitem"/>
    <updated>2011-05-30T06:15:29.952Z</updated>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5Ul75Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <email></email>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5Ul75Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <email></email>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:name>SampleProjectName</wcm:name>
    <wcm:description>SampleProjectDescription</wcm:description>
    <wcm:type>Project</wcm:type>
</entry>
```

Deleting a project

To delete a project, send a DELETE request that contains the appropriate data to the following URI.

`/project/item-uuid`

For example:

```
HTTP/1.1 DELETE
http://host:port/wps/mycontenthandler/wcmrest/Project/35b9120a-17d0-4dcb-b0ba-b034e34b50a6
Accept-Type: application/atom+xml

200 OK
```

Updating a project

To update a project, send a PUT request that contains the appropriate data to the following URI.

```
/project/item-uuid
```

For example:

```
HTTP/1.1 PUT
http://host:port/wps/mycontenthandler/wcmrest/Project/35b9120a-17d0-4dcb-b0ba-b034e34b50a6
Accept-Type: application/atom+xml

200 OK
```

Adding an item to a project

To add an item to a project, update the item by using a PUT request that specifies a link with relation "project" specifying the project to add the item to. For example:

```
HTTP/1.1 PUT
http://host:port/wps/mycontenthandler/!ut/p/digest!MetYLHV_M5sJbvslxI8twA/
wcmrest/LibraryTextComponent/fd34a8bf-7ca1-499c-80ab-acdc2f33cf3e</a>
Accept-Type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm">
    ... data ...
<link rel="project" href="/wps/mycontenthandler/wcmrest/Project/77d08cf6-88f6-4577-a929-34d43a8e150e
... data...
</entry>

200 OK
```

Note: If you use REST inside a Portal project context, posting to the create-draft link relation creates the draft in a project.

Removing an item from a project

To remove an item from a project, update the item by using a PUT request without specifying the project link relation. For example:

```
HTTP/1.1 PUT
http://host:port/wps/mycontenthandler/!ut/p/digest!MetYLHV_M5sJbvslxI8twA/
wcmrest/LibraryTextComponent/fd34a8bf-7ca1-499c-80ab-acdc2f33cf3e</a>
Accept-Type: application/atom+xml

200 OK
```

Submitting a project for review

Editors of a project can submit a project for review by sending a POST request that contains the appropriate data to the following URI:

```
/project/project-uuid/submit-for-review
```

For example:

```
HTTP/1.1 POST
http://host:port/wps/mycontenthandler/wcmrest/Project/PROJECT-UUID/submit-for-review
Accept-Type: application/atom+xml

201 Created
```

Withdrawing a project from review

Editors of a project can withdraw a project from review by sending a POST request that contains the appropriate data to the following URI:

```
/project/project-uuid/withdraw-from-review
```

For example:

```
HTTP/1.1 POST
http://host:port/wps/mycontenthandler/wcmrest/Project/PROJECT-UUID/withdraw-from-review
Accept-Type: application/atom+xml

201 Created
```

When withdrawn, the project returns to an active state.

Approving a project

Approvers of a project can approve a project by sending a POST request that contains the appropriate data to the following URI:

```
/project/project-uuid/approve
```

For example:

```
HTTP/1.1 POST
http://host:port/wps/mycontenthandler/wcmrest/Project/PROJECT-UUID/approve
Accept-Type: application/atom+xml

201 Created
```

When approved, the project is ready to be published.

Rejecting a project

Approvers of a project can reject project approval by sending a POST request that contains the appropriate data to the following URI:

```
/project/project-uuid/reject
```

For example:

```
HTTP/1.1 POST
http://host:port/wps/mycontenthandler/wcmrest/Project/PROJECT-UUID/reject
Accept-Type: application/atom+xml

201 Created
```

When rejected, the project returns to an active state.

Withdrawing approval for a project

Approvers of a project can withdraw approval for a project by sending a POST request that contains the appropriate data to the following URI:

```
/project/project-uuid/withdraw-approval
```

For example:

```
HTTP/1.1 POST
http://host:port/wps/mycontenthandler/wcmrest/Project/PROJECT-UUID/withdraw-approval
Accept-Type: application/atom+xml

201 Created
```

Withdrawing approval does not change the state of the project, which remains in review.

How to use REST with folders

You can use the Web Content Manager REST service to create, read, update, and delete folders. You can also use the WWeb Content Manager REST service to query for the preset folders in a library.

Query For Preset Folders

A list of the preset folders in a library can be obtained by sending a GET request to the following URI:

```
/Library/library-id/preset-folders
```

For example:

```
HTTP/1.1 /wps/mycontenthandler/wcmrest/Library/
Accept: application/atom+xml

HTTP/1.1 200 OK
Content-Type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:Library/790b8ca6-5d89-4f37-9052-783dd580f860/preset-folders</id>
    <title>wcmrest:Library/790b8ca6-5d89-4f37-9052-783dd580f860/preset-folders</title>
    <updated>2014-06-27T02:30:24.312Z</updated>
    <entry>
        <id>wcmrest:fd169c35-f35b-45da-b5cb-6dab73330142</id>
        <title xml:lang="en">Content</title>
        <wcm:name>Content</wcm:name>
        <wcm:type>PresetFolder</wcm:type>
        <updated>2014-06-13T02:27:49.631Z</updated>
        <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/790b8ca6-5d89-4f37-9052-783dd580f860" label="Library"/>
        <link rel="alternate" href="/wps/mycontenthandler/wcmrest/Folder/fd169c35-f35b-45da-b5cb-6dab73330142" label="Read"/>
    </entry>
    <entry>
        <id>wcmrest:f5f8964b-7e47-4427-aca6-46982125f123</id>
        <title xml:lang="en">Taxonomies</title>
        <wcm:name>Taxonomies</wcm:name>
        <wcm:type>PresetFolder</wcm:type>
        <updated>2014-06-13T02:27:49.629Z</updated>
        <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/790b8ca6-5d89-4f37-9052-783dd580f860" label="Library"/>
        <link rel="alternate" href="/wps/mycontenthandler/wcmrest/Folder/f5f8964b-7e47-4427-aca6-46982125f123" label="Read"/>
    </entry>
    <entry>
        <id>wcmrest:b7fd466d-d430-4a72-bb33-e8733f9b47e8</id>
        <title xml:lang="en">Components</title>
        <wcm:name>Components</wcm:name>
        <wcm:type>PresetFolder</wcm:type>
        <updated>2014-06-13T02:27:49.631Z</updated>
        <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/790b8ca6-5d89-4f37-9052-783dd580f860" label="Library"/>
        <link rel="alternate" href="/wps/mycontenthandler/wcmrest/Folder/b7fd466d-d430-4a72-bb33-e8733f9b47e8" label="Read"/>
    </entry>
    ... some entries elided ...
</feed>
```

Create

A folder can be created by sending a POST request to the following URI with an Atom entry that represents the folder:

```
/Folder
```

For example:

```
HTTP/1.1 POST /wps/mycontenthandler/wcmrest/Folder
Content-Type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <wcm:name>Design Components</wcm:name>
    <link rel="parent" href="/wps/mycontenthandler/wcmrest/Folder/17fe1ab3-ba6a-4769-b5f0-a2cb2f91ebb5-10"/>
</entry>

HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:5def4f6f-0dd9-49ca-b954-706f8ceb7373</id>
    <title xml:lang="en">test create components folder</title>
    <summary xml:lang="en"></summary>
    <wcm:name>test create components folder</wcm:name>
    <wcm:type>Folder</wcm:type>
    <updated>2014-06-27T02:39:58.735Z</updated>
    <wcm:created>2014-06-27T02:39:58.733Z</wcm:created>
```

```
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
    <wcm:creator>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:creator>
    <link rel="self" href="/wps/mycontenthandler/wcmrest/Folder/5def4f6f-0dd9-49ca-b954-706f8ceb7373" xml:lang="en" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/wcmrest/Folder/5def4f6f-0dd9-49ca-b954-706f8ceb7373" xml:lang="en" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/wcmrest/Folder/5def4f6f-0dd9-49ca-b954-706f8ceb7373" xml:lang="en" label="Delete"/>
    <link rel="access-control" href="/wps/mycontenthandler/ac/access:oid:Z6QReDeL1PA6JH66JDCMM0643PIJMG6PPO2MM8CP9D8JMS6GHDC6SCC5JOEJPS6J1" xml:lang="en" label="Access Control"/>
    <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/17fe1ab3-ba6a-4769-b5f0-a2cb2f91ebb5" xml:lang="en" label="Library"/>
    <link rel="parent" href="/wps/mycontenthandler/wcmrest/Folder/17fe1ab3-ba6a-4769-b5f0-a2cb2f91ebb5-10" xml:lang="en" label="Parent"/>
    <link rel="versions" href="/wps/mycontenthandler/wcmrest/item/5def4f6f-0dd9-49ca-b954-706f8ceb7373/versions" xml:lang="en" label="Versions"/>
    <category scheme="wcmrest:favorite" term="false" xml:lang="en"/>
</entry>
```

Update

A folder can be updated by sending a PUT request to the following URI with an Atom entry that includes the fields on the item that need to be changed:

`/Folder/folder-id`

For example:

```
HTTP/1.1 PUT /wps/mycontenthandler/wcmrest/Folder/5def4f6f-0dd9-49ca-b954-706f8ceb7373
Content-Type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    ... some content elided ...
    <wcm:name>The Folder Name has Chenged</wcm:name>
    <!-- Note that the parent has changed. This will cause the folder to be moved. -->
    <link rel="parent" href="/wps/mycontenthandler/wcmrest/Folder/aceaf85a-48a8-45f7-a6cc-343ace84f337" xml:lang="en" label="Parent"/>
    ... some content elided ...
</entry>
```

Read

A folder can be read by sending a GET request to the following URI:

`/Folder/folder-id`

For example:

```
HTTP/1.1 GET /wps/mycontenthandler/wcmrest/Folder/5def4f6f-0dd9-49ca-b954-706f8ceb7373
Accept: application/atom+xml

HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:5def4f6f-0dd9-49ca-b954-706f8ceb7373</id>
    <title xml:lang="en">test create components folder</title>
    <summary xml:lang="en"></summary>
    <wcm:name>test create components folder</wcm:name>
    <wcm:type>Folder</wcm:type>
    <updated>2014-06-27T02:39:58.735Z</updated>
    <wcm:created>2014-06-27T02:39:58.733Z</wcm:created>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
    <wcm:creator>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:creator>
    <link rel="self" href="/wps/mycontenthandler/wcmrest/Folder/5def4f6f-0dd9-49ca-b954-706f8ceb7373" xml:lang="en" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/wcmrest/Folder/5def4f6f-0dd9-49ca-b954-706f8ceb7373" xml:lang="en" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/wcmrest/Folder/5def4f6f-0dd9-49ca-b954-706f8ceb7373" xml:lang="en" label="Delete"/>
    <link rel="access-control" href="/wps/mycontenthandler/ac/access:oid:Z6QReDeL1PA6JH66JDCMM0643PIJMG6PPO2MM8CP9D8JMS6GHDC6SCC5JOEJPS6J1" xml:lang="en" label="Access Control"/>
    <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/17fe1ab3-ba6a-4769-b5f0-a2cb2f91ebb5" xml:lang="en" label="Library"/>
    <link rel="parent" href="/wps/mycontenthandler/wcmrest/Folder/17fe1ab3-ba6a-4769-b5f0-a2cb2f91ebb5-10" xml:lang="en" label="Parent"/>
    <link rel="versions" href="/wps/mycontenthandler/wcmrest/item/5def4f6f-0dd9-49ca-b954-706f8ceb7373/versions" xml:lang="en" label="Versions"/>
    <category scheme="wcmrest:favorite" term="false" xml:lang="en"/>
</entry>
```

Delete

A folder can be deleted by sending a DELETE request to the following URI:

`/Folder/folder-id`

```
HTTP/1.1 DELETE /wps/mycontenthandler/wcmrest/Folder/5def4f6f-0dd9-49ca-b954-706f8ceb7373

HTTP/1.1 200 OK
```

How to use REST with taxonomies

You can use the Web Content Manager REST service to create, read, update, and delete taxonomies.

Create

A taxonomy can be created by sending a POST request to the following URI with an Atom entry that represents the item:
`/Taxonomy`

For example:

```
HTTP/1.1 POST /wps/mycontenthandler/wcmrest/Taxonomy
Content-Type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <wcm:name>Marketing Taxonomy</wcm:name>
    <title>Marketing Taxonomy</title>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/digest!muAe8T8GIzS4EJeiF9a_sw/wcmrest/Library/94a8214d-322c-4724-acc3-b6c217d1bc5f" label="Library"/>
</entry>

HTTP/1.1 201 Created
Content-Type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:501a4ab8-48c3-41fc-b1fd-6dbe0acba1d4</id>
    <wcm:name>Marketing Taxonomy</wcm:name>
    <title>Marketing Taxonomy</title>
    <wcm:type>Taxonomy</wcm:type>
    <updated>2014-06-27T00:52:59.022Z</updated>
    <wcm:created>2014-06-27T00:52:59.022Z</wcm:created>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
    <wcm:creator>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:creator>
    <link rel="self" href="/wps/mycontenthandler/wcmrest/Taxonomy/501a4ab8-48c3-41fc-b1fd-6dbe0acba1d4" xml:lang="en" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/wcmrest/Taxonomy/501a4ab8-48c3-41fc-b1fd-6dbe0acba1d4" xml:lang="en" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/wcmrest/Taxonomy/501a4ab8-48c3-41fc-b1fd-6dbe0acba1d4" xml:lang="en" label="Delete"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/wcmrest/item/501a4ab8-48c3-41fc-b1fd-6dbe0acba1d4/change-to-draft" xml:lang="en" label="Change To Draft"/>
    <link rel="create-draft" href="/wps/mycontenthandler/wcmrest/item/501a4ab8-48c3-41fc-b1fd-6dbe0acba1d4/create-draft" xml:lang="en" label="Create Draft"/>
    <link rel="access-control" href="/wps/mycontenthandler/ac/access:oid:Z6QReDeL1C2JGH61JOGJMG6OPO6JMG6HHP6MM8CHHP8MMO64JOA6O4C3JO2MOGCK1" xml:lang="en" label="Access Control"/>
    <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/94a8214d-322c-4724-acc3-b6c217d1bc5f" xml:lang="en" label="Library"/>
    <link rel="versions" href="/wps/mycontenthandler/wcmrest/item/501a4ab8-48c3-41fc-b1fd-6dbe0acba1d4/versions" xml:lang="en" label="Versions"/>
    <category scheme="wcmrest:workflowState" term="PUBLISHED" label="Published" xml:lang="en"/>
    <category scheme="wcmrest:favorite" term="false" xml:lang="en"/>
</entry>
```

Update

A taxonomy can be updated by sending a PUT request to the following URI:
`/Taxonomy/taxonomy-id`

For example:

```
HTTP/1.1 PUT /wps/mycontenthandler/wcmrest/Taxonomy/501a4ab8-48c3-41fc-b1fd-6dbe0acba1d4
Content-Type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <wcm:name>Marketing Taxonomy Has Been Changed</wcm:name>
    ... some content elided ...
</entry>

HTTP/1.1 200 OK
```

Read

A taxonomy can be read by sending a GET request to the following URI:
`/Taxonomy/taxonomy-id`

For example:

```
HTTP/1.1 GET /wps/mycontenthandler/wcmrest/Taxonomy/501a4ab8-48c3-41fc-b1fd-6dbe0acba1d4
Accept: application/atom+xml

HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:501a4ab8-48c3-41fc-b1fd-6dbe0acba1d4</id>
    <wcm:name>Marketing Taxonomy</wcm:name>
    <title>Marketing Taxonomy</title>
    <wcm:type>Taxonomy</wcm:type>
    <updated>2014-06-27T00:52:59.022Z</updated>
    <wcm:created>2014-06-27T00:52:59.022Z</wcm:created>
```

```
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
    <wcm:creator>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:creator>
    <link rel="self" href="/wps/mycontenthandler/wcmrest/Taxonomy/501a4ab8-48c3-41fc-b1fd-6dbe0acba1d4" xml:lang="en" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/wcmrest/Taxonomy/501a4ab8-48c3-41fc-b1fd-6dbe0acba1d4" xml:lang="en" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/wcmrest/Taxonomy/501a4ab8-48c3-41fc-b1fd-6dbe0acba1d4" xml:lang="en" label="Delete"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/wcmrest/item/501a4ab8-48c3-41fc-b1fd-6dbe0acba1d4/change-to-draft" xml:lang="en" label="Change To Draft"/>
    <link rel="create-draft" href="/wps/mycontenthandler/wcmrest/item/501a4ab8-48c3-41fc-b1fd-6dbe0acba1d4/create-draft" xml:lang="en" label="Create Draft"/>
    <link rel="access-control" href="/wps/mycontenthandler/ac/access:oid:Z6QReDeL1C2JGH61JOGJMG6OPO6JMG6HHP6MM8CHHP8MMO64JOA6O4C3JO2MOGCK1" xml:lang="en" label="Access Control"/>
    <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/94a8214d-322c-4724-acc3-b6c217d1bc5f" xml:lang="en" label="Library"/>
    <link rel="versions" href="/wps/mycontenthandler/wcmrest/item/501a4ab8-48c3-41fc-b1fd-6dbe0acba1d4/versions" xml:lang="en" label="Versions"/>
    <category scheme="wcmrest:workflowState" term="PUBLISHED" label="Published" xml:lang="en"/>
    <category scheme="wcmrest:favorite" term="false" xml:lang="en"/>
</entry>
```

Delete

A taxonomy can be deleted by sending a DELETE request to the following URI:

`/Taxonomy/taxonomy-id`

In this example, the type `PublishAction` can be replaced with `ExpireAction` or `VersionAction` when you work with those workflow action types.

```
HTTP/1.1 DELETE /wps/mycontenthandler/wcmrest/Taxonomy/7b40f5b6-bf52-4b9e-8062-b0755aaf1f80

HTTP/1.1 200 OK
```

How to use REST with categories

You can use the Web Content Manager REST service to create, read, update, and delete categories.

Create

A category can be created by sending a POST request to the following URI with an Atom entry that represents the item:

`/Category`

For example:

```
HTTP/1.1 POST /wps/mycontenthandler/wcmrest/Category
Content-Type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <wcm:name>Product Marketing Category</wcm:name>
    <link rel="parent" href="/wps/mycontenthandler/wcmrest/Taxonomy/5b4c58a2-68fb-45e2-9907-ef58ad95cc2f"/>
</entry>

HTTP/1.1 201 Created
Content-Type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:525900ec-0ae0-40f7-9edd-4bddc7af154f</id>
    <title xml:lang="en">Product Marketing Category</title>
    <summary xml:lang="en"></summary>
    <wcm:name>Product Marketing Category</wcm:name>
    <wcm:type>Category</wcm:type>
    <updated>2014-06-27T01:05:28.212Z</updated>
    <wcm:created>2014-06-27T01:05:28.212Z</wcm:created>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
    <wcm:creator>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:creator>
    <link rel="self" href="/wps/mycontenthandler/wcmrest/Category/525900ec-0ae0-40f7-9edd-4bddc7af154f" xml:lang="en" label="Read"/>
```

```
    <link rel="edit" href="/wps/mycontenthandler/wcmrest/Category/525900ec-0ae0-40f7-9edd-4bddc7af154f" xml:lang="en" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/wcmrest/Category/525900ec-0ae0-40f7-9edd-4bddc7af154f" xml:lang="en" label="Delete"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/wcmrest/item/525900ec-0ae0-40f7-9edd-4bddc7af154f/change-to-draft" xml:lang="en" label="Change To Draft"/>
    <link rel="create-draft" href="/wps/mycontenthandler/wcmrest/item/525900ec-0ae0-40f7-9edd-4bddc7af154f/create-draft" xml:lang="en" label="Create Draft"/>
    <link rel="access-control" href="/wps/mycontenthandler/ac/access:oid:Z6QReDeLHCAJS06G9P6MM061BP0JMG6GHPEJM4753P8MMG623P8MHT61JP2JQG663" xml:lang="en" label="Access Control"/>
    <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/078dc9ce-3cde-4eec-ab7c-2bdc0043deb7" xml:lang="en" label="Library"/>
    <link rel="parent" href="/wps/mycontenthandler/wcmrest/Taxonomy/5b4c58a2-68fb-45e2-9907-ef58ad95cc2f" xml:lang="en" label="Parent"/>
    <link rel="versions" href="/wps/mycontenthandler/wcmrest/item/525900ec-0ae0-40f7-9edd-4bddc7af154f/versions" xml:lang="en" label="Versions"/>
    <category scheme="wcmrest:workflowState" term="PUBLISHED" label="Published" xml:lang="en"/>
    <category scheme="wcmrest:favorite" term="false" xml:lang="en"/>
</entry>
```

Update

A category can be updated by sending a PUT request to the following URI `/Category/category-id`

For example:

```
HTTP/1.1 PUT /wps/mycontenthandler/wcmrest/Category/525900ec-0ae0-40f7-9edd-4bddc7af154f
Content-Type: application/atom+xml

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <wcm:name>Marketing Category Has Been Changed</wcm:name>
    ... some content elided ...
</entry>

HTTP/1.1 200 OK
```

Read

A category can be read by sending a GET request to the following URI: `/Category/category-id`

For example:

```
HTTP/1.1 GET /wps/mycontenthandler/wcmrest/Category/501a4ab8-48c3-41fc-b1fd-6dbe0acba1d4
Accept: application/atom+xml

HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm/8.0">
    <id>wcmrest:525900ec-0ae0-40f7-9edd-4bddc7af154f</id>
    <title xml:lang="en">Product Marketing Category</title>
    <summary xml:lang="en"></summary>
    <wcm:name>Product Marketing Category</wcm:name>
    <wcm:type>Category</wcm:type>
    <updated>2014-06-27T01:05:28.212Z</updated>
    <wcm:created>2014-06-27T01:05:28.212Z</wcm:created>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:lastModifier>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:lastModifier>
    <wcm:creator>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/um/users/profiles/Z9eAe5JOE3O46N1P0JM06J9OCJMG64BC6MM472RCCJMK62JD66J57OHDC3OC6N1</uri>
        <name>wpsadmin</name>
    </wcm:creator>
    <link rel="self" href="/wps/mycontenthandler/wcmrest/Category/525900ec-0ae0-40f7-9edd-4bddc7af154f" xml:lang="en" label="Read"/>
    <link rel="edit" href="/wps/mycontenthandler/wcmrest/Category/525900ec-0ae0-40f7-9edd-4bddc7af154f" xml:lang="en" label="Edit"/>
    <link rel="delete" href="/wps/mycontenthandler/wcmrest/Category/525900ec-0ae0-40f7-9edd-4bddc7af154f" xml:lang="en" label="Delete"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/wcmrest/item/525900ec-0ae0-40f7-9edd-4bddc7af154f/change-to-draft" xml:lang="en" label="Change To Draft"/>
    <link rel="create-draft" href="/wps/mycontenthandler/wcmrest/item/525900ec-0ae0-40f7-9edd-4bddc7af154f/create-draft" xml:lang="en" label="Create Draft"/>
    <link rel="access-control" href="/wps/mycontenthandler/ac/access:oid:Z6QReDeLHCAJS06G9P6MM061BP0JMG6GHPEJM4753P8MMG623P8MHT61JP2JQG663" xml:lang="en" label="Access Control"/>
    <link rel="library" href="/wps/mycontenthandler/wcmrest/Library/078dc9ce-3cde-4eec-ab7c-2bdc0043deb7" xml:lang="en" label="Library"/>
    <link rel="parent" href="/wps/mycontenthandler/wcmrest/Taxonomy/5b4c58a2-68fb-45e2-9907-ef58ad95cc2f" xml:lang="en" label="Parent"/>
    <link rel="versions" href="/wps/mycontenthandler/wcmrest/item/525900ec-0ae0-40f7-9edd-4bddc7af154f/versions" xml:lang="en" label="Versions"/>
    <category scheme="wcmrest:workflowState" term="PUBLISHED" label="Published" xml:lang="en"/>
    <category scheme="wcmrest:favorite" term="false" xml:lang="en"/>
</entry>
```

Delete

A category can be deleted by sending a DELETE request to the following URI: `/Category/category-id`

In this example, the type `PublishAction` can be replaced with `ExpireAction` or `VersionAction` when working with those workflow action types.

```
HTTP/1.1 DELETE /wps/mycontenthandler/wcmrest/Category/525900ec-0ae0-40f7-9edd-4bddc7af154f

HTTP/1.1 200 OK
```

How to use REST to work with item identity controls

The way identity controls are used with REST depend on whether a text provider is enabled or not for an item.

Creating, updating, or reading identity controls with no text provider enabled

When no text provider is enabled on an item, identity controls work this way:

Table 488. REST tags with no text provider enabled

REST Tag	Item field
`<wcm:name>text</wcm:name>`	Name
`<atom:title>text</atom:title>`	Display title
`<atom:summary>text</atom:summary>`	Description

Enabling updating or reading text provider controls

To either enable, update or read text provider controls, you add the following parameters to the title tag:

Table 489. REST tags for text providers

REST Tag	Text provider field
`<wcm:title TextProviderName>text</ wcm:title TextProviderName>`	Title name
`<wcm:title TextProviderKey>text</ wcm:title TextProviderKey>`	Title key
`<wcm:desc TextProviderName>text</ wcm:desc TextProviderName>`	Description name
`<wcm:desc TextProviderKey>text</wcm:desc TextProviderKey>`	Description key

Creating, updating, or reading identity controls with a text provider enabled

When a text provider is enabled on an item, identity controls work this way:

Table 490. REST tags with a text provider enabled

REST Tag	Item field
`<wcm:name>text</wcm:name>`	Name
`<wcm:displayTitle>text</ wcm:displayTitle>`	Display title
`<atom:title lang="language code" >text</atom:title>` For example:`<atom:title lang="fr" >text</atom:title>`	Localized title **Note:** When a text provider is enabled, only the localized title can be read by the REST service. You need to update the text provider plug-in directly to change the localized title.
`<wcm:description>text</wcm:description>`	Description

Table 490. REST tags with a text provider enabled (continued)

REST Tag	Item field
<atom:summary lang="language code" >text</atom:summary> For example:<atom:summary lang="fr" >text</atom:summary>	Localized description **Note:** When a text provider is enabled, only the localized description can be read by the REST service. You need to update the text provider plug-in directly to change the localized description.

Note: The language code that is required for the localized tags must be an IETF BCP47 compliant language code.

How to use REST with access controls

The access control section in an Atom entry or JSON entry that is generated by the REST service for Web Content Manager links to resources from the Portal Access Control REST API.

An example of how the access controls are displayed:

```
<link rel="access-control"
href="/wps/mycontenthandler/!ut/p/digest!ZXqvndUckPlBBgMoHkgcCA/ac/
access:oid:Z6QReDeOHP0JQK62BO4JM462RO6JMG64JC2JM07K1D6JM07MHCE3QKCH1D86I9633"/>
```

You can send a GET request to the URI and obtain the access control information of an item that is generated by the "Portal Access Control REST API" on page 2923.

How to use REST to work with author and owner parameters

Information for the author and owner of an item can be specified by using the REST service.

The following attributes are used:

- `name`
- `distinguishedName`
- A URI to fetch detailed information of the author or owner by using the PUMA REST SPI

You can also specify such information in the entry in the same format for a POST or PUT operation to Create or Update an item with the information you specified.

Example

This example is in application/atom+xml format:

```
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
   ...
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5Ul75Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5Ul75Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <name>wpsadmin</name>
    </wcm:owner>
   ...
</entry>
```

How to use REST with versions

Item versions can be listed and read by using the REST service.

Retrieving a list of versions

To retrieve a list of versions, you use a GET request to the following URI:

/item/*item-uuid*/versions

A feed is returned containing the identifying information of each version, along with a relation "versioned-item" that links to the specified version.

Version link relations

```
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
  ...
    <link rel="versions" href="/wps/mycontenthandler/!ut/p/wcmrest/item/0d678334-69ae-4d3a-a525-91bb551e5a18/versions"/>
  ...
</entry>
```

Versions feed

```
<feed xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <entry>
        <link rel="versioned-item" href="/wps/mycontenthandler/!ut/p/wcmrest/item/0d678334-69ae-4d3a-a525-91bb551e5a18/version/2"/>
        <content>
            <wcm:versionContent>
                <wcm:versionName>2</wcm:versionName>
                <wcm:versionDate>2011-05-30T04:38:49.540Z</wcm:versionDate>
            </wcm:versionContent>
        </content>
    </entry>
    <entry>
        <link rel="versioned-item" href="/wps/mycontenthandler/!ut/p/wcmrest/item/0d678334-69ae-4d3a-a525-91bb551e5a18/version/1"/>
        <content>
            <wcm:versionContent>
                <wcm:versionName>1</wcm:versionName>
                <wcm:versionDate>2011-05-30T04:33:40.677Z</wcm:versionDate>
            </wcm:versionContent>
        </content>
    </entry>
</feed>
```

Viewing the details of a version

To view the details of a specified version, you use a GET request in the following format:

/item/*item-uuid*/version/*version-name*

For example:

```
GET HTTP/1.0
/wps/mycontenthandler/wcmrest/item/8f055ba2-1bc3-4d21-8443-86274e14dd2c/version/1
Host: www.example.com
Accept: application/atom+xml


HTTP/1.0 200 OK

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <id>0d678334-69ae-4d3a-a525-91bb551e5a181</id>
    <title>SampleNumericComponentTitle</title>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/LibraryNumericComponent/0d678334-69ae-4d3a-a525-91bb551e5a181"/>
    <link rel="edit-media" type="text/plain" href="/wps/mycontenthandler/!ut/p/wcmrest/LibraryNumericComponent/0d678334-69ae-4d3a-a525-91bb551e5a181"/>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/item/c98d11e1-7f2a-480e-9aac-40eb1949cbda"/>
    <link rel="create-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/0d678334-69ae-4d3a-a525-91bb551e5a181/create-draft"/>
    <link rel="change-to-draft" href="/wps/mycontenthandler/!ut/p/wcmrest/item/0d678334-69ae-4d3a-a525-91bb551e5a181/change-to-draft"/>
    <link rel="versions" href="/wps/mycontenthandler/!ut/p/wcmrest/item/0d678334-69ae-4d3a-a525-91bb551e5a181/versions"/>
    <updated>2011-05-30T06:42:10.244Z</updated>
    <author>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5Ul75Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <email></email>
        <name>wpsadmin</name>
    </author>
    <wcm:owner>
        <wcm:distinguishedName>uid=wpsadmin,o=defaultWIMFileBasedRealm</wcm:distinguishedName>
        <uri>/wps/mycontenthandler/!ut/p/digest!6GVkh5Ul75Ln7DdEgvHm_g/um/users/profiles/Z9eAeH1C2JG561RC6JM47H9E4MMG6PHO6JM4C5JD0JMOC6BEEJS464JDG3I56K1</uri>
        <email></email>
        <name>wpsadmin</name>
    </wcm:owner>
    <wcm:name>SampleNumericComponentNameUpdated</wcm:name>
    <wcm:description>SampleNumericComponentDescription</wcm:description>
    <wcm:type>LibraryNumericComponent</wcm:type>
    <wcm:state>PUBLISHED</wcm:state>
    <versionContent>
        <wcm:versionName>1</wcm:versionName>
        <wcm:versionDate>2011-05-30T04:33:40.677Z</wcm:versionDate>
    </versionContent>
</entry>
```

Using REST to work with recent items

You can use REST service to display a list of recently accessed items. This is the equivalent of the **Recent Items** view in the library explorer.

Note: By default, items accessed through the REST service are added to the recent items list. To stop items accessed through the REST service appearing in the list of recent items, change the *rest.default.add-recent* parameter to false in the `WCM WCMConfigService` service using the WebSphere Integrated Solutions Console. The *rest.default.add-recent* parameter can be overridden on a per-request basis by specifying `recent=true` or `recent=false` in the query string.

URI: `/recent-items`

Example:

```
HTTP 1.1 GET /wps/mycontenthandler/wcmrest/recent-items

<feed xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm">
    <id>wcmrest:recent-items</id>
    <title>wcmrest:recent-items</title>
    <updated>2011-09-28T02:51:47.228Z</updated>
    <link rel="next-page" href="/wps/mycontenthandler/!ut/p/wcmrest/recent-items?page=2"/>
    <entry>
        <id>wcmrest:8b629b12-e16a-4afa-bbf5-a37ebee5a5b8</id>
        <title xml:lang="en">Articles List</title>
        <summary xml:lang="en"></summary>
        <wcm:name>Articles List</wcm:name>
        <wcm:description xml:lang="en"></wcm:description>
        <wcm:type>LibraryMenuComponent</wcm:type>
        <updated>2011-09-21T06:21:11.701Z</updated>
        <wcm:lastAccessed>2011-09-26T05:41:50.874Z</wcm:lastAccessed>
        <wcm:lastModifier>
            <wcm:distinguishedName>Replicator</wcm:distinguishedName>
        </wcm:lastModifier>
        <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/item/8b629b12-e16a-4afa-bbf5-a37ebee5a5b8"/>
        <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/item/8c01ede8-4ccd-410e-9e21-c4c37114f5f2"/>
        <wcm:displayTitle xml:lang="en">Articles List</wcm:displayTitle>
        <category label="Published" scheme="wcmrest:workflowState" term="PUBLISHED" xml:lang="en"/>
    </entry>
    <entry>
        <id>wcmrest:4ffeefcc-5539-4271-936a-7bd0ddf1644d</id>
        <title xml:lang="en">Article Toolbar</title>
        <summary xml:lang="en"></summary>
        <wcm:name>Article toolbar</wcm:name>
        <wcm:description xml:lang="en"></wcm:description>
        <wcm:type>LibraryAuthoringToolsComponent</wcm:type>
        <updated>2011-09-21T06:21:21.452Z</updated>
        <wcm:lastAccessed>2011-09-26T05:41:55.544Z</wcm:lastAccessed>
        <wcm:lastModifier>
            <wcm:distinguishedName>Replicator</wcm:distinguishedName>
        </wcm:lastModifier>
        <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/item/4ffeefcc-5539-4271-936a-7bd0ddf1644d"/>
        <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/item/8c01ede8-4ccd-410e-9e21-c4c37114f5f2"/>
        <wcm:displayTitle xml:lang="en">Article toolbar</wcm:displayTitle>
        <category label="Published" scheme="wcmrest:workflowState" term="PUBLISHED" xml:lang="en"/>
    </entry>
</feed>
```

There is one additional field present in each of the entries when a recent items query is performed. This field is the last accessed date of the item. This indicates the date and time at which the item was last viewed or edited through the authoring portlet. For example:

```
<wcm:lastAccessed>2011-09-26T05:41:55.544Z</wcm:lastAccessed>
```

Parameters

The following parameters, along with mime-type, are the only parameters that will work with the returned feed. All other parameters will be ignored.

Table 491. Parameters

Parameter	Details
sort	The `sort` parameter is appended to queries to determine how query results are sorted. The following values can be used with the `sort` parameter. • `accessed` • `author` • `created` • `modified` • `name` • `title` The values `_ascending` or `_descending` are appended to the query to determine sort order. For example, to sort a presentation template query in ascending order of creation, you would use: `/recent-items?type=PresentationTemplate&sort=created_ascending` To sort a presentation template query in descending order of creation, you would use: `/recent-items?type=PresentationTemplate&sort=created_descending` If `_ascending` or `_descending` are not specified, the results as displayed in ascending order.
type	This parameter is used to query items of a specific item type. For example, to query a list of components: `/recent-items?type=LibraryHTMLComponent`
page	This parameter is used with the `pagesize` parameter to define what set of results to display. For example, if `pagesize` is set to 5, and the `page` parameter is set to 2, then only results 6 - 10 are displayed. For example: `/recent-items?type=PresentationTemplate&pagesize=5&page=2`
pagesize	This parameter is used to restrict the number of items that are returned by a query to a set number. It can be used with the `page` parameter to return specific pages of results. For example, to restrict the number of queries to be returned to 5: `/recent-items?type=PresentationTemplate&pagesize=5`

How to use REST to work with favorite items

You can use REST service to display a list of favorite items. This function is the equivalent of the **Favorite Items** view in the library explorer.

URI: `/favorite-items`

Example:

```
<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm">
    <id>wcmrest:favorite-items</id>
    <title>wcmrest:favorite-items</title>
    <updated>2012-01-31T01:19:00.564Z</updated>
    <entry>
        <id>wcmrest:438dc2be-dbda-40bb-ad78-3c0f4bd11107</id>
        <title lang="en">Sample Article 2</title>
        <wcm:displayTitle lang="en">Sample Article 2</wcm:displayTitle>
        <summary lang="en"></summary>
        <wcm:name>Sample Article 2</wcm:name>
        <wcm:type>Content</wcm:type>
        <updated>2012-01-24T01:13:45.194Z</updated>
        <wcm:lastModifier>
            <wcm:distinguishedName>Replicator</wcm:distinguishedName>
        </wcm:lastModifier>
        <link label="Edit" rel="edit" href="/wps/mycontenthandler/!ut/p/digest!PQo5Yhy68oeppWcEz2sddA/wcmrest/Content/438dc2be-dbda-40bb-ad78-3c0f4bd11107" lang="en"/>
        <link label="Read" rel="alternate" href="/wps/mycontenthandler/!ut/p/digest!PQo5Yhy68oeppWcEz2sddA/wcmrest/Content/438dc2be-dbda-40bb-ad78-3c0f4bd11107" lang="en"/>
        <link label="Library" rel="library" href="/wps/mycontenthandler/!ut/p/digest!PQo5Yhy68oeppWcEz2sddA/wcmrest/item/a423287f-b0ce-4ee3-9c95-aa0939382228" lang="en"/>
        <category label="Published" scheme="wcmrest:workflowState" term="PUBLISHED" lang="en"/>
```

```
    </entry>
    <entry>
        <id>wcmrest:715cd5e8-ec36-420b-ad1c-fff80f39462b</id>
        <title lang="en">Sample Article</title>
        <wcm:displayTitle lang="en">Sample Article</wcm:displayTitle>
        <summary lang="en"></summary>
        <wcm:name>Sample Article</wcm:name>
        <wcm:type>Content</wcm:type>
        <updated>2012-01-24T01:13:47.981Z</updated>
        <wcm:lastModifier>
            <wcm:distinguishedName>Replicator</wcm:distinguishedName>
        </wcm:lastModifier>
        <link label="Edit" rel="edit" href="/wps/mycontenthandler/!ut/p/digest!PQo5Yhy68oeppWcEz2sddA/wcmrest/Content/715cd5e8-ec36-420b-ad1c-fff80f39462b" lang="en"/>
        <link label="Read" rel="alternate" href="/wps/mycontenthandler/!ut/p/digest!PQo5Yhy68oeppWcEz2sddA/wcmrest/Content/715cd5e8-ec36-420b-ad1c-fff80f39462b" lang="en"/>
        <link label="Library" rel="library" href="/wps/mycontenthandler/!ut/p/digest!PQo5Yhy68oeppWcEz2sddA/wcmrest/item/a423287f-b0ce-4ee3-9c95-aa0939382228" lang="en"/>
        <category label="Published" scheme="wcmrest:workflowState" term="PUBLISHED" lang="en"/>
    </entry>
</feed>
```

Parameters

The following parameters, along with mime-type, are the only parameters that work with the returned feed. All other parameters are ignored.

Table 492. Parameters

Parameter	Details
sort	The `sort` parameter is appended to queries to determine how query results are sorted. The following values can be used with the `sort` parameter. • `author` • `created` • `modified` • `name` • `title` The values `_ascending` or `_descending` are appended to the query to determine sort order. For example, to sort a presentation template query in ascending order of creation, you would use: `/favorite-items?type=PresentationTemplate&sort=created_ascending` To sort a presentation template query in descending order of creation, you would use: `/favorite-items?type=PresentationTemplate&sort=created_descending` If `_ascending` or `_descending` are not specified, the results as displayed in ascending order.
type	This parameter is used to query items of a specific item type. For example, to query a list of components: `/favorite-items?type=LibraryHTMLComponent`
page	This parameter is used with the `pagesize` parameter to define what set of results to display. For example, if `pagesize` is set to 5, and the `page` parameter is set to 2, then only results 6 - 10 are displayed. For example: `/favorite-items?type=PresentationTemplate&pagesize=5&page=2`
pagesize	This parameter is used to restrict the number of items that are returned by a query to a set number. It can be used with the `page` parameter to return specific pages of results. For example, to restrict the number of queries to be returned to 5: `/favorite-items?type=PresentationTemplate&pagesize=5`

Adding and removing favorite items

For item types that have an explicit REST URI, such as types that can be created or updated, the item can be added by performing an HTTP PUT to update the item, with the PUT request containing the favorites category.

For example:

```
HTTP 1.1 PUT /wps/mycontenthandler/wcmrest/LibraryHTMLComponent/47018149-fc6b-46af-a54d-1eab89a6fe
<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="http://www.ibm.com/xmlns/wcm">
  ... data ...
    <category scheme="wcmrest:favorite" term="true" lang="en"/>
    ... data ...
</entry>
```

- `term="true"` will add an item to the list of favorite items.
- `term="false"` will remove an item from the list of favorite items.
- If the "term" parameter is not present, the item will be removed from the list of favorite items.

For items that do not have an explicit REST URI, an HTTP POST request can be made specifying the generic URI of the item. For this to work the item must be created already.

For example:

Adding:

```
HTTP 1.1 POST /wps/mycontenthandler/wcmrest/favorite-items/additem?item-uri=/wps/mycontenthandler/!ut/p/digest!yGIcBv5sO9Vb0EY9LLhJyQ/wcmrest/item/65a46943-ed1c-4f5d-b497-03c18886ca8e
```

Removing:

```
HTTP 1.1 POST /wps/mycontenthandler/wcmrest/favorite-items/removeitem?item-uri=/wps/mycontenthandler/!ut/p/digest!yGIcBv5sO9Vb0EY9LLhJyQ/wcmrest/item/65a46943-ed1c-4f5d-b497-03c18886ca8e
```

Note: The item-uri parameter specifies the item to add, and can be in the expanded form, as shown in the previous examples, or the compact form. For example:

```
wcmrest:item/65a46943-ed1c-4f5d-b497-03c18886ca8e
```

REST: Attachments

You can use the REST service to attach images to some item types. This is equivalent to using the **Insert An Image** button in the authoring portlet.

An attachment is an image resource that is associated with another item that contains HTML, and can be referenced from within that item. Attachments are a special item type in the REST service because they cannot be directly referenced. This is because there is no URI associated with an attachment when it is created. An attachment cannot be read or updated, but can be deleted indirectly by running an update on the parent item.

Attachments can be added to the following item types:

- Presentation templates
- Rich text components
- HTML components

Note: You must create the item and add a rich text or HTML element before you create the attachment.

Creating

```
/ITEM-TYPE/ITEM-UUID/attachments
Content-Type: image/*
```

When you create an attachment, the binary data, but not encoding, of an image is sent to the attachments collection of an item. The Content-Type header field is set to the appropriate image type. For example: `image/jpg`, `image/png`

Example:

```
HTTP/1.1 POST
http://host:port/wps/mycontenthandler/wcmrest/LibraryHTMLComponent/ITEM-UUID/attachments
Content-Type: image/jpg
(... binary data ... )

201 Created
```

Reading

Not supported.

Updating

Not supported.

Deleting

An attachment cannot be directly referenced through the REST service, which means it cannot be directly deleted. However, it can be deleted indirectly by running an update operation on the parent item.

For example, this is some markup that is stored in an HTML component:

```
<h1> Example Delete </h1>
<img src='/wps/wcm/myconnect/65132264-fd8b-461c-b6ec-ccdd22524ea6/image.jpg?MOD=AJPERES'
 alt='' title='' border='0'  />
```

To remove the image, you would make the following update request:

```
HTTP/1.1 PUT
http://host:port/wps/mycontenthandler/wcmrest/LibraryHTMLComponent/ITEM-UUID/
Content-Type: text/html

<h1> Example Delete </h1>

200 OK
```

Generic reading by using REST services for Web Content Manager

Although not all item types are handled by the REST service, all item types can be read in a generic fashion by using the REST service.

You can send a GET request to any web content item by using the following URI:

```
/item/{item-uuid}
```

Example

```
GET /wps/mycontenthandler/wcmrest/item/c98d11e1-7f2a-480e-9aac-40eb1949cbda HTTP/1.0


HTTP/1.0 200 OK

<entry xmlns="http://www.w3.org/2005/Atom" xmlns:wcm="wcm/namespace">
    <id>c98d11e1-7f2a-480e-9aac-40eb1949cbda</id>
```

```
    <title>Web Content</title>
    <link rel="edit" href="/wps/mycontenthandler/!ut/p/wcmrest/item/c98d11e1-7f2a-480e-9aac-40eb1949cbda"/>
    <link rel="library" href="/wps/mycontenthandler/!ut/p/wcmrest/item/c98d11e1-7f2a-480e-9aac-40eb1949cbda"/>
    <updated>2011-05-16T20:42:29.979Z</updated>
    <wcm:type>interface com.ibm.workplace.wcm.api.Library</wcm:type>
    <wcm:state>PUBLISHED</wcm:state>
</entry>
```

Reference material for the Web Content Manager REST service

Reference material for REST response codes, links, media types, and attachments.

"Response codes for the Web Content Manager REST service"
These response codes are generated by the Web Content Manager REST service.

"Link relations" on page 3386
These definitions provide information on how different requests can be linked.

"Supported media types" on page 3392
The following media types are supported by the Web Content Manager REST service.

"REST Item Types" on page 3397
These item types are supported by the Web Content Manager REST service.

Response codes for the Web Content Manager REST service

These response codes are generated by the Web Content Manager REST service.

Table 493. Response codes

Response code	Message	Description
200	OK	The operation completed successfully.
201	Created	The resource was successfully created.
301	Moved Permanently	The resource addressed is known, however, its URI has changed.
400	Bad Request	Generic client side error. The request data is invalid in some way.
403	Forbidden	The request is formed correctly, but the server cannot carry out the operation.
404	Not Found	The URI specified is unknown to the REST service.
405	Method Not Allowed	The addressed resource does not support the HTTP method used.
406	Not Acceptable	The client specified an unsupported accept type.
409	Conflict	The request attempted to put the resource in an impossible or inconsistent state.
415	Unsupported Media Type	The server did not recognize the specified media type.

Table 493. Response codes (continued)

Response code	Message	Description
423	Locked	Unable to perform the requested operation, as the resource is locked.
500	Internal Server Error	An internal error occurred.

Link relations

These definitions provide information on how different requests can be linked.

Link relation purpose and operation

CF06 **access-control**

Purpose: The access settings of an item.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: NO

add-attachment

Purpose: Add attachment to the item (array component only, such as HTML, RichText component/element, and PresentationTemplate).

Supported Operation to the Link HREF: POST

Settable for PUT and POST operation: NO

approve

Purpose: Approve the item in current stage and move it into next stage in the worklfow.

Supported Operation to the Link HREF: POST

Settable for PUT and POST operation: NO

change-to-draft

Purpose: Go to the draft of the item.

Supported Operation to the Link HREF: POST

Settable for PUT and POST operation: NO

content-template

Purpose: The content template of the item.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: YES

create-draft

Purpose: Create a draft of the item. It also adds an item to a project if created within a portal project context.

Supported Operation to the Link HREF: POST

Settable for PUT and POST operation: NO

CF06 **default-presentation**

Purpose: The template set as the default presentation template for the item that is created from an authoring template.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: YES

CF06 **delete**

Purpose: Delete an item.

Supported Operation to the Link HREF: DELETE

Settable for PUT and POST operation: NO

draft-of

Purpose: The published item link of the draft item.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: NO

draft

Purpose: The draft link of the published item.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: NO

edit-media

Purpose: Pointing to the item itself with raw data type specified in the 'type' attribute. Designed to be used for 'Raw Data' Read and Update.

Supported Operation to the Link HREF: PUT, GET

Settable for PUT and POST operation: NO

edit

Purpose: Pointing to item itself.

Supported Operation to the Link HREF: PUT, POST, GET, DELETE

Settable for PUT and POST operation: NO

elements

Purpose: All the elements in the content/sitearea.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: NO

library

Purpose: The library that the item is stored in.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: YES

CF06 **new-content**

Purpose: Link to the new content item to be created from this content template.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: NO

CF06 **new-project**

Purpose: Link to the new project to be created from this project template.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: NO

CF06 **new-projecttemplate**

Purpose: Link to the new project template.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: NO

CF06 **new-sitearea**

Purpose: Link to the site area to be created from this site area template.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: NO

next-stage, expire, publish, submit-for-review

Purpose: Move item to next stage in the workflow.

Supported Operation to the Link HREF: POST

Settable for PUT and POST operation: NO

parent

Purpose: The parent of the item.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: YES, overrides the library if both specified

CF06 **presentation-override**

Purpose: The presentation template set as the presentation override on the item that is created from the authoring template.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation:

preview

Purpose: Used to preview content items and site areas.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: NO

previous-stage, withdraw-from-review

Purpose: Move item to previous stage in the workflow.

Supported Operation to the Link HREF: POST

Settable for PUT and POST operation: GET

project

Purpose: The project that the item belongs to, if it is a draft in a project.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: YES

CF06 **project-items**

Purpose: List the items associated with a project.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: NO

CF06 **prototype**

Purpose: The default content properties of an authoring template.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: YES

CF06 **prototype-properties**

Purpose: The default properties of an authoring template.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: YES

reject

Purpose: Decline the item in current stage and move it back to previous stage in the workflow.

Supported Operation to the Link HREF: POST

Settable for PUT and POST operation: NO

restart

Purpose: Restart the workflow for the item and move to the draft stage.

Supported Operation to the Link HREF: POST

Settable for PUT and POST operation: NO

CF06 **self**

Purpose: Read-only link back to item itself.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: NO

sitearea-template

Purpose: The site area template of the item.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: YES

versioned-item

Purpose: The individual version of the item.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: NO

versions

Purpose: All versions of the item.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: NO

workflow-stage

Purpose: The workflow stage of the item

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: NO

workflow

Purpose: The workflow of the item.

Supported Operation to the Link HREF: GET

Settable for PUT and POST operation: YES

CF06 access-control

```
<atom:links atom:rel="access-control"
atom:href="/wps/mycontenthandler/!ut/p/digest!2ducXmyQyM0CM4Nev9jHqw/ac/access:oid:Z6QReDe6BCAMI57IPD8MMS643C4JMG6HPC4MM47OHC4JMC6M9DCJPGCO1CE3OOCJ1"/>
```

add-attachment

```
<atom:links atom:rel="add-attachment"
atom:href="wcmrest:LibraryHTMLComponent/c0b72020-10b7-4197-a436-62a1d94ce03f/attachments"/>
```

approve

```
<atom:links atom:rel="approve"
atom:href="/wps/mycontenthandler/wcmrest/item/68b3bbb5-3b36-4c1f-94b1-6c3a037c975a/approve"/>
```

authoring-template

```
<atom:links atom:rel="authoring-template"
atom:href="/wps/mycontenthandler/wcmrest/item/18cfc80c-a490-4d75-9057-fed3db89de53"/>
```

change-to-draft

```
<atom:links atom:rel="change-to-draft"
atom:href="/wps/mycontenthandler/wcmrest/item/c0b72020-10b7-4197-a436-62a1d94ce03f/change-to-draft"/>
```

create-draft

```
<atom:links atom:rel="create-draft"
atom:href="/wps/mycontenthandler/wcmrest/item/e5846504-e4ae-496f-8f33-c06a8bfcb31d/create-draft"/>
```

decline

```
<atom:links atom:rel="decline"
atom:href="/wps/mycontenthandler/wcmrest/item/2ca1e0ce-3cc9-4810-b4d8-b28738286492/decline"/>
```

CF06 default-presentation

```
<atom:links atom:rel="default-presentation"
atom:href="/wps/mycontenthandler/wcmrest/PresentationTemplate/50c90dd5-3062-4bd9-b495-84716a9eaf58"/>
```

CF06 delete

```
<atom:links atom:rel="delete"
atom:href="/wps/mycontenthandler/!ut/p/digest!fdLLOZCYBjzgl3fVm_1pOA/wcmrest/ContentTemplate/9a30bf59-aad9-4fb0-99af-b621f09426a0"/>
```

draft-of

```
<atom:links atom:rel="draft-of"
atom:href="/wps/mycontenthandler/wcmrest/LibraryDateComponent/edbcb837-e9f0-4766-a4ec-e7fcdef12ca7"/>
```

draft

```
<atom:links atom:rel="draft"
atom:href="/wps/mycontenthandler/wcmrest/LibraryDateComponent/569537d4-79ac-4963-8b8c-6da57b8ddb55"/>
```

edit

```
<atom:links atom:rel="edit"
atom:href="/wps/mycontenthandler/wcmrest/LibraryHTMLComponent/10d5f7ca-f2a2-46b8-b649-e76a1ee8edee"/>
```

edit-media

```
<atom:links atom:rel="edit-media"
atom:type="text/html"
atom:href="/wps/mycontenthandler/wcmrest/LibraryHTMLComponent/10d5f7ca-f2a2-46b8-b649-e76a1ee8edee"/>
```

elements

```
<atom:links atom:rel="elements"
atom:href="wcmrest:Content/9d9b133b-1bab-40e7-a9bd-5b0ac86cf628/elements"/>
```

library

```
<atom:links atom:rel="library"
atom:href="/wps/mycontenthandler/wcmrest/item/957a67f2-9d70-469f-9d43-f63f78508e48"/>
```

CF06 new-content

```
<atom:links atom:rel="new-content"
atom:href="/wps/mycontenthandler/!ut/p/digest!fdLLOZCYBjzgl3fVm_1pOA/wcmrest/ContentTemplate/9a30bf59-aad9-4fb0-99af-b621f09426a0/new-content"/>
```

CF06 **new-project**

```
<atom:links atom:rel="new-project"
atom:href="/wps/mycontenthandler/!ut/p/digest!2ducXmyQyM0CM4Nev9jHqw/wcmrest/ProjectTemplate/9d7041d8-00aa-4433-90b3-bda2abdba3d4/new-project"/>
```

CF06 **new-projecttemplate**

```
<atom:links atom:rel="new-projecttemplate"
atom:href="/wps/mycontenthandler/!ut/p/digest!2ducXmyQyM0CM4Nev9jHqw/wcmrest/Project/c53c443f-30ef-425d-95fb-4310f0a12b0b/new-projecttemplate"/>
```

CF06 **new-sitearea**

```
<atom:links atom:rel="new-sitearea"
atom:href="/wps/mycontenthandler/!ut/p/digest!2ducXmyQyM0CM4Nev9jHqw/wcmrest/SiteAreaTemplate/80e3a5e1-f71a-480d-af3a-ad1fa3e86dc9/new-sitearea"/>
```

next-stage

```
<atom:links atom:rel="next-stage"
atom:href="/wps/mycontenthandler/wcmrest/item/a93ce36a-7a0d-4bda-be1f-e8db09295c8b/next-stage"/>
```

parent

```
<atom:links atom:rel="parent"
atom:href="/wps/mycontenthandler/wcmrest/item/fbcc2395-ca4c-44f2-9cb7-5f4ca359500f"/>
```

CF06 **presentation-override**

```
<atom:links atom:rel="presentation-override"
atom:href="/wps/mycontenthandler/wcmrest/PresentationTemplate/50c90dd5-3062-4bd9-b495-84716a9eaf58"/>
```

previous-stage

```
<atom:links atom:rel="previous-stage"
atom:href="/wps/mycontenthandler/wcmrest/item/e5846504-e4ae-496f-8f33-c06a8bfcb31d/previous-stage"/>
```

project

```
<atom:links atom:rel="project"
atom:href="/wps/mycontenthandler/wcmrest/Project/35b9120a-17d0-4dcb-b0ba-b034e34b50a6"/>
```

CF06 **project-items**

```
<atom:links atom:rel="project-items"
atom:href="/wps/mycontenthandler/!ut/p/digest!2ducXmyQyM0CM4Nev9jHqw/wcmrest/Project/c53c443f-30ef-425d-95fb-4310f0a12b0b/project-items"/>
```

CF06 **prototype**

```
<atom:links atom:rel="prototype"
atom:href="/wps/mycontenthandler/!ut/p/digest!fdLLOZCYBjzgl3fVm_1pOA/wcmrest/ContentTemplate/9a30bf59-aad9-4fb0-99af-b621f09426a0/Prototype"/>
```

CF06 **prototype-properties**

```
<atom:links atom:rel="prototype"
atom:href="/wps/mycontenthandler/!ut/p/digest!fdLLOZCYBjzgl3fVm_1pOA/wcmrest/ContentTemplate/9a30bf59-aad9-4fb0-99af-b621f09426a0/Prototype/properties"/>
```

restart

```
<atom:links atom:rel="restart"
atom:href="/wps/mycontenthandler/wcmrest/item/e5846504-e4ae-496f-8f33-c06a8bfcb31d/restart"/>
```

CF06 **self**

```
<atom:links atom:rel="self"
atom:href="/wps/mycontenthandler/!ut/p/digest!fdLLOZCYBjzgl3fVm_1pOA/wcmrest/ContentTemplate/9a30bf59-aad9-4fb0-99af-b621f09426a0"/>
```

versioned-item

```
<atom:links atom:rel="versioned-item"
atom:href="/wps/mycontenthandler/wcmrest/item/c0b72020-10b7-4197-a436-62a1d94ce03f/version/1"/>
```

versions

```
<atom:links atom:rel="versions"
atom:href="/wps/mycontenthandler/wcmrest/item/a93ce36a-7a0d-4bda-be1f-e8db09295c8b/versions"/>
```

workflow-stage

```
<atom:links atom:rel="workflow-stage"
atom:href="/wps/mycontenthandler/wcmrest/item/52c43f50-7a4e-4ad2-818a-8975d2362219"/>
```

workflow

```
<atom:links atom:rel="workflow"
atom:href="/wps/mycontenthandler/wcmrest/item/8d25860b-7a5c-4015-9cd5-bdcc60ce14bb"/>
```

Supported media types

The following media types are supported by the Web Content Manager REST service.

Metadata Media Types

Table 494. Metadata Media Types

Item Description	Item Type	Supported Media Types	Supported Methods
Authoring Tools Component	LibraryAuthoringToolsComponent	application/atom+xml application/json	GET, PUT, POST
Category	Category	application/atom+xml application/json	GET, PUT, POST
Content Template	ContentTemplate	application/atom+xml application/json	GET, PUT, POST
Content	Content	application/atom+xml application/json	GET, PUT, POST
Custom Workflow Action	CustomWorkflowAction	application/atom+xml application/json	GET, PUT, POST
Date Component	LibraryDateComponent	application/atom+xml application/json	GET, PUT, POST
Date Element	DateComponent	application/atom+xml application/json	GET, PUT, POST
Email Action	EmailAction	application/atom+xml application/json	GET, PUT, POST
Expire Action	ExpireAction	application/atom+xml application/json	GET, PUT, POST
File Component	LibraryFileComponent	application/atom+xml application/json	GET, PUT, POST
File Element	FileComponent	application/atom+xml application/json	GET, PUT, POST
Folder	Folder	application/atom+xml application/json	GET, PUT, POST
HTML Component	LibraryHTMLComponent	application/atom+xml application/json	GET, PUT, POST
HTML Element	HTMLComponent	application/atom+xml application/json	GET, PUT, POST
Image Component	LibraryImageComponent	application/atom+xml application/json	GET, PUT, POST
Image Element	ImageComponent	application/atom+xml application/json	GET, PUT, POST

Table 494. Metadata Media Types (continued)

Item Description	Item Type	Supported Media Types	Supported Methods
JSP Component	LibraryJSPComponent	application/ atom+xml application/json	GET, PUT, POST
JSP Element	JSPComponent	application/ atom+xml application/json	GET, PUT, POST
List Presentation Component	LibraryListPresentationComponent	application/ atom+xml application/json	GET, PUT, POST
Menu Component	LibraryMenuComponent	application/ atom+xml application/json	GET, PUT, POST
Navigator Component	LibraryNavigatorComponent	application/ atom+xml application/json	GET, PUT, POST
Numeric Component	LibraryNumericComponent	application/ atom+xml application/json	GET, PUT, POST
Numeric Element	NumericComponent	application/ atom+xml application/json	GET, PUT, POST
Option Selection Element	OptionSelectionComponent	application/ atom+xml application/json	GET, PUT, POST
Page Navigation Component	LibraryPageNavigationComponent	application/ atom+xml application/json	GET, PUT, POST
Personalization Component	LibraryPersonalizationComponent	application/ atom+xml application/json	GET, PUT, POST
Portal Page	PortalPage	application/ atom+xml application/json	GET
Presentation Template	PresentationTemplate	application/ atom+xml application/json	GET, PUT, POST
Project Template	ProjectTemplate	application/ atom+xml application/json	GET, PUT, POST
Project	Project	application/ atom+xml application/json	GET, PUT, POST
Publish Action	PublishAction	application/ atom+xml application/json	GET, PUT, POST
Rich Text Component	LibraryRichTextComponent	application/ atom+xml application/json	GET, PUT, POST

Table 494. Metadata Media Types (continued)

Item Description	Item Type	Supported Media Types	Supported Methods
Rich Text Element	RichTextComponent	application/atom+xml application/json	GET, PUT, POST
Scheduled Move Action	ScheduledMoveAction	application/atom+xml application/json	GET, PUT, POST
Search Component	LibrarySearchComponent	application/atom+xml application/json	GET, PUT, POST
Short Text Component	LibraryShortTextComponent	application/atom+xml application/json	GET, PUT, POST
Short Text Element	ShortTextComponent	application/atom+xml application/json	GET, PUT, POST
Site Area Template	SiteAreaTemplate	application/atom+xml application/json	GET, PUT, POST
Site Area	SiteArea	application/atom+xml application/json	GET, PUT, POST
Stylesheet Component	LibraryStyleSheetComponent	application/atom+xml application/json	GET, PUT, POST
Taxonomy	Taxonomy	application/atom+xml application/json	GET, PUT, POST
Text Component	LibraryTextComponent	application/atom+xml application/json	GET, PUT, POST
Text Element	TextComponent	application/atom+xml application/json	GET, PUT, POST
User Selection Component	LibraryUserSelectionComponent	application/atom+xml application/json	GET, PUT, POST
User Selection Element	UserSelectionComponent	application/atom+xml application/json	GET, PUT, POST
Username Component	LibraryUserNameComponent	application/atom+xml application/json	GET, PUT, POST
Version Action	VersionAction	application/atom+xml application/json	GET, PUT, POST
Workflow Stage	Workflow Stage	application/atom+xml application/json	GET, PUT, POST

Table 494. Metadata Media Types (continued)

Item Description	Item Type	Supported Media Types	Supported Methods
Workflow	Workflow	application/ atom+xml application/json	GET, PUT, POST

Raw Data Media Types

Table 495. Raw Data Media Types

Item Description	Item Type	Supported Media Types	Supported Methods
Content	Content	application/ vnd.ibm.wcm+xml	GET, PUT
Custom Workflow Action	CustomWorkflowAction	application/ vnd.ibm.wcm+xml	GET, PUT
Date Component	LibraryDateComponent	application/ vnd.ibm.wcm+xml	GET, PUT
Date Element	DateComponent	application/ vnd.ibm.wcm+xml	GET, PUT
Email Action	EmailAction	application/ vnd.ibm.wcm+xml	GET, PUT
Expire Action	ExpireAction	application/ vnd.ibm.wcm+xml	GET, PUT
File Component	LibraryFileComponent	*/*	GET, PUT
File Element	FileComponent	*/*	GET, PUT
HTML Component	LibraryHTMLComponent	text/html	GET, PUT
HTML Element	HTMLComponent	text/html	GET, PUT
Image Component	LibraryImageComponent	image/* application/ vnd.ibm.wcm+xml	GET, PUT
Image Element	ImageComponent	image/* application/ vnd.ibm.wcm+xml	GET, PUT
JSP Component	LibraryJSPComponent	application/ vnd.ibm.wcm+xml	GET, PUT
JSP Element	JSPComponent	application/ vnd.ibm.wcm+xml	GET, PUT
Numeric Component	LibraryNumericComponent	application/ vnd.ibm.wcm+xml	GET, PUT
Numeric Element	NumericComponent	application/ vnd.ibm.wcm+xml	GET, PUT
Option Selection Element	OptionSelectionComponent	application/ vnd.ibm.wcm+xml	GET, PUT
Portal Page	PortalPage	application/ vnd.ibm.wcm+xml	GET
Presentation Template	PresentationTemplate	text/html	GET, PUT
Project Template	ProjectTemplate	application/ vnd.ibm.wcm+xml	GET, PUT

Table 495. Raw Data Media Types (continued)

Item Description	Item Type	Supported Media Types	Supported Methods
Project	Project	application/ vnd.ibm.wcm+xml	GET, PUT
Publish Action	PublishAction	application/ vnd.ibm.wcm+xml	GET, PUT
Rich Text Component	LibraryRichTextComponent	text/html	GET, PUT
Rich Text Element	RichTextComponent	text/html	GET, PUT
Scheduled Move Action	ScheduledMoveAction	application/ vnd.ibm.wcm+xml	GET, PUT
Short Text Component	LibraryShortTextComponent	text/html	GET, PUT
Short Text Element	ShortTextComponent	text/html	GET, PUT
Site Area Template	SiteAreaTemplate	application/ vnd.ibm.wcm+xml	GET, PUT
Site Area	SiteArea	application/ vnd.ibm.wcm+xml	GET
Stylesheet Component	LibraryStyleSheetComponent	text/css application/ vnd.ibm.wcm+xml	GET, PUT
Text Component	LibraryTextComponent	text/html	GET, PUT
Text Element	TextComponent	text/html	GET, PUT
User Selection Component	LibraryUserSelectionComponent	application/ vnd.ibm.wcm+xml	GET, PUT
User Selection Element	UserSelectionComponent	application/ vnd.ibm.wcm+xml	GET, PUT
Version Action	VersionAction	application/ vnd.ibm.wcm+xml	GET, PUT
Workflow Stage	Workflow Action	application/ vnd.ibm.wcm+xml	GET, PUT
Workflow	Workflow	application/ vnd.ibm.wcm+xml	GET, PUT

REST Item Types

These item types are supported by the Web Content Manager REST service.

Level of support

Basic Basic support means that the create, read, update operations operate only on those fields that are common to all items. This includes fields such as name, title, description, text providers, and workflow.

Item Specific
Item Specific support means that the item type has support for fields beyond those fields common to all item types. For example, the element on a content item can be accessed, or the HTML element in an HTML component can be modified. Not all fields in each item type are handled.

Table 496. Item support levels

Item type	Support level
Authoring Tools Component	Basic

Table 496. Item support levels (continued)

Item type	Support level
Category	Item Specific
Content Template	Item Specific
Content	Item Specific
Custom Workflow Action	Item Specific
Date Component / Date Element	Item Specific
Email Action	Item Specific
Expire Action	Item Specific
File Component / File Element	Item Specific
Folder	Item Specific
HTML Component / HTML Element	Item Specific
Image Component / Image Element	Item Specific
JSP Component / JSP Element	Item Specific
Link Component / Link Element	Item Specific
List Presentation Component	Basic
Menu Component	Basic
Navigator Component	Basic
Numeric Component / Numeric Element	Item Specific
Page Navigation Component	Basic
Personalization Component	Basic
Portal Page	Item Specific
Presentation Template	Item Specific
Project Template	Item Specific
Project	Item Specific
Publish Action	Item Specific
Reference Component / Reference Element	Item Specific
Scheduled Move Action	Item Specific
Search Component	Basic
Short Text Component / Short Text Element	Item Specific
Site Area Template	Item Specific
Site Area	Item Specific
Stylesheet Component	Item Specific
Taxonomy	Item Specific
Text Component / Text Element	Item Specific
User Selection Component / User Selection Element	Item Specific
Username Component	Basic
Version Action	Item Specific
Workflow Stage	Item Specific
Workflow	Item Specific

How to display data from external sources

You display data from external sources by using the same methods as you would when you create a website.

Displaying data

You can display content from external sources by using standard Java tag libraries and a JSP element. Java code that uses standard Java APIs or tag libraries can be used to display and format data from databases, LDAP repositories, or send email.

If you use a rendering portlet to display web content on a portal page, you can also use other IBM WebSphere Portal Express portlets on the same portal page to display data.

Web page aggregation

Content from external websites and IBM Web Content Manager can be displayed together on a portal page by using standard WebSphere Portal portlets for displaying content from external websites. Refer to the web Page portlet and Web Clipping portlet sections of this IBM Knowledge Center for information about creating and configuring these portlets.

This function is only available when you display content in WebSphere Portal. These portlets are not accessible from the Web Content Manager servlet, though standard Java API or tag libraries can be used with a JSP element to achieve the same result.

Instrumenting web content for Active Site Analytics

You can collect information from web content for Active Site Analytics.

About this task

For collecting information about the web content that the portal renders, use the following microformat tags:

asa.wcm.content_item.path
Use this tag to identify the content path. The tag contains the unique identifier of the content item in IBM Web Content Manager.

asa.wcm.content_item.title
Use this tag to identify the content title. The tag contains the display title of the content item.

asa.wcm.content_item.authors
Use this tag to identify the authors of the content. The tag contains one of the authors of the content item.

asa.wcm.content_item.lastmodified
Use this tag to identify the last modification date of the content. The tag contains the date on which the content item was last modified.

Web Content Manager provides the following methods for collecting information about web content:

- Using the analytics data rendering plug-in tag.

- Using the sample HTML component for Active Site Analytics that is provided with Web Content Manager.
- Using the default microformat tags that are supported by web content viewers.

"Using the sample HTML component for Active Site Analytics"
The HTML – Analytics component is a sample HTML component that you can use to instrument web content for Active Site Analytics. You can use this component to insert the supported microformat tags for web content into your content or presentation templates.

"Enabling default microformat support in Web Content Viewers"
Web Content Viewers provide support for Active Site Analytics microformats by default. You can use this support to inject microformats into your content design or presentation templates.

Related tasks:

"Setting up site analysis for the Web Content Viewer" on page 411
To track usage data for the Web Content Viewer, you can configure the portal for site analysis logging for the Web Content Viewer.

Related information:

Using the analytics data rendering plug-in tag

Using the sample HTML component for Active Site Analytics

The HTML – Analytics component is a sample HTML component that you can use to instrument web content for Active Site Analytics. You can use this component to insert the supported microformat tags for web content into your content or presentation templates.

About this task

The HTML – Analytics component is located in the Web Content Templates library. You can adapt this sample to the requirements of your aggregator and Active Site Analytics setup. By default, the sample inserts the following microformat tags:

- asa.wcm.content_item.path
- asa.wcm.content_item.title
- asa.wcm.content_item.id
- asa.wcm.content_item.authors
- asa.wcm.content_item.lastmodified

Procedure

Insert the HTML – Analytics component in your content or presentation template design by adding the [Component] tag.
For example:

```
[Component name="Web Content Templates/HTML - Analytics"]
```

Enabling default microformat support in Web Content Viewers

Web Content Viewers provide support for Active Site Analytics microformats by default. You can use this support to inject microformats into your content design or presentation templates.

About this task

The viewer supports the following microformat tags:

- asa.wcm.content_item.title

- asa.wcm.content_item.path

Procedure

Edit the portlet preferences for the Web Content Viewer, and set the value of the **WCM_ENABLE_ASA_TAGS** preference to `true`. By default, the preference value is `false`.

Note: The default viewer has the unique name of ibm.portal.Web.Content.Viewer.Jsr286.

- To use the administration user interface to set the preference for all instances of the viewer, complete these steps:
  1. Click the **Administration menu** icon. Then, click **Portlet Management** > **Portlets**..
  2. Locate the Web Content Viewer from the list.
  3. Click the **Configure portlet** icon, and set the preference value.
- To set the preference for all instances or only a single instance of the viewer, you can also use the XML configuration interface.

Results

After you set the preference, the viewer automatically inserts the microformat tags into each piece of content that it renders.

Java messaging services for web content

Web Content Manager supports for the notification of events such as item state changes, or services starting and stopping. These notifications can be delivered as messages to the Java messaging service.

About this task

The event classes can be delivered as messages to the Java messaging service:

Item events:

- Item created
- Item updated
- Item moved
- Item deleted

Syndication events:

- Starting
- Stopping

Pre-render events:

- Starting
- Stopping

Procedure

1. Configure the messaging services parameters in the `WCM MessagingService` service by using the WebSphere Integrated Solutions Console.
2. Run the following command from the *wp_profile_root*`/ConfigEngine` directory:

Windows

```
ConfigEngine.bat create-wcm-jms-resources
-DPortalAdminId=username -DPortalAdminPwd=password
-DuseRemoteEndPoints=true/false
```

Linux

```
./ConfigEngine.sh create-wcm-jms-resources
-DPortalAdminId=username -DPortalAdminPwd=password
-DuseRemoteEndPoints=true/false
```

IBM i

```
ConfigEngine.sh create-wcm-jms-resources -DPortalAdminId=username
-DPortalAdminPwd=password -DuseRemoteEndPoints=true/false
```

Note: An administrator user name and password is not needed if you specify the portal administrator user name and password by using the `PortalAdminId` and `PortalAdminPwd` settings in the `wkplc.properties` file.

Note: The `-DuseRemoteEndPoints` parameter is only used on clustered systems. If set to true, the task uses all node end points on the current setup. If set to "false", the task uses the end points of the current node.

3. Restart WebSphere Portal.

Results

The `create-wcm-jms-resources` command creates a topic space that is named `IWK.Topic.Space` and the topics `IWKTopics_Items`, `IWKTopics_PreRender`, and `IWKTopics_Syndication` in that space for the different events.

The messages are sent with no expiration time and need to be consumed, otherwise the queue fills up.

Note: There is a default limit of 50.000 messages per queue topic. When that limit is reached messages cannot be stored anymore in the queue and exceptions are logged in the Portal server log file.

For information about the default implementation of the WebSphere Application Server JMS implementation and choices of message providers, read Types of messaging providers.

Related tasks:

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Related reference:

"Web Content Manager service configuration" on page 353
Configuration services for IBM Web Content Manager contain settings for the general operation of the web content system, including settings for messaging, pre-rendering, and searching.

Developing basic PAA file applications

Solution developers can create their own Portal Application Archive (PAA) files. The developers can then use the configuration wizard to add on their applications to their IBM WebSphere Portal Express environment.

About this task

Review the following requirements before you create your own Portal Application Archive (PAA) file:

"Checking server dependency"
When you install a Portal Application Archive (PAA) file, you must verify that the file is compatible with the current version of IBM WebSphere Portal Express.

"PAA dependencies for deployment and removal" on page 3405
The deployment and removal of a Portal Application Archive (PAA) file might depend on deployment of other PAA applications. You can specify the PAA dependencies for a PAA file in the assembly level `sdd.xml` file.

"Create a Portal Application Archive (PAA) file" on page 3407
Use the Portal Application Archive (PAA) file format to install applications with the Solution Installer. The PAA file is a customized compressed file that contains an application. This application is installed on IBM WebSphere Portal Express with the ConfigEngine.

Related concepts:

"Install and uninstall add-ons using the Configuration Wizard" on page 212
You can install add-on functionality to your WebSphere Portal Express with the solution installer through the Configuration Wizard. The add-ons that are accepted by the configuration options are `.paa` files. For more information, see the solution installer documentation.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

Checking server dependency

When you install a Portal Application Archive (PAA) file, you must verify that the file is compatible with the current version of IBM WebSphere Portal Express.

About this task

The Solution Installer uses the following two methods to check for dependencies:

Dependency checking for PAA files
The Solution Installer can check dependencies for the PAA file. To define the dependencies for a PAA file, you must edit the assembly level `sdd.xml` file in the PAA directory. Open the `sdd.xml` file and search for the `<rootIU>` tag. Add the `<serverVersionDependency>` tag to the `<rootIU>` tag with the following attributes:

- **`name`**: This attribute must be set to `PortalServer`
- **`lowerVersion`**: This attribute is optional. Only one WebSphere Portal Express version value is acceptable for this attribute.
- **`higherVersion`**: This attribute is optional. Only one WebSphere Portal Express version value is acceptable for this attribute.
- **`versions`**: This attribute is optional. You can add multiple WebSphere Portal Express version values for this attribute as a comma-separated list.

For example, you might add the following information to your sdd.xml file:

```
<serverVersionDependency name="PortalServer" lowerVersion="6.0.0.0"
                         higherVersion="8.5.0.0" versions="7.0.0.1,7.0.0.2" />
```

If you want to add dependencies for a specific fix level, you must add the following attributes to the <server> subelement within the <serverVersionDependency> element:

- **version**: This value is the version of WebSphere Portal Express where you want to set the level restriction.
- **fixlevel**: This attribute is optional. This attribute is compared against the current installation fix level to determine any dependencies. If the attribute is not set, then it is ignored.
- **lower**: This attribute is optional. Set this value to true if the **fixlevel** is the lowest version for the current server.
- **higher**: This attribute is optional. Set this value to true if the **fixlevel** is the highest version for the current server.

For example, you might add the following information to your sdd.xml file:

```
<serverVersionDependency name="PortalServer" lowerVersion=""
                         higherVersion="" versions="" >
<server version="8.5.0.0" fixlevel="CF02" lower="true"
     higher="false" />
</serverVersionDependency>
```

The Solution Installer uses the following rules to determine whether a PAA file is compatible with the current WebSphere Portal Express version:

- The Solution Installer assumes that the file is compatible in the following situations:
  - If the <serverVersionDependency> tag is not found in the assembly level sdd.xml file
  - If the <serverVersionDependency> tag exists but has no attributes that are defined
- The Solution Installer installs the PAA file if the following information is true:
  - The **lowerVersion** attribute is set and the value is equal to or greater than the current version of WebSphere Portal Express.
  - The **higherVersion** attribute is set and the current version of WebSphere Portal Express is less than or equal to its value.
- If the **versions** attribute is set, the Solution Installer installs the PAA file if the current WebSphere Portal Express matches one of the values.
- If a <server> element is included, the PAA file is installed if the following information is true:
  - If **version** and **fixlevel** are equal to the current server information
  - If **lower** is true, any fix pack level that is greater than **fixlevel**
  - If **higher** is true, any fix pack level that is less than or equal to **fixlevel**

Black list to block deploying PAA files that must not be deployed

The Solution Installer checks the blacklist.txt file in the /WebSphere/PortalServer/solutioninstaller/wp.si directory. The blacklist.txt file contains a list of PAA files that WebSphere Portal Express cannot deploy. The format for the blacklist.txt file is *assemblyname*: *version_number*. Multiple versions are allowed as a semicolon-separated list; for example: *assemblyname*: *version_number1*;*version_number2*.

When you install a PAA file, the Solution Installer reads the assembly name and version from the assembly level `sdd.xml` file. The Solution Installer then looks for the matching information in the `blacklist.txt` file. If the Solution Installer finds a match in the `blacklist.txt` file, the PAA file is not installed.

PAA dependencies for deployment and removal

The deployment and removal of a Portal Application Archive (PAA) file might depend on deployment of other PAA applications. You can specify the PAA dependencies for a PAA file in the assembly level `sdd.xml` file.

Add PAA dependencies for deploying a PAA file

The deployment of a PAA file might depend on deployment of other PAA applications. You can specify the PAA dependencies for deploying a PAA file in the assembly level `sdd.xml` file.

To define the PAA dependencies for a PAA file, you must edit the assembly level `sdd.xml` file in the PAA directory. Open the `sdd.xml` file and search for the `<rootIU>` tag. Add the `<paaDependencies>` tag to the `<rootIU>` tag. Also, add the `<paaDependency>` tag for each PAA dependency in the `<paaDependencies>` tag. The `<paaDependency>` tag has the following attributes:

`name`
This attribute is required. Set the value to the name of the PAA dependency.

`lowerVersion`
This attribute is optional. Only one PAA dependency version value is acceptable for this attribute.

`higherVersion`
This attribute is optional. Only one PAA dependency version value is acceptable for this attribute.

`versions`
This attribute is optional. You can add multiple PAA dependency version value for this attribute as a comma-separated list.

You can specify more than one `<paaDependency>` tag in the `<paaDependencies>` tag. For example, you might add the following information to your `sdd.xml` file:

```
<paaDependencies>
 <paaDependency name="Dependency1" lowerVersion="8.0.0.0" higherVersion="8.0.0.1" versions="8.5.0.
 <paaDependency name="Dependency2" lowerVersion="8.0.0.0" higherVersion="8.0.0.1" versions="8.5.0.
</paaDependencies>
```

All the PAA dependencies that are specified in the `<paaDependencies>` tag must be deployed on the Portal server. The version of each PAA dependency must match the values that are specified in `<paaDependencies>` tag. Then, the PAA file can be deployed into the Portal server.

The Solution Installer uses the following rules to determine whether a PAA file can be deployed:

- The Solution Installer assumes the PAA dependency requirement is satisfied in the following situations:
  - If the `<paaDependency>` tag is not found in the assembly level `sdd.xml` file
  - If the `<paaDependency>` tag exists but has no attributes that are defined

- All the PAA dependencies that are specified in the <paaDependencies> tag satisfy the requirement. The requirement is, for each of the PAA dependencies that are specified in a <paaDependency> tag, at least one of the following is true:
  - The PAA dependency is deployed. The **lowerVersion** attribute is set and the value is equal to or greater than the current version of PAA dependency.
  - The PAA dependency is deployed. The **higherVersion** attribute is set and the current version of PAA dependency is less than or equal to its value.
  - The PAA dependency is deployed. The **versions** attribute is set and the current version of PAA dependency matches one of the values.

Add PAA dependencies for removing a PAA file

The removal of a PAA file might depend on removal of other PAA applications. You can specify the PAA dependency for removing a PAA file in the assembly level sdd.xml file.

To define the PAA dependencies for a PAA file, you must edit the assembly level sdd.xml file in the PAA directory. Open the sdd.xml file and search for the <rootIU> tag. Add the <paaDependencies> tag to the <rootIU> tag. And add <removePaaDependency> tag for each PAA dependency in the <paaDependencies> tag. The <removePaaDependency> tag has the following attributes:

name
This attribute is required. Set the value to the name of the PAA dependency.

lowerVersion
This attribute is optional. Only one PAA dependency version value is acceptable for this attribute.

higherVersion
This attribute is optional. Only one PAA dependency version value is acceptable for this attribute.

versions
This attribute is optional. You can add multiple PAA dependency version value for this attribute as a comma-separated list.

You can specify more than one <paaDependency> tag in the <paaDependencies> tag. For example, you might add the following information to your sdd.xml file:

```
<paaDependencies>
 <removePaaDependency name="Dependency1" lowerVersion="8.0.0.0" higherVersion="8.0.0.1" versions="8.5
 <removePaaDependency name="Dependency2" lowerVersion="8.0.0.0" higherVersion="8.0.0.1" versions="8.5
</paaDependencies>
```

The Solution Installer uses the following rules to determine whether a PAA file can be removed:

- The Solution Installer assumes the PAA dependency requirement is satisfied in the following situations:
  - If the <removePaaDependency> tag is not found in the assembly level sdd.xml file
  - If the <removePaaDependency> tag exists but has no attributes that are defined
- All the PAA dependencies satisfy the requirement. The requirement is, for each of the PAA dependencies that are specified in a <removePaaDependency> tag, the PAA dependency is not deployed, or not any of the following is true:
  - The PAA dependency is deployed. The **lowerVersion** attribute is set and the value is less than the current version of PAA dependency.

- The PAA dependency is deployed. The **higherVersion** attribute is set and the value is greater than the current version of PAA dependency.
- The PAA dependency is deployed. The **versions** attribute is set and the current version of PAA dependency matches any one of the values.

Create a Portal Application Archive (PAA) file

Use the Portal Application Archive (PAA) file format to install applications with the Solution Installer. The PAA file is a customized compressed file that contains an application. This application is installed on IBM WebSphere Portal Express with the ConfigEngine.

Many steps might be required to install and configure the individual resources of an application to IBM WebSphere Portal Express. This fact is especially true when you deploy applications with many resource types. The Solution Installer automates many of these deployment tasks. The Solution Installer uses resources that are contained in a Portal Application Archive (PAA) file. The PAA file is a ZIP compression file that has a specific directory structure. The PAA format informs Solution Installer how to install the application and provides the installable artifacts.

Currently, the Solution Installer focus is on installing applications to WebSphere Portal Express. Potentially in the future, IBM WebSphere Application Server solutions might be supported. The Solution Installer relies on the ConfigEngine, which is not specific to WebSphere Portal Express.

The PAA format can be employed to handle deployments that range from applications with only a few configuration steps to large-scale enterprise portal solutions. The directory structure of the PAA file is important to Solution Installer when you determine how to install a specific artifact, for example, how to handle shared library files. In addition, the software definition descriptor (`sdd.xml`) files also play a significant role in determining the installation steps. All required extension points for installation must be specified in an `sdd.xml` file local to the component. The component level `sdd.xml` file can now be generated automatically during the installation phase.

The PAA format can reduce the work that is required to create a deployable solution. Many deployment tasks can use a default configuration that is found in the Solution Installer. For complex applications, extra work is required to provide custom installation features. Using the PAA format with the Solution Installer reduces the production time for creating a deployable solution for your application.

A number of sample files that demonstrate the overall structure and usage of the PAA file format are included in the *PortalServer_root*`/doc/paa-samples` directory. These examples act as reference material for the remainder of the file specification documentation.

"PAA file structure overview" on page 3408
Each PAA file is a structured ZIP compression file. Because the Solution installer uses Ant tasks to expand or parse the PAA file, it needs to be a ZIP compression file that can be decoded by the `java.util.zip` class.

"Installation tasks" on page 3417
Solution installer supports the auto generation of deployment tasks for specific resource types. The creation of these tasks occurs during the installation or registration phase for the PAA file with the Solution Installer and the ConfigEngine.

"Order of installation of scripts and artifacts" on page 3419
Often the order that scripts are run or artifacts that are deployed is important for the success of the installation. This information is both true for artifacts or scripts in a single component and the order in which scripts contained in multiple components are installed.

"The sdd.xml file" on page 3420
The sdd.xml files perform a number of different roles in the PAA file structure. They allow the developer to control, with ConfigEngine extension points, how an application is installed. They also inform the ConfigEngine of the type of installation to be processed. It also determines, in terms of the ConfigEngine, where the deployable files are stored after registration with the ConfigEngine.

"Property files" on page 3427
Two different types of properties are contained within a PAA distribution. They can be classified as being either user editable that is requiring user input, for example a database URL, or as developer-provided settings required for a component to function. However, for user convenience, a separation between user properties from those settings that are supplied by the developer is advised.

"Database properties for the Solution Installer" on page 3428
Some Portal Application Archive (PAA) files require access to an external database. The database properties are stored in either the assemblyName.properties file for the assembly or in the componentName.properties file of the component requiring database support.

"Virtual portals in the PAA file" on page 3431
There are many situations where a user might want to install the applications that are contained in a Portal Application Archive (PAA) file directly to a virtual portal.

PAA file structure overview

Each PAA file is a structured ZIP compression file. Because the Solution installer uses Ant tasks to expand or parse the PAA file, it needs to be a ZIP compression file that can be decoded by the java.util.zip class.

At the top level of a PAA file is the root directory. There are no restrictions on the name of this directory; however, use something that is unique and does not clash with other registered PAA files or assemblies. There must be only one root directory in a PAA file. The name of this directory is significant because it is the reference name for your PAA content. It is the same value as the assemblyName when you create the assembly level sdd.xml file. Read "The sdd.xml file" on page 3420 for information.

You can find a sample top-level directory in the *PortalServer_root*/doc/paa-samples/sample1.paa file. The root directory contains the documentation and components directories and an sdd.xml file.

The sdd.xml file has the following roles:

- It informs the ConfigEngine about the assembly of components.
- It provides the ConfigEngine with the list of components and their locations within the directory structure.
- It points to each component sdd.xml file so that the installation function can process the installation of the individual components.

  "Documentation directory" on page 3409
  Place all documents and ID-related artifacts for the application in the documentation directory.

"Components directory"
The components directory and its subdirectories in the PAA file archive are where all deployable artifacts must be stored. If you examine the content of the components directory in the *PortalServer_root*/doc/paa-samples/Sample1.paa file, you see one component, sample1. There is always at least one component that is contained in a PAA file. However, there is no limit on the number of extra components that you can include.

Documentation directory:

Place all documents and ID-related artifacts for the application in the documentation directory.

The PAA package developer must organize the documentation directory per the relationship between the documents and the flow of content information. The Solution Installer copies the content to the correct location in the expanded archive under the PAA offering directory; for example, *wp_profile_root*\paa\sample_paa\documentation*.*.

Note: The documentation directory is the only recommended place for documentation that is related to the application. There is no provision within the PAA format to provide documentation at the component level.

Components directory:

The components directory and its subdirectories in the PAA file archive are where all deployable artifacts must be stored. If you examine the content of the components directory in the *PortalServer_root*/doc/paa-samples/Sample1.paa file, you see one component, sample1. There is always at least one component that is contained in a PAA file. However, there is no limit on the number of extra components that you can include.

The number of available components depends on how you want to organize your deployable artifacts. Potentially all the deployable artifacts can be stored in a single component. However, if there are multiple stand-alone applications to be stored in the PAA file, create a component for each application. Also, if you want to be able to reuse components across PAA file distributions, then it makes sense to separate artifacts into multiple components.

There is no limit on the type of artifacts that can be contained within a specific component. There is a limitation on where the artifacts can be placed within the component directory sub tree structure.

A component can include artifacts and configuration details for an entire application. It can also contain only artifacts that relate to a certain part of the application. For example, you might include all your theme-related artifacts in one component, and your XML access scripts to create the pages for your site in another. The Solution Installer is not concerned with the approach selected. It processes each component based on the dependencies that are listed in the sdd.xml file. However, it might make sense to have some separation to handle reusability of components.

Using the *PortalServer_root*/doc/paa-samples/sample1.paa example, open the components/Sample1 directory. The following directories must be present with the component level sdd.xml file:

config This directory contains includes and templates directories. Both directories

are important if you plan to add custom tasks to aid the installation or configuration of the component artifacts.

includes
This directory is where the ConfigEngine looks for tasks that implement the extension points that are listed in the component level `sdd.xml` file. The name of the xml file that contains the Ant tasks is not hardcoded, so the ConfigEngine automatically loads any XML files that are found in this directory. The Ant tasks do not have to be stored in a single file. They can be spread over multiple files that are all picked up by the ConfigEngine.

templates
This directory is where extra script files are stored for configuration tasks. For example, if you deploy a WAR file and want to run portlet configuration tasks, you can place the XML scripts in this directory. Copy the war file to the `profile_dir/installableApps` directory with your custom task implementation and reference this location in your XMLAccess script.

Note: The Solution Installer does not automatically run the scripts that are stored in this location. Instead, the custom tasks are responsible for calling these scripts.

content
This directory is where you can store IBM Web Content Manager libraries and other content related artifacts for import. The `component/content` directory contains the following subdirectories:

jcr This directory contains JCR-related artifacts and a directory structure of nodes. Each entry is either a file or a directory, with associated metadata (title, last modified, permissions). Content that is found in this directory must be installed with custom code. The Solution Installer does not auto-generate any tasks to handle resources that are found in this location.

jsp Place any JSP files that you want to package as part of your application in this directory. Usually, the custom code handles resources in this location. However, Web Content Manager related JSP files are treated as a special case. Place such files in a sub folder that is called `wcm` under the `jsp` directory, for example, `jsp/wcm`. The Solution Installer copies all files and folders to the relevant location on the server: `${WasUserHome}/installedApps/${NodeName}/WCM_EXTENSION.ear/wp.wcmextension.war/jsp/wcm/content`.

Note: The Solution Installer does not run any default tasks to handle other JSP content, excluding those tasks that are found in the `jsp/wcm` directory. A developer must provide a custom task to ensure that this content is copied to the correct location.

wcm This directory contains Web Content Manager libraries. Each subdirectory of the `wcm` directory represents a separate library. These libraries are a specialized form of JCR artifacts. Web Content Manager libraries are separated into their own directory due to the process required to install them with the default function. For

example, if you have a library that is called lib1, you might place the content in the `content/wcm/lib1` directory; for example, `content/wcm/lib1/554ee7f5`.

Multiple Web Content Manager libraries that are contained within a single subdirectory are supported. However, it places a limitation on the Solution Installer's ability to delete and replace libraries. That is, the task that Solution installer relies upon to import takes everything within a supplied subdirectory and attempts to import to the server. Therefore, to be able to successfully replace all the libraries in the directory, they must all be marked for deletion. Where libraries are allocated to their own subdirectory, they can be replaced on the system individually when there are no interlibrary dependencies. Another existing library does not have a dependency on items in the library that is selected for deletion.

webdav This directory and its sub tree structure contain artifacts that must be uploaded to the WebDAV file store. There are four possible subdirectories. Each one is named to reflect the type of function that is provided in the files supplied. The following options are available:

iwidgets
Place `.zip` files that contain iWidgets that must be uploaded to the WebDAV file store and registered with WebSphere Portal Express. The Solution Installer automatically uploads any `.zip` files that are found in this directory to the `dav:fs-type1/iwidgets/` directory. Extra work is necessary to have the iWidget definitions that are registered with WebSphere Portal Express. The installer must know about the widget definition files to register with WebSphere Portal. A properties file called `iwidgets.properties` must be included in the `iwidgets` directory in the PAA. The properties are generated with the `.zip` file name that contains the definition file as the name of the property. It also supplies a comma-separated list of definition files that are contained in this file as the value.

layout-templates
Place `.zip` files that contain layout-templates in this directory. When such files are detected by the installer, code is generated to automatically upload such content to the `dav:fs-type1/layout-templates/` directory in the WebDAV file store.

skins Place any `.zip` files that contains skins not specific to a theme in this directory. The content is automatically uploaded to the `dav:fs-type1/skins/` directory in the WebDAV file store.

themes Place `.zip` files that contain static theme content in this directory. They are automatically uploaded to the `dav:fs-type1/themes/` directory in the WebDAV file store.

Note: If your theme contains dynamic content, include them in a WAR file that is deployed at run time.

The Solution Installer uploads WebDAV content with the `dav:fs-type1/*.* WebDAV` entry point. The themes and skins are

not automatically made available through the themelist or skinlist entry points. To ensure that they are available through the administration pages, an XMLAccess script must be created to register the resources with WebSphere Portal Express. The context root, where the content of the individual `.zip` files is installed, is set in the following manner:

- The root directory is contained within the `.zip` file. For example, all the content is enclosed within a directory. Then, it is appended to the `TargetURI`. For example, a `.zip` file in the `componentName/content/webdav/themes` directory with a root directory of sample would result in `TargetURI 'dav:fs-type1/themes/sample/'`.
- No root directory is contained in the `.zip` file. All the items are in the parent level. Then, the name of the `.zip` file minus the '`.zip`' suffix is used. For example, a `.zip` file with the name `sample1.zip` would result in `TargetURI 'dav:fs-type1/themes/sample1/'`. The upload task for the `.zip` files to the WebDAV file system is set to replace the current directory with the new content. However, the Solution Installer alters this function to merge the content by default instead. When the **`UpdateMode`** parameter is set to `replace`, which is the default, the upload replaces all content found at the `TargetURI`. For example, if you upload a `.zip` file to `dav:fs-type1/themes/`, it does not replace the equivalent content that is stored in the directory. It replaces everything in this directory. Therefore, it was decided that it would be better to have the content that is set to merge with the existing content as the default behavior. If you do require **`UpdateMode`** set to `replace`, then you must add a properties file to the directory. For example, if you want to replace all the themes, you would place the `webdav.properties` file in the `componentName/content/webdav/themes` directory. There is just one property available in this file: `webdav.replace=list of .zip files`. The value of the `webdav.replace` property is a comma-separated list of files that tells the Solution Installer which files must be uploaded.

`xmlaccess`
Stores any component level XML access scripts. This directory differs from the `component/config/templates` directory that stores scripts to be called by custom Ant tasks. Any scripts that must be run by default, are placed in the `content/xmlaccess` directory. This directory has two subdirectories to aid in the distinction between installation and uninstall scripts. The scripts that are required for installation are in the `/content/xmlaccess/install` directory and the uninstall scripts are placed in the `/content/xmlaccess/uninstall` directory. The type of scripts in the `component/content/xmlaccess` directory can include scripts to register a theme or create pages and assign portlets. Other examples are scripts to create a set of users and groups or a credential slot in the WebSphere Portal Express credential vault. All the scripts that are found by Solution Installer in these directories are automatically run.

`database`
This directory contains any database scripts for creating tables and pre-populating the tables with any relevant data. Solution Installer

can generate Ant tasks to create the relevant configuration settings on the underlying WebSphere Application Server. For details of the required properties, go to "Database properties for the Solution Installer" on page 3428.

There are two directories in the `database` directory in the PAA file: `install` and `uninstall`. The PAA file developer must place the following scripts in these directories:

- `.ddl` and `.sql` scripts to create and populate tables in the `install` directory
- `.ddl` and `.sql` scripts for dropping tables in the `uninstall` directory

Where multiple scripts are required, include an `order.properties` file in the appropriate directory to specify the correct order for installation.

If you are providing setup scripts for different database types in a single PAA file, there is an extra step. Run this step during the PAA creation phase. For each database type, a properties file must be added to the `components/componentName/content/database/install` directory.

- For Derby scripts, call the `scripts.derby.properties` file.
- For DB2 scripts, call the `scripts.db2.properties` file.
- For Oracle scripts, call the `scripts.oracle.properties` file.
- For SQL Server scripts, call the `scripts.sql.properties` file.

Add a comma-separated list of scripts to the properties file in the order that they must be run for a specific database type. The Solution Installer determines at run time, which set of scripts to run for the database type requested.

pzn PZN-related artifacts, such as JAR files that contain business rules and personalization `.nodes` files are in this directory. The Solution installer automatically copies any JAR files that contain custom Java classes to the correct location under the `profile` directory and upload any `.nodes` files to the server.

installableApps

The `installableApps` directory is where any artifacts that must be installed to WebSphere Portal Express or directly to the application server are to be stored. Solution Installer copies the relevant files across to the *wp_profile_root*`/installableApps` directory automatically when the default implementation tasks are used. The artifacts are stored in the PAA file in subdirectories of the `installableApps` directory. The subdirectories are based on their resource type to allow for default code to be easily generated to manage installation and deployment of artifacts. However, when the artifacts are copied to the *wp_profile_root*`/installableApps` directory, it is just the content of the subdirectories that is copied and not the directories themselves. The following is a list of currently supported resource sub directories:

ear Contains any EAR files that must be deployed to the application server. Wrap `.war` files that do not contain any portlets and must be installed to the application server in an EAR file. The reason for this is that the default scripts used to deploy artifacts require specific information to run the installation. For example, the display name and context root for the application. This information

can be found in the application.xml file of an EAR file. The context root information would not be available with just a WAR file. An example of a WAR file that must be wrapped in this way is a theme.war file. If you are providing a custom Ant deployment script, it is unnecessary because you can provide the required information in the script or a properties file.

EAR files are deployed automatically to the server. However, if a WAR file that contains a portlet is wrapped inside of an EAR file, the developer must supply an extra script to register the portlets with WebSphere Portal Express. It is the responsibility of the developer to ensure that the portlets are registered in the correct manner.

portlets

Place any WAR files that contain JSR portlets into the portlets directory. IBM legacy portlets are not handled automatically by the installer. They require custom code to install and must not be placed in this directory. The reason for separating these WAR files from those files that contain servlets or other application types is due to the installation method required. Those files that contain JSR portlets are typically installed and deployed with an XML access script. Those files that need to be installed directly to the application server are deployed with a ConfigEngine Ant task or with a wsadmin script. The Solution Installer can then read the portlets directory and install the WAR files automatically and does not need to worry that it might encounter a non-portlet WAR. However, if there is extra configuration setup for a portlet, then overwrite the default installation task with an Ant task with a custom XMLAccess script to do the installation. Add a <SCU> element to the sdd.xml file for the **deploy-portlets-applySIFeaturePack** extension point and add an Ant task that implements this extension point in the config/includes directory. This Ant task must call the **XMLAccess** task to run any supplied XMLAccess script against the portal server.

The automatic installation uses the unique ID from the <portlet-app> element that is found in the *warfile*/WEB-INF/portlet.xml file, and the location of the WAR file itself to enable the deployment. If the unique ID is not available, the name of the WAR file is used. The WAR file is automatically copied over to the *wp_profile_root*/installableApps directory. The following is a sample XMLAccess script that shows how the information is used to drive the installation.

```
<?xml version="1.0" encoding="UTF-8"?>
<request xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.0.xsd" type="update"
  create-oids="true">
  <portal action="locate">
   <web-app action="update" active="true"
    uid="portletXmlUniqueId.webmod">
    <url>file:///$profile_dir$/installableApps/warfile.war</url>
   </web-app>
  </portal>
</request>
```

WAR files that are placed at this location also have unique names that are automatically generated for the individual portlets during installation. The generation of unique name values is based on the

scheme *componentName.portletName*. The following is a sample XMLAccess script that shows how the unique name values are specified:

```
<?xml version="1.0" encoding="UTF-8"?>
<request xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.0.xsd" type="update" create-oids
  <portal action="locate">
   <web-app action="locate" domain="rel" objectid="WebAppID.webmod"
uid="WebAppId.webmod">
   <portlet-app action="update" domain="rel" name="PortletAppId"
uid="PortletAppId">
   <portlet action="update" domain="rel" name="PortletName"
uniquename="componentName.PortletName"/>
   </portlet-app>
  </web-app>
 </portal>
</request>
```

Prefix the component name to the WAR file name with the *componentName.WARname*.war scheme. Although not essential, it allows the user to easily track which WAR files were installed with the Solution Installer.

war This directory contains any WAR files that the developer does not want to automatically install. The developer extends the extension point with a custom Ant task to install the artifacts. An example is if the WAR file contains legacy IBM portlets. Alternatively, the WAR file does not contain any portlets and the developer does not want to wrap it in an EAR. Also, extra customization steps might be required for the configuration of the deployed artifacts. Therefore, a custom script is necessary. The installer generally ignores this directory apart from copying the files to the *wp_profile_root*/installableApps directory and then starting any custom code that is provided to handle these artifacts.

zip This directory contains any .zip file content that you must include for a component. For example, you might have some artifacts that must be copied and installed to a server that currently is not supported by Solution Installer. Solution Installer does not automatically process the content of this directory. The individual files remain in their compressed state after the PAA file is extracted. Instead, any processing of these files is completed by either custom Ant tasks that are provided by the developer or through manual steps.

shared When you deploy an application to WebSphere Portal Express, often extra shared libraries are required for the application to function correctly. These libraries can be at different levels of scope for the application. If stored in the WAR file itself, then only classes within that WAR file are able to access the library files. The second situation is when a set of libraries are solution-specific. That is, the classes in the shared library are available only to the overall solution. The WAR file option is not appropriate as it might mean a number of separate applications that work together as a larger solution, all requiring access to the library. The third level of scope is global, meaning that many applications on the server can access these classes. The first situation where the library JAR files are stored in the WAR file is out of scope for this document. However, the ability to handle the other two situations is provided by the Solution Installer.

For globally available JAR files, place the JAR files into either the component/shared/app or component/shared/ext directories. These files are not copied to the equivalent directories under the *PortalServer_root* directory. Instead, a task is run to register all the JAR files found in the shared/app and shared/ext directories of the components. When objects are found in these directories, they are registered directly with a Solution Installer-specific set of shared libraries that are inside the profile. This action makes the libraries profile-specific thus different versions of the same files might be installed to different profiles.

JAR files that are registered in the component/shared/app directory are registered in the Solution Installer-specific shared.app.jar file. Similarly, for the .jar files in componentName/shared/ext directory, these files are registered with a profile-specific shared.ext.jar file. These files can be found under the *wp_profile_root*/PortalServer/solutionInstaller directory sub tree. The specific library 'SiSharedLib' is registered at the cell or node level of the profile. A reference is then added to the class path of the application server to ensure that the files are available at run time. Only the shared.app.jar file is loaded automatically by the SiSharedLib library. When the registration of these files is finished, a server restart is required to reload the libraries and make the classes available on the class path. When complete, the library-specific classes are now available globally to that server for an application to access.

To allow a library scope that is limited to a specific solution, place the relevant JAR files in the component/shared/common directory. Files in this directory are not copied to a location inside of the profile directory. Instead, a shared library for this component that points to this directory are added to the WebSphere Application Server. This shared library then must be associated with either the application or made available on the server-wide class path so it is available to all applications. A properties file called shared-library.properties is in the component/shared/common directory. This file contains information on the scope to which the shared library must be registered. It also provides information on any required class loader properties, such as class loading precedence. The following properties are available in the shared-library.properties file:

```
# set whether the library should be at the server scope or application scope
# Can have the following values:
# cell, cluster, node, server
library-scope=server
# specify whether the library should be added to the server class path
# or associated with a specific application.
Library-ref=application
# Set the name of the application(s) to which the library is to be associated.
# name of application(s) found in the integrated administration
# console/applications/enterprise applications.
applicationName=# your application name
# Set class loading preference, options are either
# 'PARENT_FIRST' or 'PARENT_LAST'.
classLoadingMode=PARENT_FIRST
```

If multiple applications are associated with the library, specify a comma-separated list of application name; for example: **applicationName=**AppName1,AppName2,AppName3, where AppName1,AppName2,AppName3 represent the applications to which the library is associated.

template

The template directory is where a developer can place files to create a website template. The template is based on one or more of the components

that are supplied in the PAA distribution. One or more subdirectories might be contained within the `template` directory, one for each template that you want to provide. The default directory name is reserved by Solution Installer as the content of this directory is always run.

The default directory and any subsequent template directories have their content split into two further subdirectories to aid in the distinction between installation and uninstallation scripts. The installation scripts are in the `/component/template/template_Name/install` directory and the uninstallation scripts are placed in the `/component/template/template_Name/uninstall` directory. The scripts that are contained in the default directory are always run. Therefore, if you are offering multiple templates and do not want one to be installed by default, leave this directory structure empty. In general the content of the default, or template-specific directories, include XMLAccess scripts to create pages, put portlets on the pages, and create users. That is, any task that is site-related and not covered by the other directories or components. For example, do not place an XMLAccess script to install a WAR file here. Instead, if this file is required, place it in the `component/config/templates` directory if a custom Ant task is required or in the `component/content/xmlaccess/install` directory otherwise.

To use a different template to the default, set the **templateName** property in the `componentName.properties` file. The value of the **templateName** property should reflect the name of the template to be used. For example: **templateName=**`template2`.

Note: Store any site-wide template-related artifacts in a single component separate from the components on which they depend. This placement allows the overall site presentation to be separate from the underlying technologies they surface and it makes it easier for the installer to handle updates in the future. It also makes it much easier to manage the provision of multiple site templates in a single PAA distribution. Store sample or demonstration pages for a single component local to that component.

version

The `component/version` directory contains a single `.component` file with the application version information. The file contains build date, build version, name, and spec-version. This file is what tells the ConfigEngine the version of the application. Knowledge of the version that is already installed is necessary to facilitate updates to an application. The following is an example of a `.component` file:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE component PUBLIC "componentId" "component.dtd">
<component build-date="6/10/14" build-version="20140610-1200" name="components/sample1" sp
```

Installation tasks

Solution installer supports the auto generation of deployment tasks for specific resource types. The creation of these tasks occurs during the installation or registration phase for the PAA file with the Solution Installer and the ConfigEngine.

At this stage, analysis is done on the PAA content after expansion to the *wp_profile_root*`/paa` directory to determine which resources are included. When complete, the Solution Installer generates the code to deploy the artifacts. The developer might decide that the default code is not robust enough to handle the deployment of their artifacts. Or the developer might decide that extra configuration that is not covered by the Solution Installer is required. The

developer can overwrite the code generation step for the specific resources. The developer must provide an implementation task for the extension point that is mapped to that resource directory. For information, read “Developing advanced PAA file applications” on page 3431.

The Solution installer determines what resources are available and looks up the mapping between individual resource types and their assigned extension point. Then, it checks the `config/includes` directory of the current component to determine whether a task exists for the extension point that is related to the resource type. If one is found, then it is assumed that all resources in the directory are covered by the task. No attempt is made to auto-generate code to deploy those resources. No code inspection is done on the implementation task to establish if all resources are covered.

The naming scheme for tasks is important for recognizing which tasks implement extension points. The naming scheme is `'action-' + '%extension point' + '-componentname'`.

The extension points that are started by the Solution Installer differ from the portal core extension points. They have the suffix 'applySIFeaturePack' or 'removeSIFeaturePack' appended to the end of the core extension point name.

Take for example two ear files that are provided in the `components/sample1/installableApps/ear` directory. The Solution Installer checks if a task called **`action-create-ear-applySIFeaturePack-components/sample1`** exists. The `'action-'` *`extension point name`* + *`componentName`*`'` is the pattern that is used to determine that a task implements an extension point. If such a task is found, then the Solution Installer adds the task name to the list of implemented extension points and moves on to the next resource type. Otherwise, it implements a task to cover the deployment of both ear files.

The ConfigEngine runs the installation of the set of components by extension point and controls the order in which they are run. For example, all component tasks that implement the **`create-ear-applySIFeaturePack`** extension point are run together. Portlet deployments are done at the same time. If you need an artifact of a component that is installed before you install another, then you must create a dependency in the `sdd.xml` file. The dependency works on an extension point basis because you can specify that artifacts covered by that extension point can have an order. For example, you can specify that the EAR file in component1 is installed before the EAR file in component2. Read “Component level `sdd.xml` file overview” on page 3424 for information.

Any extra Ant tasks that are found in the `config/includes` directory that do not comply with the naming scheme for extension point tasks are not run. Instead, these tasks must be called directly by extension point tasks or they are not run.

In earlier versions of the Solution Installer, it was necessary to add an `SCU` element to the component level `sdd.xml` file to register any developer-provided extension point implementation tasks. However, it is no longer necessary because `SCU` elements are now added for all recognized extension point tasks.

Developers can also implement extra configuration extension points that are not directly related to deploying resources in the PAA file. An example is a task to create a resource environment custom property. Read the “Developing advanced PAA file applications” on page 3431 for information. Tasks to implement these extension points can also be added to an XML file in the `config/includes`

directory. When they follow the naming scheme pattern, they are registered in the component level sdd.xml file with the SCU elements and automatically run at run time. In this way, the Solution Installer can handle both auto-generated and developer-provided code.

Order of installation of scripts and artifacts

Often the order that scripts are run or artifacts that are deployed is important for the success of the installation. This information is both true for artifacts or scripts in a single component and the order in which scripts contained in multiple components are installed.

At the component level all the scripts or artifacts in a directory are installed or deployed by a single task with one of the extension points. Therefore, if there are multiple scripts or resources, the Solution installer needs a mechanism to determine the correct order to run the scripts. If a custom task is provided, then there is no problem as it is assumed that the task runs all the scripts in the expected order. However, an issue arises when the Solution Installer auto-generated code is used to handle the deployment of the resources.

To solve this issue, a properties file called order.properties can be added to any of the directories within the component hierarchy that contain artifacts that must be run or installed. For example, an order.properties file can be added to the components/componentN/content/xmlaccess/install directory to handle the order of two or more XMLAccess scripts. This file contains a comma-separated list of the file names in the correct order that they are to be deployed.

When the **install-paa** task is run for the Portal Application Archive (PAA) file, a new Ant task is created. It runs the scripts in the order that is outlined in the order.properties file. If no order.properties file exists, the Solution Installer works on the assumption that the order of installation is not important.

The previous solution covers multiple artifacts for an extension point in a component. However, multiple components might require the same extension point and might have a dependency on the resources of one of these components to be deployed or configured before deployment. The strategy that is used by ConfigEngine and therefore Solution Installer is to collect all the different implementations of an extension point in the PAA file and run them consecutively. For example, all implementations of the create-ear-applySIFeaturePack extension point are run before you move on to the next extension point type. The Solution Installer determines the order in which the extension points are run. Each extension point is run in a preset order that is based on the type of function required. Where there are multiple implementations for an extension point across components, it might be necessary to have an order that is placed on when they are started.

These dependencies between the component extension points are set in the component level sdd.xml file. Add a requirements element to the SCU element for the extension point. This requirements element must point to the component on which it depends. Read "Component level sdd.xml file overview" on page 3424 for information.

Starting with version 8.5, the dependencies can be automatically created between extension points of different components with the **install-paa** task. Add an order.properties file to the /components directory of a PAA file. This file contains a comma-separated list of components in the order in which they are to be installed. The components that can be run outside of this order do not need to be

added to the list. After the Solution Installer creates default code and adds the SCU elements for the extension points, it analyses the components in the order.properties file. It sets the requirements elements on the relevant SCU elements to add dependencies between the shared extension points of these components.

Note: The full component name must be used in the order.properties file, Take for example a PAA file with two components components/component1 and components/component2. The line in the order.properties file is components/component1,components/component2.

Important: The Solution Installer adds only a requirements element where one does not exist. Therefore, it does not overwrite any developer-provided settings.

For the removal of the resources from the portal with the **remove-paa** task, the order that the Solution Installer uses is the reverse of what is found in the order.properties file.

The sdd.xml file

The sdd.xml files perform a number of different roles in the PAA file structure. They allow the developer to control, with ConfigEngine extension points, how an application is installed. They also inform the ConfigEngine of the type of installation to be processed. It also determines, in terms of the ConfigEngine, where the deployable files are stored after registration with the ConfigEngine.

The ConfigEngine takes a granular approach to the installation and deployment of an application. The sdd.xml file controls the granularity to the ConfigEngine. There are three different levels of installation:

- Offering level: Large applications such as WebSphere Portal Express are offering level deployments.
- Assembly level: An assembly is usually a group of applications that can be stand-alone or form the basis of a much larger application. An example of an assembly is the ap or base directories under WebSphere Portal Express.
- Component level: The component level is the finest level of granularity. A component can be an application or can be one of many small applications that make up a much larger application.

An assembly is made up of one or more components and an offering is made up of one or more assemblies.

The Solution Installer includes all the PAA files in a higher-level PAA offering. The directory structure for the Solution Installer and any offering setup details that are applied to the ConfigEngine are processed automatically. It is easier to manage the installation of the different components when they are installed under one hierarchy.

Each PAA file is treated as a separate assembly. The top-level sdd.xml contains installation information for the ConfigEngine relating to the assembly and its components. A PAA file that contains an update has the same assembly name in its sdd.xml file. The Solution Installer checks that the assembly exists before it adds each new component or update to the current assembly.

The assembly can be made up of one or more components, each requiring a component level sdd.xml file that can be automatically generated. The number of components depends on how the developer structures the application to be installed. Every artifact can potentially be included in a single component,

although it might not be the most practical approach. However, you might want to separate larger applications into multiple components that contain related artifacts to facilitate reuse. Because a component might be required by multiple PAA distributions, this approach allows it to be reused with little extra work for the developer.

"The assembly level sdd.xml file"
When you create a Portal Application Archive (PAA) file, it might be necessary to add an assembly level sdd.xml file. This file registers the PAA content with the ConfigEngine.

"Component level sdd.xml file overview" on page 3424
For most installation scenarios, generation of the component level sdd.xml file is automated. However, there are still occasions where it might be necessary to create it manually. The component level sdd.xml file contains the information on how to install the artifacts of the component. It lists the extension points that need to be processed. The extension points ensure that the resources can be installed and configured on the server. Ant tasks complete the deployment and configuration work.

Related concepts:

"The assembly level sdd.xml file"
When you create a Portal Application Archive (PAA) file, it might be necessary to add an assembly level sdd.xml file. This file registers the PAA content with the ConfigEngine.

The assembly level sdd.xml file:

When you create a Portal Application Archive (PAA) file, it might be necessary to add an assembly level sdd.xml file. This file registers the PAA content with the ConfigEngine.

The information in the sdd.xml file informs the Solution Installer about the name and type of the application to install. It also includes information about the versions of IBM WebSphere Portal Express to which installation of the PAA content is compatible. In addition, the sdd.xml tells Solution Installer the names and locations of the components to be installed. The different elements of the assembly level sdd.xml file are described in terms of how they are required for the Solution Installer. You can edit a sample sdd.xml file with the required information to ensure that the sdd.xml file is complete and accurate.

Starting with Version 8.5, a basic version of the assembly level sdd.xml file can be automatically generated. Solution installer uses the name of the root directory in the PAA file as the assembly name and can add any components that are found in the /components/ directory. Each component is added based on the names of the sub directories that are found in the /components directory. Each components/ subdirectory is recognized as a separate component. However, there might be cases where a developer does not want to have a component included in the assembly level sdd.xml file. For example, an extra component that contains files that should not be deployed is included. Then, the developer might need a mechanism to restrict the list of components. You can place a limitation on the components that you want to include. Create an order.properties file in the components directory. Populate it with a comma-separated list of the components in the order in which they should be installed. If there is a components/order.properties file, the Solution Installer restricts the list of components added to the sdd.xml file to only those components that are listed in the order.properties file. This order has other meanings for automatically creating dependencies between the components so it is important to ensure that the components can be successfully installed following

this order. See the section 'Order of installation of scripts and artifacts' for more details on how the ordering of artifacts is handled. The mechanism in which the dependencies are created is described in *The component level `sdd.xml` file* section in the advanced development documentation.

Although the assembly level `sdd.xml` file can be auto-generated, there are many circumstances where this option is not a viable solution. For example, there are restrictions on server versions on which the PAA file can be installed. It is necessary to include a `<ServerVersionDependencies>` element to illustrate the versions. This information is not handled automatically by the Solution Installer so you must generate the file manually. If you added a `/components/order.properties` file and do not want to limit the list to only those components in the `order.properties` file, add the additional `<containedPackage>` elements for each component to the file.

Note: In the `PortalServer_root/doc/paa-samples` directory, some sample files illustrate the PAA directory structure. You can use the assembly level `sdd.xml` file of the `sample1.paa` file as an example. This file can be used as a starting point for developers to create their own working `sdd.xml` files. This sample PAA file is an installable application. It can be registered with the ConfigEngine. However, it is not a working PAA file because there are no installable resources. The structure is provided so that a developer can use it to create their own PAA files.

The following is an example of an assembly level `sdd.xml` file:

```
<?xml version="1.0" encoding="UTF-8"?>

h<iudd:iudd
 xmlns:iudd="http://www.ibm.com/xmlns/prod/autonomic/solutioninstall/IUDD"
 xmlns:iurtype="http://www.ibm.com/xmlns/prod/autonomic/resourcemodel/IU/resourcetypes"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:OSRT="http://www.ibm.com/xmlns/prod/autonomic/resourcemodel/OS/resourcetypes"
 xsi:schemaLocation="http://www.ibm.com/xmlns/prod/autonomic/solutioninstall/IUDD ../iudd/iudd.xsd "
 schemaVersion="2.0.0" buildID="MySoln-IUDD-1.0" buildDate="2006-01-19T12:00:00">

 <packageIdentity contentType="Assembly">
  <name>sample_paa</name>
  <version>8.5.0.0</version>
  <displayName key="d0001" default="sample_paa" />
  <manufacturer>
   <displayName key="DU_01" default="IBM" />
  </manufacturer>
 </packageIdentity>

 <topology>
  <resource type="OSRT:OperatingSystem" id="OS" />
 </topology>

 <content xsi:type="iudd:RootIUContent">
  <rootIU id="sample_paa" targetRef="OS">
   <identity>
    <name>sample_paa</name>
   </identity>

   <containedPackage id="components/componentN" pathname="components/componentN/sdd.xml" />


   <serverVersionDependency name="PortalServer" lowerVersion="6.0.0.0" higherVersion="8.5.0.0" versi

  </rootIU>
 </content>
</iudd:iudd>
```

Inside the root <iudd:iudd> element, the following three subelements are required for installation with the Solution Installer:

The <packageIdentity> element

The first element is the <packageIdentity> element; for example:

```
<packageIdentity contentType="Assembly">
  <name>sample_paa</name>
  <version>8.5.0.0</version>
  <displayName key="d0001" default="sample_paa" />
  <manufacturer>
   <displayName key="DU_01" default="IBM" />
  </manufacturer>
 </packageIdentity>
```

The <packageIdentity> element informs the Solution Installer and the ConfigEngine of the type of application to be installed. In this case, the **contentType** attribute equals Assembly. An assembly is a grouping of one or more components. The Solution Installer considers each PAA file as an assembly, even though it might contain only one component. Do not edit this attribute.

The <packageIdentity> element contains a number of subelements that provide information about the content. The name and version elements need to be altered to suit the application you are installing. The <displayName> element can also be edited to include the assembly name, but is not required.

Important: The name element that is provided for an assembly must match the name of the PAA file root directory. For example, a name of sample_paa requires the root directory to also be called sample_paa.

The <topology> element

The next subelement is the <topology> element. This element does not require any alterations. For example:

```
<topology>
  <resource type="OSRT:OperatingSystem" id="OS" />
 </topology>
```

The <content> element

The final element block is the <content> element. This element provides the Solution Installer and the ConfigEngine with the information about what to install and on which servers. The <rootIU> element is where the actual information is included. Set the 'ID' attribute of the <rootIU> element to the name of the application as used in the <name> element of the <packageIdentity> element.

```
<content xsi:type="iudd:RootIUContent">
  <rootIU id="sample_paa" targetRef="OS">
  <identity>
    <name>sample_paa</name>
   </identity>

   <containedPackage id="components/componentN" pathname="components/componentN/sdd.xml" /


   <serverVersionDependency name="PortalServer" lowerVersion="6.0.0.0" higherVersion="8.5.

  </rootIU>
 </content>
```

The <identity> element contains the <name> subelement. Set this element to the name of the assembly found in the <packageIdentity> name element.

Although there is only one <containedPackage> element in this example, there can be a number of these elements. One for each component included in the PAA file. This element allows for ConfigEngine to register the components and informs it on where to find the component level sdd.xml file. The Solution Installer can auto-generate the component level sdd.xml file during the **install-paa** command. The developer does not need to add this file for a basic PAA file. However, when the developer needs to provide specific component dependencies outside of an order.properties file, then the component level sdd.xml file must be provided.

There are two attributes for the <containedPackage> element. The ID must equal the path of the component relative to the assembly level sdd.xml file. For example, as the PAA file puts all components in the components directory, a component name needs to also include the components/. For example, a component that is called 'componentN' would have the id="components/componentN". The path name contains the path from the assembly sdd.xml file to the component level sdd.xml file. Continuing with the previous example, the pathname element is pathname="components/componentN/sdd.xml".

The final element is the <serverVersionDependency> element. This element informs the Solution Installer of the WebSphere Portal Express versions on which the content can be installed. Not all the attributes that are shown are required. Go to the *Checking server dependency* section for more details.

Related concepts:

"The sdd.xml file" on page 3420
The sdd.xml files perform a number of different roles in the PAA file structure. They allow the developer to control, with ConfigEngine extension points, how an application is installed. They also inform the ConfigEngine of the type of installation to be processed. It also determines, in terms of the ConfigEngine, where the deployable files are stored after registration with the ConfigEngine.

Component level sdd.xml file overview:

For most installation scenarios, generation of the component level sdd.xml file is automated. However, there are still occasions where it might be necessary to create it manually. The component level sdd.xml file contains the information on how to install the artifacts of the component. It lists the extension points that need to be processed. The extension points ensure that the resources can be installed and configured on the server. Ant tasks complete the deployment and configuration work.

Sample sdd.xml file

```
<iudd:iudd
 xmlns:iudd="http://www.ibm.com/xmlns/prod/autonomic/solutioninstall/IUDD"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:OSRT="http://www.ibm.com/xmlns/prod/autonomic/resourcemodel/OS/resourcetypes"
 xmlns:OSAT="http://www.ibm.com/xmlns/prod/autonomic/resourcemodel/OS/artifacttypes"
 xmlns:J2EERT="http://www.ibm.com/xmlns/prod/autonomic/resourcemodel/J2EE/resourcetypes"
 xsi:schemaLocation="http://www.ibm.com/xmlns/prod/autonomic/solutioninstall/IUDD

../iudd/iudd.xsd"
 schemaVersion="2.0.0" buildID="112220" buildDate="2006-01-19T12:00:00">

 <packageIdentity contentType="Component">
```

```
  <name>components/componentN</name>
  <version>8.0.0.0</version>
  <displayName key="d0001" default="components/componentN" />
  <manufacturer>
   <displayName key="AC_01" default="IBM" />
  </manufacturer>
 </packageIdentity>

 <topology>
  <resource type="OSRT:OperatingSystem" id="OS" />
 </topology>

 <content xsi:type="iudd:RootIUContent">
  <rootIU id="components/componentN">
   <variables>
    <parameters>
     <parameter name="installLocation"

defaultValue="/usr/dummy.offr.1" />
    <parameter name="FunctionalArea" defaultValue="featurepackSI" />
    </parameters>
   </variables>

   <SCU id="deploy-portlets-applySIFeaturePack" targetRef="OS">
    <identity>
     <name>Do Configuration Task</name>
     <version>8.0.0.0</version>
     <displayName key="keyInBundle"
      default="Executes Configuration for this component" />
     <description key="keyInBundle"
      default="This section runs configuration for this component" />
    </identity>
    <unit>
     <configArtifact type="ConfigEngine">
      <parameters>
       <parameter name="targetName" value="deploy-portlets-applySIFeaturePack" />
      </parameters>
     </configArtifact>
    </unit>
   </SCU>
   <SCU id="remove-portlets-applySIFeaturePack" targetRef="OS">
    <identity>
     <name>Do Configuration Task</name>
     <version>1.0.0.0</version>
     <displayName key="keyInBundle"
      default="Executes Configuration for this component" />
     <description key="keyInBundle"
      default="This section runs configuration for this component" />
    </identity>
    <unit>
     <configArtifact type="ConfigEngine">
      <parameters>
       <parameter name="targetName" value="remove-portlets-applySIFeaturePack" />
      </parameters>
     </configArtifact>
    </unit>
   </SCU>
  </rootIU>
 </content>

</iudd:iudd>
```

See "The component level sdd.xml file" on page 3432 for information about editing the sdd.xml file.

Syntax information for component names

The syntax for naming the component is important. There are dependencies within the PAA file that determine part of the component name that is registered in the `sdd.xml` file. You are free to name your components as you like; however, the component names must be unique. The ConfigEngine and by extension Solution Installer impose a restriction that a component name not clash with an existing component in the ConfigEngine registry. This restriction is not limited to an assembly, but is registry wide. Therefore, if a component has the same name as an existing component, it is not automatically installed. If the user wants to install this new component, they can use the **update-paa-components** task. This task removes the existing version and installs the new component as part of the new assembly.

When you register the names in the `sdd.xml` files, you must adhere to the following syntax: `components/componentName`. This syntax is necessary for the Solution Installer to resolve the component name. It must contain the full path from the directory that contains the assembly level `sdd.xml` file, usually the root directory, to the component level `sdd.xml` file.

Looking at the *PortalServer_root*`/doc/paa-samples/sample1.paa` file, it contains one component in the components directory called '`Sample1`'. Therefore, The component name would appear in the `sdd.xml` file as follows: `components/Sample1`. It is the full path relative to the directory that contains the assembly level `sdd.xml` file.

The Solution Installer follows the feature pack approach that is provided by the ConfigEngine to install the individual components. The advantage of this approach is that the ConfigEngine manages the installation order of the individual components. Using the feature pack approach ensures a distinction between portal core extension points and those points that are used to install the PAA content. This approach reduces the risk of a user accidentally removing portal core components when you run a remove extension point.

The ConfigEngine runs the installation of the set of components by extension point and controls the order in which they are run. For example, all EAR files are installed together; all portlet deployments are done at the same time. If there is a need to have an artifact of a component that is installed before another, then you need to create a dependency in the `sdd.xml` file. The dependency works on an extension point basis in that you can specify that artifacts covered by that extension point can have an order. For example, you can specify that the EAR file in component1 is installed before the EAR file in component2.

Note: The value of the **FunctionalArea** parameter is set to featurepackSI. This value is important so the Solution Installer can determine that the components are to be installed with the Solution Installer specific feature pack. See the `sdd.xml` file example at the beginning of this file for how to implement the feature pack approach.

In the sample `sdd.xml` file, there are two `SCU` elements. The first one shows how to register the extension point to deploy a portlet and the second one demonstrates how to remove the portlets.

```
<SCU id="deploy-portlets-applySIFeaturePack" targetRef="OS">
 <identity>
 <name>Do Configuration Task</name>
 <version>1.0.0.0</version>
  <displayName key="keyInBundle" default="Executes Configuration for this component" />
  <description key="keyInBundle" default="This section runs configuration for this
   component" />
```

```
 </identity>
 <unit>
 <configArtifact type="ConfigEngine">
 <parameters>
  <parameter name="targetName" value="deploy-portlets-applySIFeaturePack" />
  </parameters>
  </configArtifact>
 </unit>
  <!–register a dependency on a previous component-->
  <requirements>
   <requirement name=" deploy-portlets-applySIFeaturePack">
   <alternative name="dependentComponent"/>
  </requirement>
  </requirements>
</SCU>
```

The `SCU` element example notifies the Solution Installer about the task you want to run. The extension point in this case is called **`deploy-portlets-applySIFeaturePack`**. It deviates from the core extension points for portal by adding 'applySIFeaturePack' to the end of the extension point. With an uninstall task, 'removeSIFeaturePack' is appended instead. It allows the Solution Installer to distinguish between installation and uninstall tasks. Append these strings to the extension point names to create a definite distinction between core extension points and those points included for the Solution Installer to handle a PAA distribution. The Solution Installer expects and runs only extension points that conform to this format.

To provide an implementation task for an extension point, read "Installation tasks" on page 3417 for information.

Property files

Two different types of properties are contained within a PAA distribution. They can be classified as being either user editable that is requiring user input, for example a database URL, or as developer-provided settings required for a component to function. However, for user convenience, a separation between user properties from those settings that are supplied by the developer is advised.

In general custom Ant tasks that are provided by the developer handle such properties. The exception is when properties for connecting to an external database are included. The other exception is a selection property for cases when the user wants to install a template other than the default template in the PAA templates directory is required. However, there are cases where properties required by the Solution Installer might also need to be set. In this case, they need to be included with user editable properties.

Users might need to edit such content before they run the installation. Therefore, it makes sense to consolidate such properties in the one place instead of having the user edit multiple files in different locations. However, there is a trade-off with keeping all the properties required by a component together to allow for component reuse. Therefore, a number of approaches to storing properties in the PAA file are allowed.

User editable properties can be defined in the properties file specific to a component, `component_name.properties`, which can be found in the top-level directory of a component. The naming scheme of this file is important because the Solution Installer loads it automatically. When the PAA file is expanded, users are free to edit them as needed. They can then rerun any ConfigEngine function that is

associated with the component to take account of these new values. The values in this file are the defaults for the properties and can be overwritten by parent properties files.

There are two ways in which the properties in the component level properties files can be overwritten by parent properties. The properties files that contain user editable properties for the components of the PAA assembly can be consolidated into one properties file. They are placed in the top-level directory of the PAA file. It is called `assembly_name.properties`. The user can edit this file to set any values that are required for the installation to ConfigEngine and IBM WebSphere Portal Express to run smoothly. When the user starts the installation process for the ConfigEngine, these properties are read first. The values in the default properties files are redundant because the Ant properties cannot be overwritten, unless they are out of scope.

Alternatively, the user editable properties can be passed in on the command line as parameters to the Solution Installer tasks. Individual property **`name=value`** pairs can be typed on the command line with a **`-D`** prefix to inform solution installer and the ConfigEngine that they are to be treated as Ant properties. Alternatively a properties file containing all of the user editable properties can be passed in on the command line with the Ant **`-propertyfile`** parameter.

The Solution Installer checks the command line to establish if properties were passed in when the task was started. If so, the properties are loaded by the installer and the values of the properties set are the ones that are used during the deployment to WebSphere Portal Express. Similarly, if a properties file is not received on the command line, then it looks in the highest level directory of the PAA distribution for such files. When found, these values are loaded as the values for the properties. If there are extra properties in the assembly level file, then these properties are also read. If no properties file is found, values from the component level property files are used. Otherwise, it is assumed that no user editable properties are required and the installation continues.

Property files that contain developer-provided settings must be at the component level. To aid reuse of components across multiple PAA files, keep required settings local to a component.

When you decide on property names, take great care. A limitation of Ant means that when a property is created in a run of the scripts, it cannot be overwritten. Therefore, if you install multiple components that have a setting with the same name but different values, the first one to be created is used throughout the installation. It can cause unexpected results. Employ a pattern for naming the properties to ensure that property names across components do not clash. For example, a property name might have the name of the component to which it relates, prefixed to the name to ensure uniqueness.

Database properties for the Solution Installer

Some Portal Application Archive (PAA) files require access to an external database. The database properties are stored in either the `assemblyName.properties` file for the assembly or in the `componentName.properties` file of the component requiring database support.

The Solution Installer uses the following database properties:

dbName
The name of the database created to store tables required by the PAA file.

Ideally, this should be specified in the `default.properties` file, especially if the name of the database is explicitly included in the table population scripts.

DbPort
Specify the database port number. The following examples are default port numbers for each database, but your server port number might be different if you changed the port values:

- Derby: `1527`
- DB2: `50000`
- Oracle: `1521`
- SQL Server : `1433`

DbHostname
The host name or IP address of the server hosting the database.

dbType
The type of database. Use the following valid database types:

- Derby: `derby`
- DB2: `db2`
- Oracle: `oracle`
- SQL Server : `SQL Server`

dbProviderName
The name of jdbc provider to be used. Use the following example jdbc provider names:

- Derby: `wpdbJDBC_derby`
- DB2: `wpdbJDBC_db2`
- Oracle: `wpdbJDBC_oracle`
- SQL Server : `wpdbJDBC_sqlserver`

dbDriverType
Connection pool data source

dbUsername
The username for connecting to the database.

dbPassword
The password for connecting to the database.

dbDriverName
The name of database driver. Use the following valid driver names:

- Derby: `org.apache.derby.jdbc.EmbeddedDriver`
- DB2: `com.ibm.db2.cc.DB2Driver`
- Oracle: `oracle.jdbc.driver.OracleDriver`
- SQL Server : `com.microsoft.sqlserver.jdbc.SQLServerDriver`

dbDriverPath
The path to the database driver. Use one of the following examples with values specific to your database:

- Derby: `${WasHome}/derby/lib`
- DB2: `${WasHome}/deploytool/itp/plugins/${dbPlugin}/driver`
- Oracle: `{$ORACLE_HOME}/jdbc/lib/`

  Note: ORACLE_HOME is the environment variable specified during the installation of the Oracle database.

- SQL Server : *installation_directory*/sqljdbc_2.0/enu

dbClasspath
The database class path value. Use one of the following examples with values specific to your database:

- Derby: ${dbDriverPath}/derby.jar:${dbDriverPath}/derbyclient.jar:${dbDriverPath}/derbytools.jar:${dbDriverPath}/derbynet.jar
- DB2: ${dbDriverPath}/db2jcc4.jar:${dbDriverPath}/db2jcc_license_cisuz.jar:${dbDriverPath}/db2jcc_license_cu.jar
- Oracle: ${dbDriverPath}/ojdbc6.jar
- SQL Server : *installation_directory*/sqljdbc_2.0/enu/sqljdbc4.jar

dbUrl The database URL value. Use one of the following examples with values specific to your database:

- Derby: jdbc:${dbType}:${dbName}
- DB2: jdbc:${dbType}://${dbHostname}:${dbPort}/${dbName}
- Oracle: jdbc:${dbType}:thin:@${dbHostname}:${dbPort}:${dbName}
- SQL Server : jdbc:sqlserver://*hostname*:${DbPort};SelectMethod=cursor;DatabaseName=tbmesg

dbJndiName
Specify the JNDI name that will be used for a component.

dsTemplateName
The data source template name. Use one of the following examples with values specific to your database:

- Derby: Derby JDBC Driver DataSource
- DB2: DB2 Universal JDBC Driver DataSource
- Oracle: Oracle JDBC Driver DataSource
- SQL Server : Microsoft SQL Server JDBC Driver - XA DataSource

jpTemplateName
Use one of the following examples with values specific to your database:

- Derby: Derby JDBC Provider
- DB2: DB2 Universal JDBC Driver Provider
- Oracle: Oracle JDBC Driver Provider
- SQL Server : Microsoft SQL Server JDBC Driver

dsDbDriverType
The type of the database driver that the data source connects to. Valid values are 2 and 4.

dataSourceName
Specifies the Data Source name will be used for the component.

dbAuthDataAlias
Specifies the Authentication Alias.

db.connectionTimeout
The interval, in seconds, after which a connection request times out and a ConnectionWaitTimeoutException is thrown. Default value is 180.

db.maxConnections
The maximum number of physical connections that you can create in this pool. Default value is 30.

db.reapTime
The interval, in seconds, between runs of the pool maintenance thread. Default value is 120.

db.agedTimeout
The interval in seconds before a physical connection is discarded. Default value is 1800.

Virtual portals in the PAA file

There are many situations where a user might want to install the applications that are contained in a Portal Application Archive (PAA) file directly to a virtual portal.

No additional configuration steps are needed when you create the PAA file. The Solution Installer takes advantage of the virtual portal-related properties that are available in the `wkplc.properties` file when it connects to the portal server. The Solution Installer checks the value of the **`VirtualPortalHostName`** and **`VirtualPortalContext`** properties. If one or more of these properties are set, then their values are used when it creates a connection. However, if neither of these properties are set, it is assumed that the PAA file content is installed to the base WebSphere Portal Express. These properties can be set in the `wkplc.properties` file or included on the command line with a prefix of **`-D`** when you run the deployment. For example, `-D`**`VirtualPortalContext`**`=testVP`. In addition, when you remove resources from a virtual portal, these properties must also be set.

Note: The properties and parameters must be set before you run the installer to deploy the PAA application. There is no facility to install individual components of the PAA file to different virtual portals during a single run of the **`deploy-paa`** task. Therefore, run the deployment for the specific components that are required for each virtual portal separately. All components that are listed for deployment are installed to the same location.

If you deploy a PAA file to more than one Virtual Portal, some components are not deployed on the other Virtual Portals. Therefore, items such as Libraries are not created on the second Virtual Portal. If you downloaded the PAA file from the IBM WebSphere Portal Business Solutions Catalog, refer to the associated documentation for instructions on how to deploy to Virtual Portals. If the PAA file is not from the IBM WebSphere Portal Business Solutions Catalog, choose one of the following options:

- Open the *wp_profile_root*`/paa/`*paaName*`/components.properties` file. Set the components that you want to deploy to true. Then, run the **`deploy-paa`** task.
- Run the **`deploy-paa`** task with the `-DforceDeploy=true` option. This parameter tells the Solution Installer to ignore the `components.properties` file.

Developing advanced PAA file applications

Developers can create their own advanced Portal Application Archive (PAA) file. The advanced PAA file contains custom content. The developers can then use the configuration wizard to add on their applications to their IBM WebSphere Portal Express environment.

About this task

Review the following requirements before you create your own advanced Portal Application Archive (PAA) file:

"The component level `sdd.xml` file" on page 3432
When you run the **`install-paa`** task, the Solution Installer examines each

component to verify whether an `sdd.xml` file is included for that component. If an `sdd.xml` file is not found, one is generated with the information gathered from the directory parsing step.

"Add custom code to a Portal Application Archive (PAA) file" on page 3437
Solution developers can create applications that use resource types that the Solution Installer does not automatically generate. The Solution Installer handles many resource types. However, there are some resource types that have no mechanism for accurate installation procedures. Your application might require additional configuration settings for installation. Those applications require custom code for the installation.

"Updating a Portal Application Archive (PAA) file" on page 3438
Before Version 8.5, updating the PAA file content in the portal server was not a simple task. It usually involved removing and uninstalling the previous version, then installing or deploying an updated version of the PAA file. This work can lead to a number of problems because it might break existing links between pages or portlets that were created outside of the PAA file deployment. Since Version 8.5, the Solution Installer can handle this type of function. Only some of the PAA file resources might change. Therefore, it does not make sense to overwrite the full set of resources on the system, but only those resources that changed since the new PAA file was released.

"ConfigEngine extension points for the Solution Installer" on page 3440
Some ConfigEngine extension points are required when you install an application from a Portal Application Archive (PAA) file.

"Tasks and extension points for custom code" on page 3442
You can use the following tasks and extension points when you create custom code to work with the ConfigEngine framework.

Related concepts:

"Install and uninstall add-ons using the Configuration Wizard" on page 212
You can install add-on functionality to your WebSphere Portal Express with the solution installer through the Configuration Wizard. The add-ons that are accepted by the configuration options are `.paa` files. For more information, see the solution installer documentation.

Related tasks:

"Developing basic PAA file applications" on page 3402
Solution developers can create their own Portal Application Archive (PAA) files. The developers can then use the configuration wizard to add on their applications to their IBM WebSphere Portal Express environment.

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

The component level `sdd.xml` file

When you run the **`install-paa`** task, the Solution Installer examines each component to verify whether an `sdd.xml` file is included for that component. If an `sdd.xml` file is not found, one is generated with the information gathered from the directory parsing step.

The type of data that is gathered from the components directory structure includes information on the different resource types. For example, the list of resources that are included in the individual sub directories and any custom extension point implementation tasks. It also uses the name of the directory when it generates the

component name. After it gathers the information, it can generate the sdd.xml file. The file includes the information on the extension points for registration with the Solution Installer or ConfigEngine. The Solution Installer can add any information about extension points whether auto-generated or custom implementations to the sdd.xml file. There is no longer a need to manually register any custom tasks with <SCU> elements.

The ability to automatically generate dependencies between components was added. Include an order.properties file in the /components directory of the Portal Application Archive (PAA) file. Read *Order of installation of scripts and artifacts* for more information.

For most situations, the SDD automation is enough to allow the PAA files to be deployed successfully. However, there are still some situations where the developer needs to overwrite the function of the sdd.xml file. One example might be in the ordering of components where the order.properties file does not meet the requirements for the PAA file. For example, a component contains extension points that have dependencies on 2 or more different components.

This information outlines the different pieces of the component level sdd.xml file in terms of their use. In addition, steps on how to create your own custom sdd.xml file and where to include relevant information on the component for successful deployment are also covered.

Note: If you are providing a component level sdd.xml file, you do not need to provide information for all the resources included in the PAA file. Provide information only where you are overwriting the function of the Solution Installer.

The following is an example component level sdd.xml file:

```
<?xml version="1.0" encoding="UTF-8"?>

<iudd:iudd
 xmlns:iudd="http://www.ibm.com/xmlns/prod/autonomic/solutioninstall/IUDD"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:OSRT="http://www.ibm.com/xmlns/prod/autonomic/resourcemodel/OS/resourcetypes"
 xmlns:OSAT="http://www.ibm.com/xmlns/prod/autonomic/resourcemodel/OS/artifacttypes"
 xmlns:J2EERT="http://www.ibm.com/xmlns/prod/autonomic/resourcemodel/J2EE/resourcetypes"
 xsi:schemaLocation="http://www.ibm.com/xmlns/prod/autonomic/solutioninstall/IUDD ../iudd/iudd.xsd
 schemaVersion="2.0.0" buildID="112220" buildDate="2006-01-19T12:00:00">

 <packageIdentity contentType="Component">
  <name>components/componentN</name>
  <version>8.0.0.0</version>
  <displayName key="d0001" default="components/componentN" />
  <manufacturer>
   <displayName key="AC_01" default="IBM" />
  </manufacturer>
 </packageIdentity>

 <topology>
  <resource type="OSRT:OperatingSystem" id="OS" />
 </topology>

 <content xsi:type="iudd:RootIUContent">
  <rootIU id="components/componentN">
   <variables>
    <parameters>
     <parameter name="installLocation" defaultValue="/usr/dummy.offr.1" />
    </parameters>
   </variables>
```

```
   <SCU id="deploy-portlets-applySIFeaturePack" targetRef="OS">
    <identity>
     <name>Do Configuration Task</name>
     <version>8.0.0.0</version>
     <displayName key="keyInBundle"
      default="Executes Configuration for this component" />
     <description key="keyInBundle"
      default="This section runs configuration for this component" />
    </identity>
    <unit>ts-applySIFeaturePack" />
      </parameters>
     </configArtifact>
    </unit>
   </SCU>
   <SCU id="remove-portlets-applySIFeaturePack" targetRef="OS">
    <identity>
     <name>Do Configuration Task</name>
     <version>1.0.0.0</version>
     <displayName key="keyInBundle"
      default="Executes Configuration for this component" />
     <description key="keyInBundle"
      default="This section runs configuration for this component" />
    </identity>
    <unit>
     <configArtifact type="ConfigEngine">
      <parameters>
       <parameter name="targetName" value="remove-portlets-applySIFeaturePack" />
      </parameters>
     </configArtifact>
    </unit>
   </SCU>
  </rootIU>
 </content>

</iudd:iudd>
```

Inside the root <iudd:iudd> element the following important subelements are required for installation by the Solution Installer:

The <packageIdentity> element

The first element is the <packageIdentity> element; for example:

```
<packageIdentity contentType="Component">
  <name>components/componentN</name>
  <version>8.5.0.0</version>
  <displayName key="d0001" default="components/componentN" />
  <manufacturer>
   <displayName key="AC_01" default="IBM" />
  </manufacturer>
 </packageIdentity>
```

The <packageIdentity> element informs Solution Installer and the ConfigEngine of the type of application to be installed. In this case, the **contentType** attribute equals "component". A component is the lowest level of granularity that is used by the ConfigEngine and thus Solution Installer for grouping application resources. One or more components are grouped in an assembly. This grouping is the next level of granularity for grouping resources that are recognized by the ConfigEngine or the Solution Installer. Each PAA file is considered by Solution Installer as an assembly, even though it might contain only one component. With the PAA file, a component must have a higher-level assembly that is associated with it. Read *The assembly level sdd.xml file* for information.

If you look at the <packageIdentity> element, a number of the subelements provide information on the content that is being installed. The

<name> and <version> element need to be altered to suit the application that is being installed. The <displayName> element can also be edited to include the component name but is not required.

The <name> element must equal the path of the component relative to the assembly level sdd.xml file. For example, as the PAA file puts all components in the components directory, a component name needs to also include the components/. For example, a component that is called 'componentN' would have a component name of 'components/componentN'.

The <topology> element

The next sub element is the <topology> element. This element does not require any alterations. For example:

```
<topology>
  <resource type="OSRT:OperatingSystem" id="OS" />
 </topology>
```

The <content> element

The final element block is the <content> element. This element provides the Solution Installer and the ConfigEngine with the information on what is being installed and on which servers the package can be installed on. The <rootIU> element is where the actual information is included. Set the **id** attribute of the <rootIU> element to the name of the component as used in the <name> element of the <packageIdentity> element. For example:

```
<content xsi:type="iudd:RootIUContent">
  <rootIU id="components/componentN">
   <variables>
    <parameters>
     <parameter name="installLocation" defaultValue="/usr/dummy.offr.1" />
    </parameters>
   </variables>

   <SCU id="deploy-portlets-applySIFeaturePack" targetRef="OS">
    <identity>
     <name>Do Configuration Task</name>
     <version>1.0.0.0</version>
     <displayName key="keyInBundle"
      default="Executes Configuration for this component" />
     <description key="keyInBundle"
      default="This section runs configuration for this component" />
    </identity>
    <unit>
     <configArtifact type="ConfigEngine">
      <parameters>
       <parameter name="targetName" value="deploy-portlets-applySIFeaturePack" />
      </parameters>
     </configArtifact>
    </unit>
   </SCU>
  </rootIU>
 </content>
```

The ConfigEngine requires the <variables> element for processing the content. You do not need to edit this code section. Include it in the component level sdd.xml file per the previous example.

The <SCU> element is the element of the file that is of most relevant to adding custom code to your PAA file. Each <SCU> element represents a ConfigEngine extension point. These extension points govern how and when resources are installed or uninstalled on the server during the **deploy-paa** task. Each extension point has an equivalent implementation task that needs to be added to an XML file

in the components/componentN/config/includes directory. Read *ConfigEngine extension points for the Solution Installer* for a list of exported extensions and how they map to directories. For details on extension points and how they are used, read *Component level sdd.xml file overview*.

The Solution Installer automatically adds the correct <SCU> element to the sdd.xml file after it detects an extension point implementation task. The Solution Installer first examines the components/componentName/config/includes directory for xml files that contain any custom Ant tasks. Any tasks that meet the extension point implementation criteria are then registered in the sdd.xml file where an existing <SCU> element for the extension point does not exist. However, you can register the <SCU> elements manually, especially where you need to set dependencies between multiple components on different extension points.

In the previous example, there are two places to changes. The first place is the **id** attribute of the <SCU> element. Set the ID to the name of the required extension point with **applySIFeaturePack** appended to the end. For example: deploy-apps-applySIFeaturePack. The second to change is the nested <parameter> element. Set the value attribute of this element to the extension point string. For example, set it to deploy-apps-applySIFeaturePack.

For each extension point, a separate <SCU> element is required by the Solution Installer or ConfigEngine. Therefore, an extra <SCU> element is required to provide an equivalent remove-apps-removeSIFeaturePack extension.

A second and rather important role is assigned to the <SCU> element. These elements also influence the order in which components are installed. You might have a PAA file with two components, where the second component is installed before the first component. A mechanism is necessary to ensure that the correct order is imposed on the deployment. This action can be done by adding a <requirements> element to the <SCU> element as shown in the following example:

```
<SCU id="deploy-portlets-applySIFeaturePack" targetRef="OS">
  <identity>
    <name>Do Configuration Task</name>
    <version>1.0.0.0</version>
    <displayName key="keyInBundle" default="Executes Configuration for this component" />
    <description key="keyInBundle" default="This section runs configuration for this component" />
  </identity>
  <unit>
    <configArtifact type="ConfigEngine">
      <parameters>
        <parameter name="targetName" value="deploy-portlets-applySIFeaturePack" />
      </parameters>
    </configArtifact>
  </unit>
<!—register a dependency on a previous component-->
<requirements>
<requirement name=" deploy-portlets-applySIFeaturePack">
<alternative name="dependentComponent"/>
</requirement>
</requirements>
</SCU>
```

The <requirements> element can contain one or more <requirement> elements, each one representing a component on which the current component depends on being previously installed. The name attribute of the <requirement> element can be set to whatever value the developer chooses. The name attribute of the <alternative> element must equal the full name of the component on which the current extension point depends. The dependency link is between extension points.

If you add the <requirements> element to the <SCU> element for the deploy-apps-applySIFeaturePack, register an equivalent deploy-apps-applySIFeaturePack extension point for the dependent component.

Not all extension points allow for a component dependency to be set. The extension points that are called as part of the deploy-apps-applySIFeaturePack, such as **install-content-xmlaccess**, apply the <requirements> element to the deploy-apps-applySIFeaturePack extension point. Read *ConfigEngine extension points for the Solution Installer* for the full list of extension points that are supported by the Solution Installer and whether they can handle component dependencies.

Add custom code to a Portal Application Archive (PAA) file

Solution developers can create applications that use resource types that the Solution Installer does not automatically generate. The Solution Installer handles many resource types. However, there are some resource types that have no mechanism for accurate installation procedures. Your application might require additional configuration settings for installation. Those applications require custom code for the installation.

Where to place custom code in the Portal Application Archive (PAA) file

An example of a PAA file is located in the *PortalServer_root*/doc/paa-samples/sample1.paa directory. This example contains a sample component file: sample1/components/sample1. The sample component file contains the config/include and config/templates directories. Custom code and Solution Installer generated code are stored in these two directories. The config/templates directory should contain any additional scripts that are required for installation. An example of an additional script is one that configures a .war file. The config/include directory should contain any custom ANT task. Any scripts in the config/include directory are picked up at run time.

How to name custom code files

There are no restrictions on how to name files containing custom code. The Solution Installer adds automatically generated code to the file. It uses the portion of the component name following the / combined with the string _cfg.xml. Using the sample_paa/components/componentN component example, the component name is components/componentN. Therefore, the file is called componentN_cfg.xml.

For WAR file installations from the installableApps/portlets directory, install and uninstall XML access scripts are generated with file names using the following strings:

- -portlets-install.xml
- -portlets-uninstall.xml
- -portletDataGen.xml

For example, if you have a WAR file called MyWar.war, the following files are created:

- MyWar.war-portlets-install.xml
- MyWar.war-portlets-uninstall.xml
- MyWar.war-portletDataGen.xml

You should name your files that contain custom ANT tasks so you avoid situations where the Solution Installer overwrites it.

Component level sdd.xml file

The component level sdd.xml file is located in the component directory. Using the sample_paa/components/componentN example, the sdd.xml file is located in the componentN directory. The sdd.xml file is only required if you have custom code; otherwise, the Solution Installer generates it.

Add an <scu> element to your component level sdd.xml file to register your code with the ConfigEngine.

Sample custom code

The following sample code uses the <scu> element for the extension point **create-ear**. The code tells the Solution Installer to overwrite the **create-ear** functionality. The full name of the extension is **create-ear-applySIFeaturePack**. The **create-ear** part tells the ConfigEngine what type of installation functionality is used to group the tasks to run. The **-applySIFeaturePack** part allows the Solution Installer to invoke functionality of the ConfigEngine for components that conform to this naming scheme. Hence it reduces the likelihood that other components, for example core portal components, from being removed during uninstallation.

```
<SCU id="create-ear-applySIFeaturePack" targetRef="OS">
 <identity>
  <name>create-ear-applySIFeaturePack</name>
  <version>1.0.0</version>
  <displayName key="deploy-apps-applySIFeaturePack" default="Executes
   Configuration for this component" />
  <description key="deploy-apps-applySIFeaturePack" default="Executes
   Configuration for this component" />
 </identity>
 <unit>
  <configArtifact type="ConfigEngine">
  <parameters>
  <parameter name="targetName" value="create-ear-applySIFeaturePack" />
  </parameters>
  </configArtifact>
 </unit>
</SCU>
```

After registering the extension point with the sdd.xml file, an implementation task must be created to overwrite the default functionality. The task should reside in the config/includes directory of the component. There is a specific naming convention required for the task so that it is directly associated with the <scu> element in the sdd.xml file. Use the following naming convention:

```
'action;' + extension point name = '-' = componentname
```

For example: action-create-ear-applySIFeaturePack-components/componentN.

Tip: Use the full path of the component as the component name.

Updating a Portal Application Archive (PAA) file

Before Version 8.5, updating the PAA file content in the portal server was not a simple task. It usually involved removing and uninstalling the previous version, then installing or deploying an updated version of the PAA file. This work can lead to a number of problems because it might break existing links between pages

or portlets that were created outside of the PAA file deployment. Since Version 8.5, the Solution Installer can handle this type of function. Only some of the PAA file resources might change. Therefore, it does not make sense to overwrite the full set of resources on the system, but only those resources that changed since the new PAA file was released.

In addition, some resource types are difficult to update without removing the existing version and redeploying the new resource. For example, IBM Web Content Manager libraries. The developer must decide whether the library needs to be replaced or whether the existing version is good. A number of new Solution Installer tasks handle the update of a PAA file. Read *Managing your existing Portal Application Archive (PAA) file* for information on the tasks.

The **install-paa-update** task verifies that an existing PAA file is installed to the ConfigEngine and creates a backup of the current set of files that are placed in the `profile_dir/paa/backup` directory. The existing PAA file is removed from the ConfigEngine registry and most of the files from the expanded PAA file directory are deleted. The auto-generated code files are not replaced during this step because the PAA files might be different. Therefore, the removal tasks are still available to the Solution Installer and called by the custom code.

After this step completes, the Solution Installer continues with the regular **install-paa** task. However, it registers that the new PAA file is an update and records the location of the previous backed up version. It is this information that allows the **deploy-paa** task to run only the update-related tasks and not all the Ant tasks that are registered for the PAA file.

It would be difficult for the Solution Installer to establish what pieces of the PAA file are deployed as part of the update. Therefore, as the developer is in the best position to provide this input, the Solution Installer does not attempt to generate any default installation code for the update tasks. Instead, when the **deploy-paa** task detects that a PAA file is an update, it runs only extension point implementation tasks that meet certain naming criteria. For deploying an update, the extension point uses the `updateSIFeaturePack` suffix instead of the `applySIFeaturePack` suffix. For example, the **deploy-apps-applySIFeaturePack** in terms of an update would be **deploy-apps-updateSIFeaturePack**.

The content of these tasks is up to the developer of the PAA file. However, the existing Solution Installer generated tasks are present. The developer can start this code as part of any tasks they add. The Solution Installer does not overwrite any extension point implementation tasks during the **install-paa-update** task. If more resources are added, for example another ear file is created in the `components/componentName/installableApps/ear` directory, it must be deployed with custom code.

Although these update tasks are not automatically generated, the Solution Installer can automatically register them with the `sdd.xml` file for the component. Read *The component level `sdd.xml` file* for information on how to register the extension points. After an extension point implementation task is provided, read *Add custom code to a Portal Application Archive (PAA) file*. The Solution Installer automatically registers the task. The following is an example task for the 'componentN' component and the **deploy-apps-updateSIFeaturePack** task:

```
<target name="action-deploy-apps-updateSIFeaturePack-components/componentN">
</target>
```

In a similar fashion, the rollback of the update with the **remove-paa-update** task runs the extension points with the `rollbackSIFeaturePack` suffix. This task removes all the update PAA resources and restore the artifacts from the backup. When complete, the Solution Installer runs any extension point with the `rollbackSIFeaturePack` suffix. It is the developers responsibility to generate the tasks that can remove the resources and then deploy the previous versions. The original Ant tasks both custom and auto-generated are available and can be called from the custom Ant tasks. The following is an example of a rollback task:

```
<target name="action-remove-apps-rollbackSIFeaturePack-components/componentN">
</target>
```

Tasks with the `updateSiFeaturePack` and `rollbackSIFeaturePack` suffix are run only when the PAA file is registered as an update during the **install-paa-update** task. If the installation is on a fresh system, only the `applySIFeaturePack` and `removeSIFeaturePack` extension points are run.

ConfigEngine extension points for the Solution Installer

Some ConfigEngine extension points are required when you install an application from a Portal Application Archive (PAA) file.

Table 497. ConfigEngine extension points

PAA directory	Extension point name	Description
Config		No extension points required
Content		
Database	**create-jdbc-provider-applySIFeaturePack**	Create the JDBC provider for Solution Installer
Database	**remove-jdbc-provider-removeSIFeaturePack**	Remove the JDBC provider for Solution Installer
Database	**create-j2c-auth-applySIFeaturePack**	Create J2C authentication alias
Database	**remove-j2c-auth-removeSIFeaturePack**	Remove J2C authentication alias
Database	**create-dataSource-applySIFeaturePack**	Create the datasource for Solution Installer
Database	**remove-dataSource-removeSIFeaturePack**	Remove the datasource for Solution Installer
Database	**create-database-applySIFeaturePack**	Runs script to create the database structures, e.g. db schema, tables buffers etc.
Database	**setup-database-applySIFeaturePack**	Runs script to populate the database with sample data
Database	**remove-database-removeSIFeaturePack**	Runs scripts to remove the database structures from the db, e.g. drops all tables.
Database	**create-cmp-connection-factory-applySIFeaturePack**	Creates cmp connection factory for container management. This extension point does not exist so it needs to be created.

Table 497. ConfigEngine extension points (continued)

PAA directory	Extension point name	Description
Database	**remove-cmp-connection-factory-removeSIFeaturePack**	Removes cmp connection factory for container management. This extension point does not exist so it needs to be created.
JCR		No extension points available because this is custom code
JSP		No extension points available because this is custom code
Personalization	**create-personalisation-rules-applySIFeaturePack**	Creates the personalization rules for the library
Personalization	**remove-personalisation-rules-removeSIFeaturePack**	Currently not available.
WCM	**import-wcm-applySIFeaturePack**	Imports WCM Library
XMLAccess	**install-content-xmlaccess-applySIFeaturePack**	Runs XML install scripts in install/configure application
XMLAccess	**remove-content-xmlaccess-removeSIFeaturePack**	Run XML scripts to remove application
webdav	**Import-webdav-applySIFeaturePack**	Uploads WebDav artefacts to the IBM WebSphere Portal Express WebDav file system
InstallableApps		
EAR	**create-ear-applySIFeaturePack**	Installs EAR file
EAR	**remove-ear-removeSIFeaturePack**	Removes EAR file
Portlets	**deploy-portlets-applySIFeaturePack**	Deploy portlets from WAR file
Portlets	**remove-portlets-removeSIFeaturePack**	Removes portlets from portal
WAR	**Copy-war-files-applySIFeaturePack**	Copy war files to the Profile dir directory
WAR	**Delete-war-files-applySIFeaturePack**	Delete the war files copied across by the copy-warfiles extension point implementation
ZIP	No extension points	
Shared	No extension points	
App	No extension points	
Common	**create-library-applySIFeaturePack**	Creates shared library
Common	**remove-library-removeSIFeaturePack**	Removes shared library
Common	**create-application-library-references-applySIFeaturePack**	Creates shared library references

Table 497. ConfigEngine extension points (continued)

PAA directory	Extension point name	Description
Common	**remove-application-library-references-applySIFeaturePack**	Removes shared library references
Common	**create-app-server-library-references-applySIFeaturePack**	Adds shared library to server classpath
Common	**remove-app-server-library-references-applySIFeaturePack**	Removes shared library to server classpath
Ext	No extension points	
Templates	**deploy-pages-applySIFeaturePack**	Deploys pages in this template
Templates	**remove-pages-applySIFeaturePack**	Removes pages in this template
Templates	**add-templates-applySIFeaturePack**	Installs template
Templates	**delete-templates-applySIFeaturePack**	Removes templates

Tasks and extension points for custom code

You can use the following tasks and extension points when you create custom code to work with the ConfigEngine framework.

Common Properties

These attributes are a reflection of similar attributes present on the **WsAdmin** task. Not all attributes on the **WsAdmin** task are used and some have a restricted range of values. None of these values are written into the dynamic profile.

Table 498. Common WPLC properties. This table displays a list of the common properties that you can use with your extension points.

Attribute	Description	Required	Scope
wasuser	Contains the user ID to authenticate with the server.	Yes	Global
waspassword	Contains the password that is associated with the **wasuser** user ID to authenticate with the server.	Yes	Global
conntype	Specifies the type of server connection. The following values are allowed: • SOAP: Establishes a SOAP connection. This value is the default. • NONE: No server connection is used. Instead, a direct connection to the local WebSphere Application Server configuration repository is used.	No	Global

Table 498. Common WPLC properties (continued). This table displays a list of the common properties that you can use with your extension points.

Attribute	Description	Required	Scope
properties	Java properties file containing attributes to set in the JVM System properties. The default is `${EngineInstallLocation}/config/work/was/jacl.properties`.	No	Global
script	Provides the location of the deployment script file. This file contains a set of commands to be passed to the **WsAdmin** script processor. Set the **lang** attribute to determine the script processor. If not specified, each deployment task assigns a default script location. It is built from the home directory of the configuration engine and a name specific to the task. For example, `wplc_deployEar`. Refer to the individual task for this name. Set the **lang** attribute to determine the file extension.	No	Global
lang	Contains the language to be used to interpret scripts. The supported values are as follows: • `jacl`: Use the Jacl interpreter. This value is the default. • `jpython`: Use the Java Python interpreter.	No	Global

Framework-specific Attributes

These attributes provide information about the runtime environment that is associated with the configuration engine.

Table 499. Framework-specific attributes. Runtime environment attributes.

Attribute	Description	Required	Scope
engineinstalllocation	The location of the configuration engine home directory. This location is used as the root when you construct many of the default locations. It defaults to the setting of the Ant property: **engineInstallLocation**.	No	Global
engineinstalllocation	The location of a deployment file that contains some or all of the deployment attributes of the targeted resource. This file is an XML file whose syntax mirrors that of the Ant task that references the file.	No	not applicable

Embedded Markup Tag

The following markup tags are supported.

Comment markup tag

Use the comment markup tag to embed a comment into a task value. The comment text is displayed during the task output trace, which is enclosed by parentheses. It is not included in the value that is passed to the task execution. Use <c> or <comment> to begin a comment. This tag has no attributes.

Default markup tag

Use the default markup tag to force a default value for a task value. The default is chosen if the target value resolves to an empty string. Otherwise, it is ignored. Use <d> or <default> to delimit text that is defaulted. This tag has the following single attribute:

`value`: The default value. This attribute is required.

<lower> markup tag

Use the lower markup tag to force text to lowercase. Use `<l>` or `<lower>` to delimit lowercase text. This tag has no attributes.

Password markup tag

Use the password markup tag to indicate a portion of text that is to remain hidden (or obscured) whenever the text value is logged. Use `<p>`, `<pw>`, or `<password>` to indicate the beginning of password text. This tag has no attributes. The default is to substitute the string, PASSWORD_REMOVED, for the text value during logging. You can override this value with the log attribute.

`log`: The value that is substituted for the real password during logging of the task. Defaults to PASSWORD_REMOVED. This attribute is not required.

Upper markup tag

Use the upper markup tag to force text to uppercase. Use `<u>` or `<upper>` to delimit uppercase text. This tag has no attributes.

Examples

The following example illustrates the use for each of the supported tags. The last instance embeds an undefined tag. This tag causes the parsing to stop and results in a value that is unchanged.

```
<wplc-create-variable server="server"
      node="node"
      wasuser="${WasUserid}"
      waspassword="${WasPassword}"
      testbeanclass="com.ibm.wplc.deploy.tasks.impl.TestAdminBeanImpl">
      <resource name="r1" value="&lt;cValue for r1&lt;/>v1"/>
    <resource name="r2" value="&lt;dvalue=&quot;xxx&quot;> &lt;/>"/>
    <resource name="r3" value="<u>uPpEr</u>"/>
    <resource name="r4" value="<l>LoWeR</l>"/>
    <resource name="r5" value="<p>password</p>"/>
    <resource name="r6" value="<p log="SECRET">password</p>"/>
    <resource name="r7" value="<undefined attr="Attribute">password</undefined>"/>
</wplc-create-variable>
```

Running the previous task results in the following trace. The second set of output for each instance is produced by the test bean class. This bean class is a dummy bean implementation that is used to echo the output that is passed in to a bean for each WPLC task.

```
    [wplc-create-variable] Task parameters:
    [wplc-create-variable]   Global attributes:
    [wplc-create-variable]     cell=""
    [wplc-create-variable]     engineinstalllocation="directory_path"
    [wplc-create-variable]     pathseparator=";"
    [wplc-create-variable]     osarch="x86"
    [wplc-create-variable]     node="node"
    [wplc-create-variable]     server="server"
```

```
[wplc-create-variable]   Instance attributes (Set 1 of 7):
[wplc-create-variable]     name="r1"
[wplc-create-variable]     description= *** NOT_SPECIFIED ***
[wplc-create-variable]     value="(Comment:  Value for r1)v1"
[wplc-create-variable] name="r1"
[wplc-create-variable] engineinstalllocation="directory_path"
[wplc-create-variable] node="node"
[wplc-create-variable] server="server"
[wplc-create-variable] pathseparator=";"
[wplc-create-variable] osarch="x86"
[wplc-create-variable] value="v1"
[wplc-create-variable] cell=""
[wplc-create-variable] description=""

[wplc-create-variable]   Instance attributes (Set 2 of 7):
[wplc-create-variable]     name="r2"
[wplc-create-variable]     description= *** NOT_SPECIFIED ***
[wplc-create-variable]     value="xxx"
[wplc-create-variable] name="r2"
[wplc-create-variable] engineinstalllocation="directory_path"
[wplc-create-variable] node="node"
[wplc-create-variable] server="server"
[wplc-create-variable] pathseparator=";"
[wplc-create-variable] osarch="x86"
[wplc-create-variable] value="xxx"
[wplc-create-variable] cell=""
[wplc-create-variable] description=""

[wplc-create-variable]   Instance attributes (Set 3 of 7):
[wplc-create-variable]     name="r3"
[wplc-create-variable]     description= *** NOT_SPECIFIED ***
[wplc-create-variable]     value="UPPER"
[wplc-create-variable] name="r3"
[wplc-create-variable] engineinstalllocation="directory_path"
[wplc-create-variable] node="node"
[wplc-create-variable] server="server"
[wplc-create-variable] pathseparator=";"
[wplc-create-variable] osarch="x86"
[wplc-create-variable] value="UPPER"
[wplc-create-variable] cell=""
[wplc-create-variable] description=""

[wplc-create-variable]   Instance attributes (Set 4 of 7):
[wplc-create-variable]     name="r4"
[wplc-create-variable]     description= *** NOT_SPECIFIED ***
[wplc-create-variable]     value="lower"
[wplc-create-variable] name="r4"
[wplc-create-variable] engineinstalllocation="directory_path"
[wplc-create-variable] node="node"
[wplc-create-variable] server="server"
[wplc-create-variable] pathseparator=";"
[wplc-create-variable] osarch="x86"
[wplc-create-variable] value="lower"
[wplc-create-variable] cell=""
[wplc-create-variable] description=""

[wplc-create-variable]   Instance attributes (Set 5 of 7):
[wplc-create-variable]     name="r5"
[wplc-create-variable]     description= *** NOT_SPECIFIED ***
[wplc-create-variable]     value="PASSWORD_REMOVED"
[wplc-create-variable] name="r5"
[wplc-create-variable] engineinstalllocation="directory_path"
[wplc-create-variable] node="node"
[wplc-create-variable] server="server"
[wplc-create-variable] pathseparator=";"
[wplc-create-variable] osarch="x86"
[wplc-create-variable] value="password"
[wplc-create-variable] cell=""
[wplc-create-variable] description=""

[wplc-create-variable]   Instance attributes (Set 6 of 7):
[wplc-create-variable]     name="r6"
[wplc-create-variable]     description= *** NOT_SPECIFIED ***
[wplc-create-variable]     value="SECRET"
[wplc-create-variable] name="r6"
[wplc-create-variable] engineinstalllocation="directory_path"
[wplc-create-variable] node="node"
[wplc-create-variable] server="server"
[wplc-create-variable] pathseparator=";"
[wplc-create-variable] osarch="x86"
[wplc-create-variable] value="password"
[wplc-create-variable] cell=""
[wplc-create-variable] description=""

[wplc-create-variable]   Instance attributes (Set 7 of 7):
[wplc-create-variable]     name="r7"
[wplc-create-variable]     description= *** NOT_SPECIFIED ***
[wplc-create-variable]     value="password"
[wplc-create-variable] name="r7"
[wplc-create-variable] engineinstalllocation="directory_path"
[wplc-create-variable] node="node"
[wplc-create-variable] server="server"
[wplc-create-variable] pathseparator=";"
[wplc-create-variable] osarch="x86"
[wplc-create-variable] value="password"
[wplc-create-variable] cell=""
[wplc-create-variable] description=""
```

wplc-create-application-library-ref

Definition and usage

Use this Ant task to create or modify the specified library reference.

Required parameters

The following parameters are required:

- **appName**: The name of the application with required library reference. The scope is instance.
- **libraryName**: The name of the library reference. The scope is instance.

Optional parameters

The following parameters are optional:

- **mode**: The mode of the class loader. The scope is instance.
- **sharedClassloader**: Boolean - specifies whether to use a shared class loader. The scope is instance.

Parameters that are specified as nested elements

attribute - a generic attribute for this resource. Users can use this attribute to specify any additional name-value pairs that are not already covered by the **wplc** task.

- **name**: The name of the attribute.
- **value**: The value of the attribute.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.ApplicationLibraryRefCreateImpl** class implements this task.

wplc-remove-application-library-ref

Definition and usage

Use this Ant task to remove the specified library reference.

Required parameters

The following parameters are required:

- **appName**: The name of the application with required library reference. The scope is instance.
- **libraryName**: The name of the library reference. The scope is instance.

Optional parameters

The following parameter is optional:

- **mode**: The mode of the class loader. The scope is instance.

Parameters that are specified as nested elements

attribute - a generic attribute for this resource. Users can use this attribute to specify any additional name-value pairs that are not already covered by the **wplc** task.

- **name**: The name of the attribute.
- **value**: The value of the attribute.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.ApplicationLibraryRefRemoveImpl** class implements this task.

wplc-create-app-server-classloader

Description and usage

Use this Ant task to create the specified application server class loader.

Required parameters

The following parameters are required:

- **mode**: The mode to operate the application server class loader in. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: Cell scope of resource action. The scope is global.
- **node**: The node scope of resource action. The scope is global.
- **server**: Server scope of resource action. The scope is global.

Parameters that are specified as nested Elements

A generic **attribute** parameter for this resource. Users can use this attribute to specify any additional name-value pairs that are not already covered by the **wplc** task.

- **name**: The name of the attribute.
- **value**: The value of the attribute.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.ApplicationServerClassloaderCreateImpl** class implements this task.

wplc-remove-app-server-classloader

Description and usage

Use this Ant task to create the specified application server class loader.

Required parameters

The following parameters are required:

- **mode**: The mode to operate the application server class loader in. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: Cell scope of resource action. The scope is global.
- **node**: The node scope of resource action. The scope is global.
- **server**: Server scope of resource action. The scope is global.

Parameters that are specified as nested Elements

A generic **attribute** parameter for this resource. Users can use this attribute to specify any additional name-value pairs that are not already covered by the **wplc** task.

- **name**: The name of the attribute.
- **value**: The value of the attribute.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.ApplicationServerClassloaderCreateImpl** class implements this task.

wplc-create-app-server-custom-property

Description and usage

Use this Ant task to create or modify a custom property.

Required parameters

The following parameters are required:

- **name**: The name of the property to create. The scope is instance.
- **value**: The value of the property to create. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: Cell scope of resource action. The scope is global.
- **description**: The description of the property you are creating. The scope is instance.
- **node**: Node scope of resource action. The scope is global.
- **required**: Boolean - specifies whether the property is required or not. The scope is instance.
- **server**: Server scope of resource action. The scope is global.

Parameters that are specified as nested Elements

attribute - a generic attribute for this resource. Users can use this attribute to specify any additional name-value pairs that are not already covered by the **wplc** task.

- **name**: The name of the attribute.
- **value**: The value of the attribute.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.ApplicationServerCustomPropertyCreateImpl** class implements this task.

wplc-remove-app-server-custom-property

Description and usage

Use this Ant task to remove a custom property.

Required parameters

The following parameter is required:

- **name**: The name of the property to remove. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: Cell scope of resource action. The scope is global.
- **node**: The node scope of resource action. The scope is global.
- **server**: Server scope of resource action. The scope is global.

Parameters that are specified as nested Elements

A generic **attribute** parameter for this resource. Users can use this attribute to specify any additional name-value pairs that are not already covered by the **wplc** task.

- **name**: The name of the attribute.
- **value**: The value of the attribute.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.ApplicationServerCustomPropertyRemoveImpl** class implements this task.

wplc-create-app-server-library-ref

Description and usage

Use this Ant task to create or modify the specified reference.

Required parameters

The following parameters are required:

- **libraryName**: Tells what library reference to add. The scope is instance.
- **mode**: The mode of the class loader. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: Cell scope of resource action. The scope is global.
- **node**: Node scope of resource action. The scope is global.
- **server**: Server scope of resource action. The scope is global.

Parameters that are specified as nested Elements

attribute - a generic attribute for this resource. Users can use this attribute to specify any additional name-value pairs that are not already covered by the **wplc** task.

- **name**: The name of the attribute.
- **value**: The value of the attribute.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.ApplicationServerLibraryRefCreateImpl** class implements this task.

wplc-remove-app-server-library-ref

Description and usage

Use this Ant task to remove the specified library reference.

Required parameters

The following parameters are required:

- **libraryName**: Tells what library reference to remove. The scope is instance.
- **mode**: The mode of the class loader. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: Cell scope of resource action. The scope is global.
- **node**: Node scope of resource action. The scope is global.
- **server**: Server scope of resource action. The scope is global.

Parameters that are specified as nested Elements

attribute - a generic attribute for this resource. Users can use this attribute to specify any additional name-value pairs that are not already covered by the **wplc** task.

- **name**: The name of the attribute.
- **value**: The value of the attribute.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.ApplicationServerLibraryRefRemoveImpl** class implements this task.

wplc-create-cmp-connection-factory

Description and usage

Use this Ant task to add a data source to the specified CMP J2CResourceAdaptor.

Required parameters

The following parameters are required:

- **authDataAlias**: The AuthAlias used this binding procedure. The scope is instance.
- **datasourcename**: Name of the data source. The scope is instance.
- **j2cresourceadaptorname**: The J2C Resource adapter name. The scope is instance.
- **jdbcprovidername**: Name of the JDBC provider that identifies this resource. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: Cell of the resource action. The scope is global.
- **containerauthalias**: Authorization alias for container. The scope is instance.
- **containermappingalias**: Mapping alias for container. The scope is instance.
- **description**: A description of the data source. The scope is instance.
- **forceScope**: Set to true if do not want to scope to the existing cluster. The scope is instance.
- **node**: Node of the resource action. The scope is global.
- **server**: Server of the resource action. The scope is global.
- **usecontainer**: Use the container. The value must be set to `true` or `false`. The scope is instance.

Parameters that are specified as nested Elements

attribute - a generic attribute for this resource. Users can use this attribute to specify any additional name-value pairs that are not already covered by the **wplc** task.

- **name**: The name of the attribute.
- **value**: The value of the attribute.

customproperty - used to create a custom property under this data source.

- **name**: The name of the custom property.
- **value**: The value of the custom property.

connectionpool - used to set the connection pool properties for this data source.

- **name**: The name of the connection pool.
- **value**: The value of the connection pool.

relationalresourceadapter - used to set the Relational Resource adapter properties for this data source.

- **name**: The name of the relational resource adapter.
- **value**: The value of the relational resource adapter.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.CMPConnectionFactoryCreateImpl** class implements this task.

wplc-remove-cmp-connection-factory

Description and usage

Use this Ant task to remove a data source to the specified CMP J2CResourceAdaptor.

Required parameters

The following parameters are required:

- **containermappingalias**: The mapping alias for the container.
- **datasourcename**: Name of the data source. The scope is instance.
- **j2cresourceadaptorname**: The J2C Resource adapter name. The scope is instance.
- **jdbcprovidername**: Name of the JDBC provider that identifies this resource. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: Cell of the resource action. The scope is global.
- **containerauthalias**: Authorization alias for container. The scope is instance.
- **containermappingalias**: Mapping alias for container. The scope is instance.
- **description**: A description of the data source. The scope is instance.
- **forceScope**: Set to true if do not want to scope to cluster, if exists. The scope is instance.
- **node**: Node of the resource action. The scope is global.
- **server**: Server of the resource action. The scope is global.
- **usecontainer**: Use the container. The value must be set to `true` or `false`. The scope is instance.

Parameters that are specified as nested Elements

attribute - a generic attribute for this resource. Users can use this attribute to specify any additional name-value pairs that are not already covered by the **wplc** task.

- **name**: The name of the attribute.
- **value**: The value of the attribute.

customproperty - used to create a custom property under this data source.

- **name**: The name of the custom property.
- **value**: The value of the custom property.

connectionpool - used to set the connection pool properties for this data source.

- **name**: The name of the connection pool.
- **value**: The value of the connection pool.

relationalresourceadapter - used to set the Relational Resource adapter properties for this data source.

- **name**: The name of the relational resource adapter.
- **value**: The value of the relational resource adapter.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.CMPConnectionFactoryRemoveImpl** class implements this task.

wplc-create-datasource

Description and usage

Use this Ant task to create a data source at the target scope.

Required parameters

The following parameters are required:

- **datasourcename**: Name of the data source. The scope is instance.
- **jdbcprovidername**: Name of the JDBC provider that identifies this resource. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: Cell scope of resource action. The scope is global.
- **containerauthalias**: Authorization alias for container. The scope is instance.
- **containermappingalias**: Mapping alias for container. The scope is instance.
- **DbDriverType**: Specifies the driver type for this data source. Can be the type 2 or type 4 driver for this database. The scope is instance.
- **DbType**: The database type for this data source. Use this parameter to specify the database type; for example: `db2`, `oracle`, `sqlserver`, `db2 for z0S`. The scope is instance.
- **DbUrl**: The database URL. Use the data source to connect to the database. The database name displays in this URL. The scope is instance.
- **description**: A description of the data source. The scope is instance.
- **forceScope**: Set to `true` if do not want to scope to existing cluster. The scope is instance.
- **isPortalDatasource**: Specifies whether this parameter is a Portal data source domain. Portal data source domains are governed by special rules for config-split and database domain function in Portal. If your data source is not governed by these rules, enter `false`. The default for this property is `true`. The scope is instance.
- **node**: Node of the resource action. The scope is global.
- **portalDomain**: Required only for WebSphere Portal domain-based data sources. This setting tells the name of the domain of the data source. The scope is instance.
- **server**: Server of the resource action. The scope is global.
- **templatename**: Name of the WebSphere template to follow when you create the data source. The scope is instance.
- **usecontainer**: Use the container. The value must be set to `true` or `false`. The scope is instance.
- **useXA**: Specify whether this data source uses XA connections or not. The value must be set to `true` or `false`. The scope is instance.

Parameters that are specified as nested Elements

attribute - a generic attribute for this resource. Users can use this attribute to specify any additional name-value pairs that are not already covered by the **wplc** task.

- **name**: The name of the attribute.
- **value**: The value of the attribute.

customproperty - used to create a custom property under this data source.

- **name**: The name of the custom property.
- **value**: The value of the custom property.

connectionpool - used to set the connection pool properties for this data source.

- **name**: The name of the connection pool.
- **value**: The value of the connection pool.

relationalresourceadapter - used to set the Relational Resource adapter properties for this data source.

- **name**: The name of the relational resource adapter.
- **value**: The value of the relational resource adapter.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.DataSourceCreateImpl** class implements this task.

wplc-modify-datasource

Description and usage

Use this Ant task to modify a data source at the target scope.

Required parameters

The following parameters are required:

- **datasourcename**: Name of the data source. The scope is instance.
- **jdbcprovidername**: Name of the JDBC provider that identifies this resource. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: Cell scope of resource action. The scope is global.
- **containerauthalias**: Authorization alias for container. The scope is instance.
- **containermappingalias**: Mapping alias for container. The scope is instance.
- **description**: A description of the data source. The scope is instance.
- **forceScope**: Set to `true` if do not want to scope to existing cluster. The scope is instance.
- **node**: Node of the resource action. The scope is global.
- **server**: Server of the resource action. The scope is global.
- **usecontainer**: Use the container. The value must be set to `true` or `false`. The scope is instance.

Parameters that are specified as nested Elements

attribute - a generic attribute for this resource. Users can use this attribute to specify any additional name-value pairs that are not already covered by the **wplc** task.

- **name**: The name of the attribute.
- **value**: The value of the attribute.

customproperty - used to create a custom property under this data source.

- **name**: The name of the custom property.
- **value**: The value of the custom property.

connectionpool - used to set the connection pool properties for this data source.

- **name**: The name of the connection pool.
- **value**: The value of the connection pool.

relationalresourceadapter - used to set the Relational Resource adapter properties for this data source.

- **name**: The name of the relational resource adapter.
- **value**: The value of the relational resource adapter.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.DataSourceModifyImpl** class implements this task.

wplc-remove-datasource

Description and usage

Use this Ant task to remove a data source at the target scope.

Required parameters

The following parameters are required:

- **datasourcename**: Name of the data source. The scope is instance.
- **jdbcprovidername**: Name of the JDBC provider that identifies this resource. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: Cell scope of resource action. The scope is global.
- **forceScope**: Set to `true` if do not want to scope to existing cluster. The scope is instance.
- **node**: Node of the resource action. The scope is global.
- **portalDomain**: Required only for WebSphere Portal domain-based data sources. This setting tells the name of the domain of the data source. The scope is instance.
- **server**: Server of the resource action. The scope is global.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.DataSourceRemoveImpl** class implements this task.

wplc-create-host-alias

Description and usage

Use this Ant task to create the specified Host Alias.

Required parameters

The following parameters are required:

- **host**: The host name for the host alias. The scope is instance.
- **port**: The port number for this host alias. The scope is instance.
- **virtualHostName**: The virtual host name to use for this host alias. The scope is instance.

Optional parameters

The following parameter is optional:

- **cell**: The cell scope of resource action. The scope is global.

Parameters that are specified as nested Elements

attribute - a generic attribute for this resource. Users can use this attribute to specify any additional name-value pairs that are not already covered by the **wplc** task.

- **name**: The name of the attribute.
- **value**: The value of the attribute.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.HostAliasCreateImpl** class implements this task.

wplc-remove-host-alias

Description and usage

Use this Ant task to remove the specified Host Alias.

Required parameters

The following parameters are required:

- **host**: The host name for the host alias. The scope is instance.
- **port**: The port number for this host alias. The scope is instance.
- **virtualHostName**: The virtual host name to use for this host alias. The scope is instance.

Optional parameters

The following parameter is optional:

- **cell**: The cell scope of resource action. The scope is global.

Parameters that are specified as nested Elements

attribute - a generic attribute for this resource. Users can use this attribute to specify any additional name-value pairs that are not already covered by the **wplc** task.

- **name**: The name of the attribute.
- **value**: The value of the attribute.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.HostAliasRemoveImpl** class implements this task.

wplc-create-j2c-auth

Description and usage

Use this Ant task to create an entry in the security table of the target cell. This entry can be used as a user-password pair for your configuration.

Required parameters

The following parameters are required:

- **alias**: The alias that is used to identify this authentication data. The scope is instance.
- **password**: The password that is associated with this user ID.
- **user**: The user name that is used to create a connection with this authentication data.

Optional parameters

The following parameters are optional:

- **cell**: The name of the cell where this J2C authentication data is created, removed, or modified. The scope is global.

- **description**: A description of what this authentication data is used for and what it connects to. The scope is instance.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.J2CAuthCreateImpl** class implements this task.

wplc-modify-j2c-auth

Description and usage

Use this Ant task to modify an entry in the security table of the target cell. This entry can be used as a user-password pair for your configuration.

Required parameters

The following parameters are required:

- **alias**: The alias that is used to identify this authentication data. The scope is instance.
- **password**: The password that is associated with this user ID.
- **user**: The user name that is used to create a connection with this authentication data.

Optional parameters

The following parameters are optional:

- **cell**: The name of the cell where this J2C authentication data is created, removed, or modified. The scope is global.
- **description**: A description of what this authentication data is used for and what it connects to. The scope is instance.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.J2CAuthModifyImpl** class implements this task.

wplc-remove-j2c-auth

Description and usage

Use this Ant task to remove an entry in the security table of the target cell.

Required parameters

The following parameter is required:

- **alias**: The alias that is used to identify this authentication data. The scope is instance.

Optional parameters

The following parameter is optional:

- **cell**: The name of the cell where this J2C authentication data is created, removed, or modified. The scope is global.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.J2CAuthRemoveImpl** class implements this task.

wplc-create-jvm-custom-property

Description and usage

Use this Ant task to create or update a custom JVM property in the WebSphere Application Server environment.

Required parameters

The following parameters are required:

- **name**: The name of the JVM custom property. The scope is instance.
- **value**: The value of the property to create.

Optional parameters

The following parameters are optional:

- **cell**: The cell scope of the resource action. The scope is global.
- **description**: A description of the JVM custom property. The scope is instance.
- **node**: The node scope of the resource action.
- **required**: The Boolean value that is used to tell WebSphere if this property is required or not.
- **server**: The server scope of the resource action.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.JVMCustomPropertyCreateImpl** class implements this task.

wplc-remove-jvm-custom-property

Description and usage

Use this Ant task to remove a custom JVM property in the WebSphere Application Server environment.

Required parameters

The following parameters are required:

- **name**: The name of the JVM custom property. The scope is instance.
- **value**: The value of the property to create.

Optional parameters

The following parameters are optional:

- **cell**: The cell scope of the resource action. The scope is global.
- **node**: The node scope of the resource action.
- **server**: The server scope of the resource action.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.JVMCustomPropertyRemoveImpl** class implements this task.

wplc-create-jdbc-provider

Description and usage

Use this Ant task to create a JDBC provider resource on the target scope.

Required parameters

The following parameter is required:

- **jdbcprovidername**: The name of the JDBC provider that identities this resource. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: Cell scope of resource action. The scope is global.
- **DbDriverType**: Specifies the driver type for this JDBC Provider. Can be the type 2 or type 4 driver for this database. The scope is instance.
- **DbType**: The database type for this JDBC Provider. Use this parameter to specify the database type; for example: `db2`, `oracle`, `sqlserver`, `db2 for z0S`. The scope is instance.

- **DbUrl**: The database URL. This JDBC Provider is used to connect to the database. The database name displays in this URL. The scope is instance.
- **forceScope**: Set to `true` if do not want to scope to an existing cluster. The scope is instance.
- **implementationClassName**: The class name that is used to create this JDBC Provider. The scope is instance.
- **node**: The node of the resource action. The scope is global.
- **server**: The server of the resource action. The scope is global.
- **templatename**: Name of the template to follow when you create the JDBC Provider. The scope is instance.
- **useXA**: Specify whether this JDBC Provider uses XA connections or not. Set to `true` or `false`. The scope is instance.

Parameters that are specified as nested Elements

attribute - a generic attribute for this resource. Users can use this attribute to specify any additional name-value pairs that are not already covered by the **wplc** task.

- **name**: The name of the attribute.
- **value**: The value of the attribute.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.JDBCProviderCreateImpl** class implements this task.

wplc-modify-jdbc-provider

Description and usage

Use this Ant task to create a JDBC provider resource on the target scope.

Required parameters

The following parameter is required:

- **jdbcprovidername**: The name of the JDBC provider that identities this resource. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: Cell scope of resource action. The scope is global.
- **forceScope**: Set to `true` if do not want to scope to an existing cluster. The scope is instance.
- **node**: The node of the resource action. The scope is global.
- **server**: The server of the resource action. The scope is global.

Parameters that are specified as nested Elements

attribute - a generic attribute for this resource. Users can use this attribute to specify any additional name-value pairs that are not already covered by the **wplc** task.

- **name**: The name of the attribute.
- **value**: The value of the attribute.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.JDBCProviderModifyImpl** class implements this task.

wplc-remove-jdbc-provider

Description and usage

Use this Ant task to remove a JDBC provider resource on the target scope.

Required parameters

The following parameter is required:

- **jdbcprovidername**: The name of the JDBC provider that identities this resource. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: Cell scope of resource action. The scope is global.
- **forceScope**: Set to `true` if do not want to scope to an existing cluster. The scope is instance.
- **node**: The node of the resource action. The scope is global.
- **server**: The server of the resource action. The scope is global.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.JDBCProviderRemoveImpl** class implements this task.

wplc-create-library

Description and usage

Use this Ant task to create or modify the specified library.

Required parameters

The following parameter is required:

- **name**: The name of the library. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: The cell of the resource action. The scope is global.
- **classpath**: Tells what class path entries to add. The scope is instance.
- **node**: The node of the resource action. The scope is global.
- **server**: The server of the resource action. The scope is global.

Parameters that are specified as nested Elements

attribute - a generic attribute for this resource. Users can use this attribute to specify any additional name-value pairs that are not already covered by the **wplc** task.

- **name**: The name of the attribute.
- **value**: The value of the attribute.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.LibraryCreateImpl** class implements this task.

wplc-remove-library

Description and usage

Use this Ant task to remove the specified library.

Required parameters

The following parameter is required:

- **name**: The name of the library. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: The cell of the resource action. The scope is global.
- **node**: The node of the resource action. The scope is global.
- **server**: The server of the resource action. The scope is global.

Parameters that are specified as nested Elements

attribute - a generic attribute for this resource. Users can use this attribute to specify any additional name-value pairs that are not already covered by the **wplc** task.

- **name**: The name of the attribute.
- **value**: The value of the attribute.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.LibraryRemoveImpl** class implements this task.

wplc-create-res-env-custom-property

Description and usage

Use this Ant task to create or reassign a Resource Environment Provider custom property.

Required parameters

The following parameters are required:

- **name**: The name of the resource to create, update, or modify. The scope is instance.
- **providerName**: The name of the resource environment provider. The scope is instance.
- **value**: The value of the custom property. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: The cell of the resource action. The scope is global.
- **description**: The description of the custom property. The scope is instance.
- **forceScope**: Set to `true` if do not want to scope to an existing cluster. The scope is instance.
- **node**: The node of the resource action. The scope is global.
- **required**: The Boolean value to specify whether this property is required or not. The scope is instance.
- **server**: The server of the resource action. The scope is global.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.ResourceEnvProviderCustomPropertyCreateImpl** class implements this task.

wplc-retrieve-res-env-custom-property

Description and usage

Use this Ant task to return the value of a Resource Environment Provider custom property.

Required parameters

The following parameters are required:

- **name**: The name of the resource to create, update, or modify. The scope is instance.
- **propToSet**: The property that is set in the Ant project with the value retrieved from this property. The scope is instance.
- **providerName**: The name of the resource environment provider. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: The cell of the resource action. The scope is global.
- **forceScope**: Set to `true` if do not want to scope to an existing cluster. The scope is instance.
- **node**: The node of the resource action. The scope is global.
- **server**: The server of the resource action. The scope is global.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.ResourceEnvProviderCustomPropertyRetreiveImpl** class implements this task.

wplc-remove-res-env-custom-property

Description and usage

Use this Ant task to remove a Resource Environment Provider custom property.

Required parameters

The following parameters are required:

- **name**: The name of the resource to create, update, or modify. The scope is instance.
- **providerName**: The name of the resource environment provider. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: The cell of the resource action. The scope is global.
- **forceScope**: Set to `true` if do not want to scope to an existing cluster. The scope is instance.
- **node**: The node of the resource action. The scope is global.
- **server**: The server of the resource action. The scope is global.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.ResourceEnvProviderCustomPropertyRemoveImpl** class implements this task.

wplc-create-res-env-entry

Description and usage

Use this Ant task to create or modify a resource environment entry.

Prerequisites

Run the **wplc-create-res-referenceable** task before you run the **wplc-create-res-env-entry** task. The resource entry that is referenced in the **wplc-create-res-env-entry** task is not automatically created. Instead, the **wplc-create-res-referenceable** task creates the resource.

Required parameters

The following parameters are required:

- **jndiName**: The JNDI name. The scope is instance.
- **name**: The name of the resource to create, update, or modify. The scope is instance.
- **providerName**: The name of the resource environment provider. The scope is instance.
- **referenceable.Class**: The class of the resource environment provider. The scope is instance.
- **referenceable.FactoryClass**: The factory class of the resource environment provider. The scope is instance.

Optional parameters

The following parameters are optional:

- **category**: The name of the category. The scope is instance.
- **cell**: The cell of the resource action. The scope is global.
- **description**: The description of the resource action. The scope is instance.
- **forceScope**: Set to `true` if do not want to scope to an existing cluster. The scope is instance.
- **node**: The node of the resource action. The scope is global.
- **server**: The server of the resource action. The scope is global.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.ResourceEnvEntryCreateImpl** class implements this task.

wplc-remove-res-env-entry

Description and usage

Use this Ant task to remove a resource environment entry.

Required parameters

The following parameters are required:

- **name**: The name of the resource to create, update, or modify. The scope is instance.
- **providerName**: The name of the resource environment provider. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: The cell of the resource action. The scope is global.
- **forceScope**: Set to `true` if do not want to scope to an existing cluster. The scope is instance.
- **node**: The node of the resource action. The scope is global.
- **server**: The server of the resource action. The scope is global.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.ResourceEnvEntryRemoveImpl** class implements this task.

wplc-create-res-env-entry-custom-property

Description and usage

Use this Ant task to create or reassign a custom SSL property in WebSphere Application Server security.

Required parameters

The following parameters are required:

- **entryName**: The name of the resource entry to update. The scope is instance.
- **name**: The name of the resource to create, update, or modify. The scope is instance.
- **providerName**: The name of the resource environment provider. The scope is instance.
- **value**: The value of the custom property. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: The cell of the resource action. The scope is global.
- **description**: The description of the custom property. The scope is instance.
- **forceScope**: Set to `true` if do not want to scope to an existing cluster. The scope is instance.
- **node**: The node of the resource action. The scope is global.
- **required**: The Boolean value to specify whether this property is required or not. The scope is instance.
- **server**: The server of the resource action. The scope is global.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.ResourceEnvEntryCustomPropertyCreateImpl** class implements this task.

wplc-remove-res-env-entry-custom-property

Description and usage

Use this Ant task to remove a custom SSL property in WebSphere Application Server security.

Required parameters

The following parameters are required:

- **entryName**: The name of the resource entry to update. The scope is instance.
- **name**: The name of the resource to create, update, or modify. The scope is instance.
- **providerName**: The name of the resource environment provider. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: The cell of the resource action. The scope is global.
- **forceScope**: Set to `true` if do not want to scope to an existing cluster. The scope is instance.
- **node**: The node of the resource action. The scope is global.
- **server**: The server of the resource action. The scope is global.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.ResourceEnvEntryCustomPropertyRemoveImpl** class implements this task.

wplc-create-resource-env-provider

Description and usage

Use this Ant task to create a resource environment provider at the specified scope.

Required parameters

The following parameter is required:

- **name**: The name of the resource to create, update, or modify. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: The cell of the resource action. The scope is global.
- **description**: The description of the custom property. The scope is instance.
- **forceScope**: Set to `true` if do not want to scope to an existing cluster. The scope is instance.
- **node**: The node of the resource action. The scope is global.
- **server**: The server of the resource action. The scope is global.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.ResourceEnvProviderCreateImpl** class implements this task.

wplc-remove-resource-env-provider

Description and usage

Use this Ant task to remove a resource environment provider at the specified scope and with the specified name.

Required parameters

The following parameter is required:

- **name**: The name of the resource to create, update, or modify. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: The cell of the resource action. The scope is global.
- **forceScope**: Set to `true` if do not want to scope to an existing cluster. The scope is instance.
- **node**: The node of the resource action. The scope is global.
- **server**: The server of the resource action. The scope is global.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.ResourceEnvProviderRemoveImpl** class implements this task.

wplc-create-res-referenceable

Description and usage

Use this Ant task to create or modify a resource entry that you can reference.

Important: Run this task before you run the **wplc-create-res-env-entry** task. The **wplc-create-res-referenceable** task creates the resource that is referenced in the **wplc-create-res-env-entry** task.

Required parameters

The following parameters are required:

- **providerName**: The name of the resource environment provider. The scope is instance.
- **referenceable.Class**: The reference class of the resource environment provider. The scope is instance.
- **referenceable.FactoryClass**: The reference factory class of the resource environment provider. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: Cell of the resource action. The scope is global.
- **forceScope**: Set to true if do not want to scope to cluster, if exists. The scope is instance.
- **node**: Node of the resource action. The scope is global.
- **server**: Server of the resource action. The scope is global.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.ResourceReferenceableCreateImpl** class implements this task.

wplc-remove-res-referenceable

Description and usage

Use this Ant task to remove a resource entry you can reference.

Required parameters

The following parameters are required:

- **providerName**: The name of the resource environment provider. The scope is instance.
- **referenceable.Class**: The reference class of the resource environment provider. The scope is instance.
- **referenceable.FactoryClass**: The reference factory class of the resource environment provider. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: Cell of the resource action. The scope is global.
- **forceScope**: Set to `true` if do not want to scope to an existing cluster. The scope is instance.
- **node**: Node of the resource action. The scope is global.
- **server**: Server of the resource action. The scope is global.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.ResourceReferenceableRemoveImpl** class implements this task.

wplc-create-variable

Description and usage

Use this Ant task to create or reassign a WebSphere Application Server environment variable and value.

Required parameters

The following parameters are required:

- **name**: The name of the variable. The scope is instance.
- **value**: The value that is assigned to the variable. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: The cell of the resource action. The scope is global.
- **description**: The description of the variable.
- **node**: The node of the resource action. The scope is global.
- **server**: The server of the resource action. The scope is global.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.VariableCreateImpl** class implements this task.

wplc-remove-variable

Description and usage

Use this Ant task to remove a WebSphere Application Server environment variable and value.

Required parameters

The following parameter is required:

- **name**: The name of the variable. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: The cell of the resource action. The scope is global.
- **node**: The node of the resource action. The scope is global.
- **server**: The server of the resource action. The scope is global.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.VariableRemoveImpl** class implements this task.

wplc-retrieve-variable

Description and usage

Use this Ant task to retrieve a WebSphere Application Server environment variable and value.

Required parameters

The following parameter is required:

- **name**: The name of the variable. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: The cell of the resource action. The scope is global.
- **node**: The node of the resource action. The scope is global.
- **server**: The server of the resource action. The scope is global.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.VariableRetrieveImpl** class implements this task.

wplc-create-res-env-directory

Description and usage
Use this Ant task to create RP file names in the directory.

Required parameters
The following parameter is required:

- **rpPropertyDirectoryName**: The property directory name. The scope is instance.

Optional parameters
The following parameters are optional:

- **cell**: Cell of the resource action. The scope is global.
- **forceScope**: Set to true if do not want to scope to an existing cluster. The scope is instance.
- **node**: Node of the resource action. The scope is global.
- **rpPrefix**: The name of the prefix. The scope is instance.
- **server**: Server of the resource action. The scope is global.

Implementation bean
The **com.ibm.wplc.deploy.tasks.impl.REProviderCreateByDirectoryImpl** class implements this task.

wplc-cluster-sync-all-node

Description and usage
Use this Ant task to sync all nodes in the cluster.

Optional parameters
The following parameters are optional:

- **cell**: Cell of the resource action. The scope is global.
- **node**: Node of the resource action. The scope is global.
- **server**: Server of the resource action. The scope is global.

Implementation bean
The **com.ibm.wplc.deploy.tasks.impl.ClusterSyncAllNodesImpl** class implements this task.

wplc-cluster-sync-single-node

Description and usage
Use this Ant task to sync a single cluster node.

Required parameters
The following parameter is required:

- **syncNode**: The name of the node to sync. The scope is instance.

Optional parameters
The following parameters are optional:

- **cell**: Cell of the resource action. The scope is global.
- **node**: Node of the resource action. The scope is global.
- **server**: Server of the resource action. The scope is global.

Implementation bean
The **com.ibm.wplc.deploy.tasks.impl.ClusterSyncSingleNodeImpl** class implements this task.

wplc-extract-ear-local-or-remote

Description and usage

Use this Ant task to extract ear files locally or remotely.

Required parameters

The following parameters are required:

- **appName**: The application name for the ear file. The scope is instance.
- **earFile**: The local directory and file where the ear file is exported to. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: Cell of the resource action. The scope is global.
- **node**: Node of the resource action. The scope is global.
- **server**: Server of the resource action. The scope is global.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.ExtractEarLocalOrRemoteImpl** class implements this task.

wplc-query-application-list

Description and usage

Use this Ant task to query a list of applications that are installed in the cell and to return a comma delimited string. The application list is placed in an Ant property called **${filteredAppList}**.

Optional parameters

The following parameters are optional:

- **cell**: Cell of the resource action. The scope is global.
- **filter**: The filter that is used for the application query; for example: PA_.
- **node**: Node of the resource action. The scope is global.
- **server**: Server of the resource action. The scope is global.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.ApplicationsQueryImpl** class implements this task.

wplc-retrieve-datasource-custom-property

Description and usage

Use this Ant task to retrieve a value from the data source custom properties.

Required parameters

The following parameters are required:

- **datasourcename**: Name of the data source. The scope is instance.
- **jdbcprovidername**: Name of the JDBC provider that identifies this resource. The scope is instance.

Optional parameters

The following parameters are optional:

- **cell**: Cell of the resource action. The scope is global.
- **forceScope**: Set to true if do not want to scope to the existing cluster. The scope is instance.
- **node**: Node of the resource action. The scope is global.

- **propertyName**: The name of the custom property to retrieve the value from. The scope is instance.
- **propToSet**: The property that is set in the Ant project with the value retrieved from WebSphere Application Server. The scope is instance.
- **server**: Server of the resource action. The scope is global.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.DataSourceRetrieveCustomPropertyValue** class implements this task.

wplc-wait-for-sync-to-complete

Description and usage

Use this Ant task to wait for the ear expansion and distribution to complete.

Optional parameters

The following parameters are optional:

- **cell**: Cell of the resource action. The scope is global.
- **maxAppTimeToWait**: The maximum time in minutes to wait for each application expansion process to complete.
- **maxTimeToWait**: The maximum time in minutes to wait for the expansion process to complete.
- **node**: Node of the resource action. The scope is global.
- **server**: Server of the resource action. The scope is global.
- **waitTime**: The time to wait between checks for the application to deploy.

Implementation bean

The **com.ibm.wplc.deploy.tasks.impl.ClusterWaitForSyncToCompleteImpl** class implements this task.

wplc-create-ear

Description and usage

Use this Ant task to deploy an EAR resource to the target Application or portal server.

Required parameters

The following parameters are required:

- **appname**: The name of the application for the EAR file.
- **earfile**: The location of the EAR file to deploy.

Optional parameters

The following parameters are optional:

- **cluster**: Defines the cluster name. Do not include spaces with this value. If not set, the value defaults to the context that is defined in the EAR file's `application.xml` file.
- **classloadermode**: Sets the class-loader delegation mode, also known as the class loader order. This value determines whether a class loader delegates the loading of classes to the parent class loader. The following values are supported:
  - `PARENT_FIRST`: Delegates the loading of classes to its parent before you load the class from its local path. This value is the default for the class-loader policy and for standard JVM class loaders.

– PARENT_LAST: Causes the class loader to attempt to load classes from its local class path before you delegate the class loading to its parent. Using this policy, an application class loader can override and provide its own version of a class that exists in the parent class loader.

- **startingweight**: Sets the starting weight for the application. The starting weight specifies the order in which applications are started when the server starts. The application with the lowest starting weight is started first.
- **server**: The name of the application server for WebSphere Portal Express.
- **node**: The name of the WebSphere Application Server node.
- **cell**: The name of the WebSphere Application Server cell.

Parameters that are specified as nested Elements

attribute - a generic attribute for this resource. Users can use this attribute to specify any additional name-value pairs that are not already covered by the **wplc** task. This parameter is used to pass any attribute that is associated with the resource action.

- **name**: The name of the attribute.
- **value**: The value of the attribute.

option - Not all available EAR deployment options are supported by the **wplc-create-ear** syntax as attributes. The **option** element passes more information to the underlying framework inside the **WsAdmin** profile. This feature is similar to how other parameters are passed into the **WsAdmin** script file. However, in this situation, the full set of options are assembled into a string corresponding to the syntax of the scripting language.
Do not code parameters that are included as attributes on the task as options. Use the corresponding attribute to avoid redundancy and to prevent runtime exception during the task parsing.
The syntax of the nested **option** element allows for the following three formats: keyword only, keyword/value pairing, or keyword with value list.

- **key**: The name of the option
- **value**: The value of the option. If no value is present and the option does not contain nested value elements, then this key does not take a value.

If a keyword can take a list of values, then a single nested **value** element is used for each value in the list.

Table 500. Examples of the keyword only, keyword/value pairing, or keyword with value list formats.. This table describes how to use the keyword only, keyword/value pairing, or keyword with value list formats for the **options** element.

Parameter type	Example (with AdminApp syntax)	Option element syntax
Keyword only	`-nodeployejb`	`<option key="nodeployejb"/>`
Keyword with simple value	`-custom value`	`<option key="custom" value="value"/>`
Keyword with list of values	`-MapresrefToEJB {{value1} {value2} {value3}}`	`<option key="MapResRefToEJB"/> <value>value1</value> <value>value2</value> <value>value3</value> </option>`

Implementation bean

The **com.ibm.wplc.deploy.tasks.EarCreate** class implements this task.

wplc-edit-ear

Description and usage

Use this Ant task to edit the attributes of the target application.

Required parameters

The following parameter is required:

- **appname**: The name of the application for the EAR file.

Parameters that are specified as nested Elements

option - Not all available EAR deployment options are supported by the **wplc-create-ear** syntax as attributes. The **option** element passes more information to the underlying framework inside the **WsAdmin** profile. This feature is similar to how other parameters are passed into the **WsAdmin** script file. However, in this situation, the full set of options are assembled into a string corresponding to the syntax of the scripting language.

Do not code parameters that are included as attributes on the task as options. Use the corresponding attribute to avoid redundancy and to prevent runtime exception during the task parsing.

The syntax of the nested **option** element allows for the following three formats: keyword only, keyword/value pairing, or keyword with value list.

- **key**: The name of the option
- **value**: The value of the option. If no value is present and the option does not contain nested value elements, then this key does not take a value.

If a keyword can take a list of values, then a single nested **value** element is used for each value in the list.

Table 501. Examples of the keyword only, keyword/value pairing, or keyword with value list formats.. This table describes how to use the keyword only, keyword/value pairing, or keyword with value list formats for the **options** element.

Parameter type	Example (with AdminApp syntax)	Option element syntax
Keyword only	`-nodeployejb`	`<option key="nodeployejb"/>`
Keyword with simple value	`-custom value`	`<option key="custom" value="value"/>`
Keyword with list of values	`-MapresrefToEJB {{value1} {value2} {value3}}`	`<option key="MapResRefToEJB"/> <value>value1</value> <value>value2</value> <value>value3</value> </option>`

Implementation bean

The **com.ibm.wplc.deploy.tasks.EarEdit** class implements this task.

wplc-update-ear

Description and usage

Use this Ant task to update an existing deployed EAR file with a new or modified version.

Required parameters

The following parameters are required:

- **appname**: The name of the application for the EAR file.
- **earfile**: The location of the EAR file to deploy.

Optional parameters

The following parameters are optional:

- **classloadermode**: Sets the class-loader delegation mode, also known as the class loader order. This value determines whether a class loader delegates the loading of classes to the parent class loader. The following values are supported:
  - PARENT_FIRST: Delegates the loading of classes to its parent before you load the class from its local class path. This value is the default for the class-loader policy and for standard JVM class loaders.
  - PARENT_LAST: Causes the class loader to attempt to load classes from its local class path before you delegate the class loading to its parent. Using this policy, an application class loader can override and provide its own version of a class that exists in the parent class loader.
- **startingweight**: Sets the starting weight for the application. The starting weight specifies the order in which applications are started when the server starts. The application with the lowest starting weight is started first.

Parameters that are specified as nested Elements

attribute - a generic attribute for this resource. Users can use this attribute to specify any additional name-value pairs that are not already covered by the **wplc** task. This parameter is used to pass any attribute that is associated with the resource action.

- **name**: The name of the attribute.
- **value**: The value of the attribute.

option - Not all available EAR deployment options are supported by the **wplc-create-ear** syntax as attributes. The **option** element passes more information to the underlying framework inside the **WsAdmin** profile. This feature is similar to how other parameters are passed into the **WsAdmin** script file. However, in this situation, the full set of options are assembled into a string corresponding to the syntax of the scripting language.
Do not code parameters that are included as attributes on the task as options. Use the corresponding attribute to avoid redundancy and to prevent runtime exception during the task parsing.
The syntax of the nested **option** element allows for the following three formats: keyword only, keyword/value pairing, or keyword with value list.

- **key**: The name of the option
- **value**: The value of the option. If no value is present and the option does not contain nested value elements, then this key does not take a value.

If a keyword can take a list of values, then a single nested **value** element is used for each value in the list.

Table 502. Examples of the keyword only, keyword/value pairing, or keyword with value list formats.. This table describes how to use the keyword only, keyword/value pairing, or keyword with value list formats for the **options** element.

Parameter type	Example (with AdminApp syntax)	Option element syntax
Keyword only	`-nodeployejb`	`<option key="nodeployejb"/>`
Keyword with simple value	`-custom value`	`<option key="custom" value="value"/>`
Keyword with list of values	`-MapresrefToEJB {{value1} {value2} {value3}}`	`<option key="MapResRefToEJB"/> <value>value1</value> <value>value2</value> <value>value3</value> </option>`

Implementation bean

The **com.ibm.wplc.deploy.tasks.EarUpdate** class implements this task.

wplc-modify-ear

Description and usage

Use this Ant task to modify a previously deployed EAR resource to the target application or Portal server instance.

Required parameters

The following parameters are required:

- **appname**: The name of the application for the EAR file.

Optional parameters

The following parameters are optional:

- **classloadermode**: Sets the class-loader delegation mode, also known as the class loader order. This parameter determines whether a class loader delegates the loading classes to the parent class loader. The following values are supported:
  - PARENT_FIRST: Delegates the loading of classes to its parent before you load the class from its local class path. This value is the default for the class-loader policy and for standard JVM class loaders.
  - PARENT_LAST: Causes the class loader to attempt to load classes from its local class path before you delegate the class loading to its parent. Using this policy, an application class loader can override and provide its own version of a class that exists in the parent class loader.
- **startingweight**: Set the starting weight for the application. The starting weight specifies the order in which applications are started when the server starts. The application with the lowest starting weight is started first.
- **server**: The name of the WebSphere Portal Express server.
- **node**: The name of the WebSphere Application Server node.
- **cell**: The name of the WebSphere Application Server cell.

Parameters that are specified as nested Elements

attribute - a generic attribute for this resource. Users can use this attribute to specify any additional name-value pairs that are not already covered by the **wplc** task.

- **name**: The name of the attribute.
- **value**: The value of the attribute.

option - Not all available EAR deployment options are supported by the **wplc-create-ear** syntax as attributes. The **option** element passes more information to the underlying framework inside the **WsAdmin** profile. This feature is similar to how other parameters are passed into the **WsAdmin** script file. However, in this situation, the full set of options are assembled into a string corresponding to the syntax of the scripting language.
Do not code parameters that are included as attributes on the task as options. Use the corresponding attribute to avoid redundancy and to prevent runtime exception during the task parsing.
The syntax of the nested **option** element allows for the following three formats: keyword only, keyword/value pairing, or keyword with value list.

- **key**: The name of the option

- **value**: The value of the option. If no value is present and the option does not contain nested value elements, then this key does not take a value.

If a keyword can take a list of values, then a single nested **value** element is used for each value in the list.

Table 503. Examples of the keyword only, keyword/value pairing, or keyword with value list formats.. This table describes how to use the keyword only, keyword/value pairing, or keyword with value list formats for the **options** element.

Parameter type	Example (with AdminApp syntax)	Option element syntax
Keyword only	`-nodeployejb`	`<option key="nodeployejb"/>`
Keyword with simple value	`-custom value`	`<option key="custom" value="value"/>`
Keyword with list of values	`-MapresrefToEJB {{value1} {value2} {value3}}`	`<option key="MapResRefToEJB"/> <value>value1</value> <value>value2</value> <value>value3</value> </option>`

Implementation bean

The **com.ibm.wplc.deploy.tasks.EarModify** class implements this task.

wplc-update-file-in-ear

Description and usage

Use this Ant task to update a file in the deployed ear.

Required parameters

The following parameters are required:

- **appname**: The name of the application for the EAR file.
- **classloadermode**: Sets the class-loader delegation mode, also known as the class loader order. This value determines whether a class loader delegates the loading of classes to the parent class loader. The following values are supported:
  - PARENT_FIRST: Delegates the loading of classes to its parent before you load the class from its local class path. This value is the default for the class-loader policy and for standard JVM class loaders.
  - PARENT_LAST: Causes the class loader to attempt to load classes from its local class path before you delegate the class loading to its parent. Using this policy, an application class loader can override and provide its own version of a class that exists in the parent class loader.
  - **startingweight**: Sets the starting weight for the application. The starting weight specifies the order in which applications are started when the server starts. The application with the lowest starting weight is started first.

Parameters that are specified as nested Elements

option - Not all available EAR deployment options are supported by the **wplc-create-ear** syntax as attributes. The **option** element passes more information to the underlying framework inside the **WsAdmin** profile. This feature is similar to how other parameters are passed into the **WsAdmin** script file. However, in this situation, the full set of options are assembled into a string corresponding to the syntax of the scripting language.

Do not code parameters that are included as attributes on the task as options. Use the corresponding attribute to avoid redundancy and to prevent runtime exception during the task parsing.

The syntax of the nested **option** element allows for the following three formats: keyword only, keyword/value pairing, or keyword with value list.

- **key**: The name of the option
- **value**: The value of the option. If no value attribute is present and the option does not contain value elements, then this key does not take a value.

If a keyword can take a list of values, then a single nested **value** element is used for each value in the list.

Add the following two <option> keys:

- An option element with a key of `contents` points to the location on disk of the file to update `option key="contents" value="locationOfFileOnDisk"/>`.
- An option element with a key of `contenturi` points to the location inside of an `.ear` file of the file to update `option key="contenturi" value="locationOfFileInEar"/>`

Implementation bean

The **com.ibm.wplc.deploy.tasks.EarCreate** class implements this task.

wplc-remove-ear

Description and usage

Use this Ant task to remove or uninstall an EAR resource from the target application or Portal server instance.

Required parameters

The following parameter is required:

- **appname**: The name of the application for the EAR file.

Implementation bean

The **com.ibm.wplc.deploy.tasks.EarRemove** class implements this task.

IBM UX Screen Flow Manager

The IBM UX Screen Flow Manager helps operators, developers, and dialog modelers develop fine-granular, small split portlets. Portlets provide the functions for the steps that are presented as screens. You can configure the sequence, transitions, and workflow of a set of screens. Screen flows work like wizards and process the screens to complete the task for the user. Screen flows are completed by a single user and have a short lifetime.

For example, screen flow modelers can model flows for processing the sequence of steps that are involved in booking a trip for a travel site. Booking for a trip might include steps such as gathering traveler information, booking flight, booking hotel, and booking car. Each of these steps might include substeps such as displaying the customer information, and updating customer information. Each of these steps is presented as screens and the functions behind them are provided by portlets. These portlets are interconnected and managed by IBM WebSphere Portal. The screen flow configurations route users along paths that interconnect user interface artifacts, such as forms or masks, to accomplish specific tasks. The mapping of individual screens to portlets affects both user experience and reusability. This approach provides both strict user guidance and high reusability.

Screen flows can be used together with business workflows. Long-running business processes that are run by a business workflow can trigger screen flows that are run by portal. After the screen flow completes, the portal gives the control

back to the business workflow. Data can be exchanged between the business workflow and the screen flow on the portal. If a user suspends a screen flow, the portal can hand over the processed data over to the business workflow to save it.

"Developing screen flows"
To develop screen flows you need to create user interface artifacts, interconnect the artifacts and deploy the artifacts.

"Advanced concepts" on page 3506
Learn about the advanced concepts of the IBM UX Screen Flow Manager.

CF05 "User interface components" on page 3518
In a default WebSphere Portal installation, the Dialog Stack and Dialog State Display portlets are deployed. The following topics describe how these portlets function.

"Configuration options" on page 3525
To change the overall behavior of the IBM UX Screen Flow Manager, several configuration options are available. You specify the options as Resource Environment Provider (REP) properties.

"Staging and migration" on page 3526
For staging or migration purposes, you can use the portal XML configuration interface (XMLAccess) to transfer IBM UX Screen Flow Manager related data from one system to another.

CF05 "Transitions reference" on page 3526
You can configure transitions in multiple ways. For example, with single portlets as source, you can configure it to transition to targets such as single portlets, multiple portlets through single or multiple transition endpoints, single page, or mixed resources. Similarly, you can configure single portlets, multiple portlets, single page, or mixed resources to become the source and transition to the target single portlet. The following reference topics show the code samples for these transitions.

"Sample screen flow application" on page 3534
The IBM UX Screen Flow Manager package can be downloaded from the Greenhouse site and includes a sample screen flow application that is ready to use. This sample guides a portal site visitor through a simple travel booking flow. It consists of a set of pages, portlets, and dialog definitions (DDs). For more details about DDs, go to Creating dialog definitions.

Related information:

Product documentation

Developing screen flows

To develop screen flows you need to create user interface artifacts, interconnect the artifacts and deploy the artifacts.

About this task

To develop screen flows, do the following steps:

Procedure

1. Develop the necessary user interface artifacts that when combined define the appropriate screen flow. These user interface artifacts are usually portlets that can send and receive JSR 286 events.
2. Create the dialog definition that defines all interconnections between the user interface artifacts that were created in the previous step.
3. Deploy all user interface artifacts and the dialog definition.

"Developing user interface artifacts"
Every screen flow needs a starting point from which it can be triggered. The user interface components that you develop and their functions enable the screen flow to be triggered and processed. Independent of the type of user interface artifact you develop, it must send and receive JSR-286 events to enable a screen flow. Therefore, you must implement or reuse JSR-286 compliant portlets that send and receive such events.

"Creating dialog definitions" on page 3479
With the Screen Flow Manager, different teams or even third-party vendors can develop different types of user interface artifacts. The user puts together the correct set of user interface artifacts and creates the declarative model in XML known as the dialog definition. The dialog definition describes the specific screen flow that is also known as dialog, which consists of multiple steps and single steps referred to as subdialogs. The dialog definition contains all information about the subdialogs that participate and the transitions that route the user from one subdialog to another.

"Deploying user interface artifacts" on page 3505
Before you deploy the Screen Flow Manager dialog definitions, you must deploy all portal resources that are part of the dialog definitions, such as pages, and portlets. You can then deploy the dialog definitions by using the portal XML configuration interface (XMLAccess).

Developing user interface artifacts

Every screen flow needs a starting point from which it can be triggered. The user interface components that you develop and their functions enable the screen flow to be triggered and processed. Independent of the type of user interface artifact you develop, it must send and receive JSR-286 events to enable a screen flow. Therefore, you must implement or reuse JSR-286 compliant portlets that send and receive such events.

For example, assume in a travel site for a Flight booking page, the Passenger information portlet `portlet1` and the Calendar portlet `portlet2` are two portlets that you developed.

- The Passenger information portlet `portlet1` must be able to receive an event with the QName `e0` and to send an event with the QName `e1`.
- The Calendar portlet `portlet2` must be able to receive an event with the QName `e1` and to send an event with the QName `e2`.

When the Passenger information portlet `portlet1` sends the event `e1`, the user is routed from the Passenger information portlet `portlet1` to the Calendar portlet `portlet2`. For more information, go to *Transitions*.

To enable the Passenger information portlet `portlet1` to receive and send the mentioned events, you need to do the following two things:

- Specify the events in the `portlet.xml` file of the portlet.

```
<portlet id="portlet1">
 ...
  <supported-processing-event>
      <qname>e0</qname>
  </supported-processing-event>
  <supported-publishing-event>
      <qname>e1</qname>
  </supported-publishing-event>
 </portlet>

 <event-definition>
  <qname>e0</qname>
```

```
 <value-type>java.lang.String</value-type>
</event-definition>
<event-definition>
 <qname>e1</qname>
 <value-type>java.lang.String</value-type>
</event-definition>
```

- Include code in the portlet that can handle an incoming event and another code that can send an event.

```
@Override
public void processAction(ActionRequest request, ActionResponse response)
     throws PortletException, IOException {
 // ...

 response.setEvent(new QName("e1", "xyz"), new String());
}

@Override
public void processEvent(EventRequest request, EventResponse response)
     throws PortletException, IOException {

 final Event event = request.getEvent();
 // ...
```

Make similar changes to the Calendar portlet `portlet 2`.

```
<portlet id="portlet2">
 ...
 <supported-processing-event>
     <qname>e1</qname>
 </supported-processing-event>
 <supported-publishing-event>
     <qname>e2</qname>
 </supported-publishing-event>
</portlet>

<event-definition>
 <qname>e1</qname>
 <value-type>java.lang.String</value-type>
</event-definition>
<event-definition>
 <qname>e2</qname>
 <value-type>java.lang.String</value-type>
</event-definition>
```

```
@Override
public void processAction(ActionRequest request, ActionResponse response)
     throws PortletException, IOException {
 // ...

 response.setEvent(new QName("e2", "xyz"), new String());
}

@Override
public void processEvent(EventRequest request, EventResponse response)
     throws PortletException, IOException {

 final Event event = request.getEvent();
 // ...
```

If you want to integrate forms such as passenger information form or widgets such as calendar, you can use such a portlet as a wrapper for these artifacts. For more information about developing portlets, go to *Developing portlets in the WebSphere® Portal Version 8.0 product documentation.*

The portlets that can start screen flows are referred to as Dialog Instantiation and Initialization portlets (DIIPs). You can usually distinguish between two kinds of DIIPs:

- One type of DIIP triggers a new dialog instance by sending a well-defined start-event.
- The other type of DIIP triggers a new dialog that is based on the fact that a specific portlet emits a specific custom event.

In either case, the event emission causes a new screen flow to be started and initialized with some initial data that the event payload transmits.

With the Screen Flow Manager, different teams or even third-party vendors can develop different types of user interface (UI) artifacts. These artifacts are usually portlets, but can also be widgets or forms. For more details, go to *iWidgets Development in the IBM® Rational® Application Developer documentation*.

Related concepts:

"Transitions" on page 3487
Transitions define how to route a user from one subdialog to another. As subdialogs are represented by pages or portlets, they reference transition-endpoints. For example, in a travel site, a user can be routed from the Passenger information subdialog or portlet to the Calendar subdialog or portlet.

Related information:

Developing portlets

iWidget development overview

Creating dialog definitions

With the Screen Flow Manager, different teams or even third-party vendors can develop different types of user interface artifacts. The user puts together the correct set of user interface artifacts and creates the declarative model in XML known as the dialog definition. The dialog definition describes the specific screen flow that is also known as dialog, which consists of multiple steps and single steps referred to as subdialogs. The dialog definition contains all information about the subdialogs that participate and the transitions that route the user from one subdialog to another.

Dialogs

Dialogs, also known as dialog definitions, define all the artifacts of which a single dialog is comprised. They define the resources such as pages and portlets, also called transition endpoints, that are part of the specific dialog. Dialog definitions also define the transitions that route a user from one subdialog to another. For more information, go to *Transition endpoints* and *Transitions*. You can define dialog definitions by using a unique name. For example, in a travel site, the flight booking step can be described as a dialog, which defines artifacts such as Passenger information portlet, and Travel dates portlet.

```
<dialog-set>
    <dialog name="dialog1">
        <transition-endpoint name="portlet1">
            ...
        </transition-endpoint>
        <transition-endpoint name="portlet2">
            ...
        </transition-endpoint>
        <transition>
            ...
```

```
        </transition>
        <transition>
            ...
        </transition>
    </dialog>
</dialog-set>
```

During run time, dialog definitions form the templates from which concrete dialog instances are created. These instances are uniquely identified by `DialogInstanceIDs`.

Dialogs are scoped to virtual portals. This scoping means that a screen flow that is running in one virtual portal does not show in another virtual portal or vice versa.

Dialog sets

Dialog sets can contain one or more screen flow definitions. For example, in a travel site the Flight booking dialog and Car booking dialog that working together can be referred to as Dialog sets.

```
<dialog-set>
 <dialog name="dialog1">
     ...
 </dialog>
 <dialog name="dialog2">
     ...
 </dialog>
</dialog-set>
```

"Transition endpoints"
Resources that are part of a screen flow which marks an endpoint, that is, the source or target of a transition are referred to as transition endpoints. Resources can be pages and portlets and they can also wrap widgets or forms. A particular page or portlet that is the active step in a screen flow is referred to as the source and the potential next steps are referred to as targets.

"Transitions" on page 3487
Transitions define how to route a user from one subdialog to another. As subdialogs are represented by pages or portlets, they reference transition-endpoints. For example, in a travel site, a user can be routed from the Passenger information subdialog or portlet to the Calendar subdialog or portlet.

Related concepts:

"Transition endpoints"
Resources that are part of a screen flow which marks an endpoint, that is, the source or target of a transition are referred to as transition endpoints. Resources can be pages and portlets and they can also wrap widgets or forms. A particular page or portlet that is the active step in a screen flow is referred to as the source and the potential next steps are referred to as targets.

"Transitions" on page 3487
Transitions define how to route a user from one subdialog to another. As subdialogs are represented by pages or portlets, they reference transition-endpoints. For example, in a travel site, a user can be routed from the Passenger information subdialog or portlet to the Calendar subdialog or portlet.

Transition endpoints:

Resources that are part of a screen flow which marks an endpoint, that is, the source or target of a transition are referred to as transition endpoints. Resources can be pages and portlets and they can also wrap widgets or forms. A particular

page or portlet that is the active step in a screen flow is referred to as the source and the potential next steps are referred to as targets.

For example, in a travel site the Flight booking dialog contains resources such as the Passenger information portlet and Calendar portlet which wraps the Calendar widget. When the dialog defines a transition that route the user from entering Passenger information step to selecting the travel dates step, the Passenger information portlet and the Calendar portlet become the source and target of the transition.

A source can reference only one single transition-endpoint that is associated with only one single event. A target can reference multiple transition-endpoints that are associated with one or multiple events. There are different options available to reference resources.

CF05 "Referencing portlets"
A transition endpoint can reference portlets as the target of a screen flow transition.

CF05 "Reference pages" on page 3482
A transition endpoint can reference pages as the target of a screen flow transition.

CF05 "Referencing dialogs" on page 3484
A transition endpoint can reference dialog as the source of a screen flow transition.

"Referencing single resource across different dialogs" on page 3485
Often, a single resource is used across different dialogs. For example, such a resource can be a generic date selection portlet. In a travel site, such a page can be used to select the departure date and the return date for the Flight booking dialog and also for the Car booking dialog.

CF05 "Referencing through metadata markers" on page 3486
Pages and portlets can be referenced not only by their unique names but can also be referenced through metadata markers.

CF05 "Mixed referencing" on page 3487
The mechanisms for referencing a single page, a single portlet, multiple pages, or multiple portlets by unique names or metadata markers can be mixed.

CF05 "Limitations when referencing resources" on page 3487
Not all mechanisms for referencing resources are allowed to be used as part of both, a transitions source and a transitions target.

Referencing portlets:

A transition endpoint can reference portlets as the target of a screen flow transition.

Reference a single portlet

The simplest form of a transition endpoint that is used within a transitions source is a transition endpoint that references a single portlet. In that case, the portlet is referenced through its unique name. For example, in a travel site, the transition endpoint can reference a Calendar portlet as the target in a transition with Passenger information portlet as the source.

A transition with a source that points to a transition endpoint that references a single portlet is triggered. The transition is triggered when the particular portlet that is referenced emits the event that is defined as part of the transitions source. For more information, see *Transitions*.

```
<dialog name="dialog1">
 <transition-endpoint name="portlet1">
      <localedata locale="en">
           <title>Subdialog 1</title>
           <description>This is a subdialog</description>
       </localedata>
       <resource uniquename="uniquename.portlet1"/>
       <invocation type="static"/>
...
  </transition-endpoint>
```

Reference multiple portlets

It is also possible to reference multiple portlets as part of a single transition endpoint. For example, the transition with Passenger information portlet as the source can point to Calendar portlets, and Destination portlets as references in a transition endpoint.

A transition with a source that points to a transition endpoint that references multiple portlets is triggered. The transition is triggered when any of the referenced portlets emit the event that is defined as part of the transitions source.

```
<dialog name="dialog1">
 <transition-endpoint name="portlet1_2_3">
      <localedata locale="en">
          <title>Subdialog 1</title>
          <description>This is a subdialog</description>
       </localedata>
       <resource uniquename="uniquename.portlet1"/>
       <resource uniquename="uniquename.portlet2"/>
       <resource uniquename="uniquename.portlet3"/>
       <invocation type="static"/>
...
</transition-endpoint>
```

Related concepts:

CF05 "Transitions" on page 3487
Transitions define how to route a user from one subdialog to another. As subdialogs are represented by pages or portlets, they reference transition-endpoints. For example, in a travel site, a user can be routed from the Passenger information subdialog or portlet to the Calendar subdialog or portlet.

Reference pages:

A transition endpoint can reference pages as the target of a screen flow transition.

Reference a single page

A transition with a source that points to a transition endpoint that references a single page is triggered. The transition is triggered when any portlet on the particular page that is referenced emits the event that is defined as part of the transitions source. For example, in a travel site, in a Flight booking dialog the transition with Calendar portlet as the source can reference the departure date page as the target.

```
<dialog name="dialog1">
   <transition-endpoint name="page1">
        <localedata locale="page1">
```

```
            <title>Subdialog 1</title>
            <description>This is a subdialog</description>
        </localedata>
        <resource uniquename="uniquename.page1"/>
        <invocation type="static"/>
...
</transition-endpoint>
```

Reference multiple pages

As with portlets, it is also possible to reference multiple pages as part of a single transition-endpoint. For example, in a Flight booking dialog, the transition with Calendar portlet as the source can reference the departure date page and return date page as targets.

A transition with a source that points to a transition-endpoint that references multiple pages is triggered. The transition is triggered when any portlet on any of the pages that are referenced emits the event. The emitted event must be defined as part of the transitions source.

```
<dialog name="dialog1">
   <transition-endpoint name="page1_2_3">
         <localedata locale="en">
               <title>Subdialog 1</title>
               <description>This is a subdialog</description>
          </localedata>
          <resource uniquename="uniquename.page1"/>
          <resource uniquename="uniquename.page2"/>
          <resource uniquename="uniquename.page3"/>
          <invocation type="static"/>
...
</transition-endpoint>
```

Reference page hierarchies

A portlet can consist of a well-defined set of topologically connected pages. The pages are part of a connected graph. In some scenarios, you might want a transition to be triggered when any portlet on any page that is part of a connected graph emits the event. The event that is emitted must be defined as part of the transition's source.

To list all the pages, you can point to a page hierarchy by referencing the hierarchy's root page. You can reference the root page through its unique name and with the optional attribute `type` with value `hierarchy`.

A transition with a source that points to a transition-endpoint that references a page hierarchy is triggered. The transition is triggered when any page that is a direct or indirect child of the referenced root page emits the event. The event that is emitted must be defined as part of the transitions source.

For example, in a travel site, in a Flight booking dialog, the transition can point to the Destination portlet. The Destination portlet can consist of several pages that provide data on possible destination locations the user can choose. To list all the pages in the destination portlet the transition can reference the hierarchy root page, for example, Europe.

```
<dialog name="dialog1">
  <transition-endpoint name="pageHierarchy1">
       <localedata locale="en">
            <title>Subdialog 1</title>
            <description>This is a subdialog</description>
```

```
        </localedata>
        <resource uniquename="uniquename.page1" type="hierarchy" />
        <invocation type="static"/>
...
</transition-endpoint>
```

Referencing dialogs:

A transition endpoint can reference dialog as the source of a screen flow transition.

Reference a single dialog

It is possible to reference a single dialog as part of a transition-endpoint. For example, in a travel site, the steps that are needed to book a flight might represent a single dialog and booking a car might represent another dialog. A transition endpoint can reference the Flight booking dialog or the Car booking dialog.

If a transition defines a target that points to a transition-endpoint that references a dialog, the transition is referred to as outgoing transition. An outgoing transition represents a transition that starts when it leaves the calling dialog and enters the called one. For example, if the Car booking dialog is defined as the target in a transition and the Flight booking dialog is defined as the source, the Flight booking dialog would be the calling dialog and the Car booking dialog would be the called one. Therefore, an outgoing transition is one that starts when it leaves the Flight booking dialog and enters the Car booking dialog.

If a transition defines a source that points to a transition-endpoint that references a dialog, the transition is referred to as incoming transition. An incoming transition represents a transition that starts when it returns from the called dialog and continues with the calling one. For example, if the Flight booking dialog is referenced as the source, a transition that returns from the target Car booking, the called dialog to the source Flight booking, the calling dialog and continues with the steps then it is an incoming transition.

For more information about transitions, go to *Transitions* and for more information about starting dialogs from within other dialogs, go to *Dialog chaining and nesting*.

```
<dialog name="dialog1>
 <transition-endpoint name="dialog2">
      <localedata locale="en">
           <title>Dialog 2</title>
           <description>This is dialog 2</description>
      </localedata>
      <resource uniquename="dialog2"/>
      <invocation type="static"/>
...
 </transition-endpoint>
```

Related concepts:

CF05 "Transitions" on page 3487
Transitions define how to route a user from one subdialog to another. As subdialogs are represented by pages or portlets, they reference transition-endpoints. For example, in a travel site, a user can be routed from the Passenger information subdialog or portlet to the Calendar subdialog or portlet.

CF05 "Dialog chaining and nesting" on page 3494
Dialogs can start other dialogs. Dialogs that start other dialogs are referred to as calling dialogs and the dialogs that are being started are referred to as called dialogs. Two different options are available to complete this action such as dialog chaining and dialog nesting.

Referencing single resource across different dialogs:

Often, a single resource is used across different dialogs. For example, such a resource can be a generic date selection portlet. In a travel site, such a page can be used to select the departure date and the return date for the Flight booking dialog and also for the Car booking dialog.

Portlets can have only one dedicated title per locale. For example, the portlet might be called Calendar. Therefore, the dialog must make the descriptions clear to the site visitor who books a trip. The portlet must clearly show when this portlet is used for selecting the departure date, and when it is used for selecting the return date for the trip.

If no explicit titles and descriptions were specified for a transition endpoint as part of the dialog definition. The Dialog State Display (DSD) shows the step with the localized title of the referenced resource.

Thus the following applies:

- If the transition endpoint references a page, the localized page title is displayed.
- If the transition endpoint references a portlet, the localized portlet title is displayed.

But this fallback mechanism still does not provide possibilities for displaying different titles for the same resource during different steps. To resolve this problem, dialog modelers can also explicitly specify localized titles and descriptions for transition endpoints. The DSD then displays these titles and descriptions instead of the original title and description of the resource. Therefore, you can define two transition endpoints that point to the same resource but have different titles and descriptions or localized titles and descriptions for that resource.

The example references the same Calendar portlet twice as part of two separate transition endpoints. In one transition endpoint, the English title is set to `Date to leave`, in the other one to `Date to return`. This way, the DSD can display the same portlet with different titles or localized titles.

```
<dialog name="dialog1">
 <transition-endpoint name="calendar.leave">
     <localedata locale="en">
         <title>Date to leave</title>
         <description>Specify the date to leave</description>
     </localedata>
     <localedata locale="de">
         <title>Abreisedatum</title>
         <description>Geben Sie Ihr Abreisedatum an</description>
     </localedata>
     <resource uniquename="uniquename.calendar"/>
     <invocation type="static"/>
</transition-endpoint>
 <transition-endpoint name="calendar.return">
     <localedata locale="en">
        <title>Date to return</title>
         <description>Specify the date to return</description>
     </localedata>
     <localedata locale="de">
         <title>Rueckreisedatum</title>
         <description>Geben Sie Ihr Rueckreisedatum an</description>
     </localedata>
     <resource uniquename="uniquename.calendar"/>
     <invocation type="static"/>
 </transition-endpoint>
```

```
  ...
  <transition>
      <source>
          <transition-endpoint nameref="portlet1">
              <event qname="e1"/>
          </transition-endpoint>
      </source>
      <target>
          <transition-endpoint nameref="calendar.leave">
              <event qname="ecl"/>
          </transition-endpoint>
      </target>
  </transition>
  ...
  <transition>
     <source>
          <transition-endpoint nameref="portlet2">
              <event qname="e2"/>
          </transition-endpoint>
     </source>
      <target>
          <transition-endpoint nameref="calendar.return">
              <event qname="ecr"/>
          </transition-endpoint>
      </target>
  </transition>
  ...
 </dialog>
```

Referencing through metadata markers:

Pages and portlets can be referenced not only by their unique names but can also be referenced through metadata markers.

A transition has a source that points to a transition-endpoint. The transition-endpoint references one or multiple portlets through metadata markers. When the particular portlet or any of the portlets that are referenced emits the event, the transition is triggered. The event that is emitted is defined as part of the transition's source.

Alternatively, a transition has a source that points to a transition-endpoint. The transition-endpoint references one or multiple pages through metadata markers. When any portlet on the particular page or on any of the pages that are referenced emits the event, the transition is triggered. The event that is emitted is defined as part of the transitions source.

Code sample shows an example where resources, either pages or portlets, that are assigned metadata with keys *uxfm.marker.1* and *uxfm.marker.2* are being referenced.

```
<dialog name="dialog1">
  <transition-endpoint name="marked_resources">
      <localedata locale="en">
          <title>Subdialog 1</title>
          <description>This is a subdialog</description>
       </localedata>
       <resource metadat="uxfm.marker.1"/>
       <resource metadata="uxfm.marker.2"/>
       <invocation type="static"/>
...
</transition-endpoint>
```

Note: Marking portlet definitions instead of portlet windows lead to a different behavior. For more information, go to the topic *Dynamic pages and portlets*

Related concepts:

CF05 “Dynamic pages and portlets” on page 3516
The IBM UX Screen Flow Manager not only supports redirecting users between static portal resources, but also between dynamic resources. The WebSphere Portal feature Dynamic UI Management is used.

Mixed referencing:

The mechanisms for referencing a single page, a single portlet, multiple pages, or multiple portlets by unique names or metadata markers can be mixed.

Example of mixed referencing.

```
<dialog name="dialog1">
 <transition-endpoint name="mixed">
     <localedata locale="en">
           <title>Subdialog 1</title>
           <description>This is a subdialog</description>
      </localedata>
      <resource uniquename="uniquename.portlet1"/>
      <resource uniquename="uniquename.page1"/
```

Limitations when referencing resources:

Not all mechanisms for referencing resources are allowed to be used as part of both, a transitions source and a transitions target.

Table 1 provides an overview of what referencing mechanism is allowed and disallowed.

Table 504. Overview of referencing mechanisms

Referencing Mechanism	Allowed within a transitions's source	Allowed within a transitions target
Single portlet	✓	✓
Multiple portlets	✓	✓
Single page	✓	✓
Multiple pages	✓	
Page hierarchies	✓	
Metadata marker	✓	
Single dialog	✓	✓

Transitions:

Transitions define how to route a user from one subdialog to another. As subdialogs are represented by pages or portlets, they reference transition-endpoints. For example, in a travel site, a user can be routed from the Passenger information subdialog or portlet to the Calendar subdialog or portlet.

Transitions are consisted of two main subsections. One subsection defines a source and the other subsection defines a target. Sources and targets both reference transition-endpoints that are associated with events.

The following steps show how the user is routed from one subdialog to the other.

- The user is routed from the subdialog that is represented by the transition-endpoint. This transition-endpoint is referenced by the source.

- After the source's transition-endpoint emits the associated event, the user is routed from that subdialog.
- The user is routed to the subdialog that holds one or multiple transition-endpoints that are referenced by the target.
- The transition-endpoints that are routed from the source are then fed with the associated events of the target endpoints.

You can configure transitions in multiple ways, for example

- With single portlets as source, you can configure it to transition to targets such as single portlets, multiple portlets through single or multiple transition endpoints, single page, or mixed resources.
- Similarly you can configure single portlets, multiple portlets, single page or mixed resources to become the source and transition to the target single portlet.

For more information and example code samples of these transitions, see *Transitions reference* section.

"Start and end transitions" on page 3489
When the referenced transition endpoint and event matches the source of a start transition, dialogs are started. Start transitions differ from other transitions only in that they carry the attribute `type`, with the value `start`. Similarly, end transitions carry the attribute `type`, with the value `end`.

"Start transitions and wildcards" on page 3489
You can define a dialog to start if one of several source transition endpoints emits a specific event. In this case, the start of the dialog depends only on the event and not on the referenced transition endpoint and event.

"Start transitions and special events" on page 3490
You can define a dialog to starts if a special start event is emitted from a specific source or from an arbitrary source.

"End transitions and special events" on page 3491
Similar to the special start event, end transitions can emit a special end event.

"Other special transitions" on page 3492
IBM UX Screen Flow Manager supports several more special transitions, or more precisely, transition endpoints.

"Exclusive transitions or dialogs" on page 3494
Exclusive dialogs are special forms of dialogs. They are also referred to as null or one-step dialogs. They consist of a single transition only. You can use them to go into a dialog, run only one transition, and use it to end the dialog immediately afterward.

CF05 "Dialog chaining and nesting" on page 3494
Dialogs can start other dialogs. Dialogs that start other dialogs are referred to as calling dialogs and the dialogs that are being started are referred to as called dialogs. Two different options are available to complete this action such as dialog chaining and dialog nesting.

"Rules and restrictions" on page 3501
When you model a transition certain rules and restrictions apply.

CF05 "Execution priority" on page 3502
You can and must explicitly model and enforce a deterministic behavior for the transitions. Even if the entire set of dialogs is deterministic, the emission of an event by a particular portlet can trigger two different transitions of two different dialogs. Therefore, you must explicitly model the intended behavior of the transitions.

"Deploy dialog sets by using the XML configuration interface" on page 3504
You can use the portal XML configuration interface (XMLAccess) to work with dialog sets.

Related concepts:

"Transition endpoints" on page 3480
Resources that are part of a screen flow which marks an endpoint, that is, the source or target of a transition are referred to as transition endpoints. Resources can be pages and portlets and they can also wrap widgets or forms. A particular page or portlet that is the active step in a screen flow is referred to as the source and the potential next steps are referred to as targets.

Related reference:

CF05 "Transitions reference" on page 3526
You can configure transitions in multiple ways. For example, with single portlets as source, you can configure it to transition to targets such as single portlets, multiple portlets through single or multiple transition endpoints, single page, or mixed resources. Similarly, you can configure single portlets, multiple portlets, single page, or mixed resources to become the source and transition to the target single portlet. The following reference topics show the code samples for these transitions.

Start and end transitions:

When the referenced transition endpoint and event matches the source of a start transition, dialogs are started. Start transitions differ from other transitions only in that they carry the attribute `type`, with the value `start`. Similarly, end transitions carry the attribute `type`, with the value `end`.

To avoid transitions that can accidentally start or end a screen flow, dialog modelers need to define start and end transitions. Each dialog needs to define at least one start and at least one end transition.

```
<transition type="start">
    <source>
        ...
    </source>
    <target>
        ...
    </target>
</transition>
...
<transition type="end">
    <source>
        ...
    </source>
    <target>
        ...
    </target>
</transition>
```

Start transitions and wildcards:

You can define a dialog to start if one of several source transition endpoints emits a specific event. In this case, the start of the dialog depends only on the event and not on the referenced transition endpoint and event.

Syntactically, you describe an arbitrary source transition endpoint by referencing the transition endpoint with an asterisk (`*`). In this sample, the dialog `dialog1` is started whenever one of the participating portlets emits the event `e1`.

Note: Using this function increases the risk of modeling non-deterministic dialog sets. You must not use an event that is combined with an undetermined source as part of any other dialog's start transition source.

```
<transition>
    <source>
        <transition-endpoint nameref="*">
            <event qname="e1"/>
        </transition-endpoint>
    </source>
    <target>
        ...
    </target>
</transition>
```

Start transitions and special events:

You can define a dialog to starts if a special start event is emitted from a specific source or from an arbitrary source.

This special start event is identified by the following `QName:`

`{http://www.ibm.com/xmlns/prod/websphere/portal/v6.1.0/portal-pcm}StartDialog`

Its payload needs to implement the interface `com.ibm.portal.pcm.events.DialogStartPayload`.

This special start event provides extra ways to start dialogs or screen flows. For example, by implementing the methods of that interface, you can programmatically control the name of the dialog that you want to start.

The code sample shows a dialog definition that uses the special start event. In this sample, if a portlet emits the special start event, the dialog with the name returned by the `getDialogDefinitionName()` method is started.

Note: The central idea of the IBM UX Screen Flow Manager is to control transitions that are based on the declarative model. Therefore, use this more programmatic approach with care and only if you cannot implement your goal by other means.

```
<transition type="start">
    <source>
        <transition-endpoint nameref="*">
            <event qname="{http://www.ibm.com/xmlns/prod/websphere/portal/v6.1.0/portal-pcm}StartD
        </transition-endpoint>
    </source>
    <target>
        ...
    </target>
</transition>
```

It might happen that a JSR-286 event is emitted by a portlet that participates in a dialog, but is not defined as part of a transition. Events that occur outside of a screen flow transition are not affected by the IBM UX Screen Flow Manager. They are delivered as if IBM UX Screen Flow Manager was not present. The previous code sample shows as an example: If `portlet4` emits an event `eX`, a matching transition is found and therefore run. If `portlet4` emits an event `eY`, a matching transition cannot be found, and therefore no transition is run. Nevertheless, the emission of the event `eY` affects the screen flow. For example, `portlet4` might be on `page4`, together with another portlet `portlet5`. Then, if `portlet5` can receive the event `eY`. The event `eY` is delivered from `portlet4` to `portlet5`, as if the IBM UX Screen Flow Manager were not present. The central idea of the IBM UX Screen

Flow Manager is to control transitions that are based on the declarative model. Therefore, use this more programmatic approach with care and only if you cannot implement your goal by other means.

As you can start dialogs by specifying dedicated transition endpoints, undetermined transition endpoints, dedicated events, or special start events, several combinations are possible. The code sample shows the behavior and interplay of these combinations.

```
 <dialog name="dialog1">
  <transition type="start">
      <source>
          <transition-endpoint nameref="*">
              <event qname="{http://www.ibm.com/xmlns/prod/websphere/portal/v6.1.0/portal-pcm}Star
          </transition-endpoint>
      </source>
      ...
  </transition>
 </dialog>

<dialog name="dialog2">
  <transition type="start">
      <source>
         <transition-endpoint nameref="*">
              <event qname="eX"/>
          </transition-endpoint>
      </source>
      ...
  </transition>
 </dialog>

 <dialog name="dialog3">
  <transition type="start">
      <source>
          <transition-endpoint nameref="portlet3">
              <event qname="{http://www.ibm.com/xmlns/prod/websphere/portal/v6.1.0/portal-pcm}Star
          </transition-endpoint>
      </source>
      ...
  </transition>
 </dialog>

 <dialog name="dialog4">
  <transition type="start">
      <source>
          <transition-endpoint nameref="portlet4">
              <event qname="eX"/>
          </transition-endpoint>
      </source>
      ...
  </transition>
 </dialog>
```

End transitions and special events:

Similar to the special start event, end transitions can emit a special end event.

This special end event is identified by the following `QName:`

```
{http://www.ibm.com/xmlns/prod/websphere/portal/v6.1.0/portal-pcm}EndDialog
```

In contrast to the special start event, the special end event is not expected to carry a payload.

```
<transition type="end">
     <source>
         <transition-endpoint nameref="portlet1">
             <event qname="e1"/>
         </transition-endpoint>
     </source>
     <target>
         <transition-endpoint nameref="portlet2">
             <event qname="{http://www.ibm.com/xmlns/prod/websphere/portal/v6.1.0/portal-pcm}EndDia
         </transition-endpoint>
     </target>
 </transition>
```

Other special transitions:

IBM UX Screen Flow Manager supports several more special transitions, or more precisely, transition endpoints.

After a dialog ends, the user needs to be redirected to a page. The following options are available:

1. If the target of an end transition points to the special transition endpoint `DEFAULT_RETURN`, the screen flow manager redirects the user. The user is redirected to the resource referenced by the unique name that is specified for the configuration option `com.ibm.wps.pcm.dialog.default.return.uniquename`. For more details, go to *Configuration options*. The event that is associated with this target is transmitted to the referenced resource.

   ```
   <transition type="end">
        <source>
            <transition-endpoint nameref="portlet1">
                <event qname="e1"/>
            </transition-endpoint>
        </source>
        <target>
            <transition-endpoint nameref="DEFAULT_RETURN">
                <event qname="e2"/>
            </transition-endpoint>
        </target>
    </transition>
   ```

2. If the target of an end transition points to the special transition endpoint `PAGE_ORIGIN`, the user is redirected to the page from which the dialog is triggered.

   The event that is associated with this target is transmitted to all portlets on that page. This transmission is also called broadcasting.

   ```
    <transition type="end">
        <source>
            <transition-endpoint nameref="portlet1">
                <event qname="e1"/>
            </transition-endpoint>
        </source>
        <target>
            <transition-endpoint nameref="PAGE_ORIGIN">
                <event qname="e2"/>
            </transition-endpoint>
        </target>
    </transition>
   ```

3. If the target of an end transition points to the special transition endpoint `PORTLET_ORIGIN`, the user is redirected to the page that contains the portlet from which the dialog is triggered.

   The event that is associated with this target is transmitted to that specific portlet.

```
<transition type="end">
    <source>
        <transition-endpoint nameref="portlet1">
            <event qname="e1"/>
        </transition-endpoint>
    </source>
    <target>
        <transition-endpoint nameref="PORTLET_ORIGIN">
            <event qname="e2"/>
        </transition-endpoint>
    </target>
</transition>
```

Another special transition is available for standard transitions that are not end transitions. In this context, dialog modelers sometimes want to define a dialog as follows: after a specific source portlet emitted a specific event, the user is returned to the resource or portlet. This resource or portlet redirects the user to the resource that emitted the event. Therefore, you can reference the special transition endpoint named `CALLER`.

For example in the code, the first transition is triggered after `portlet1` emits event `e1`. Event `e1` causes the user to be redirected to `portlet2`. Then, after `portlet2` emits event `e2-2`, the user is redirected back to `portlet1`, which previously called `portlet2`.

Code sample

```
<transition>
    <source>
        <transition-endpoint nameref="portlet1">
            <event qname="e1"/>
        </transition-endpoint>
    </source>
    <target>
        <transition-endpoint nameref="portlet2">
            <event qname="e2-1"/>
        </transition-endpoint>
    </target>
</transition>
<transition>
    <source>
        <transition-endpoint nameref="portlet2">
            <event qname="e2-2"/>
        </transition-endpoint>
    </source>
    <target>
        <transition-endpoint nameref="CALLER">
            <event qname="e3"/>
        </transition-endpoint>
    </target>
</transition>
```

Related concepts:

"Configuration options" on page 3525
To change the overall behavior of the IBM UX Screen Flow Manager, several configuration options are available. You specify the options as Resource Environment Provider (REP) properties.

Exclusive transitions or dialogs:

Exclusive dialogs are special forms of dialogs. They are also referred to as null or one-step dialogs. They consist of a single transition only. You can use them to go into a dialog, run only one transition, and use it to end the dialog immediately afterward.

Code sample

```
<transition type="exclusive">
 <source>
     <transition-endpoint nameref="portlet1">
         <event qname="eX"/>
     </transition-endpoint>
 </source>
 <target>
     <transition-endpoint nameref="portlet2">
         <event qname="eX"/>
     </transition-endpoint>
 </target>
</transition>
```

Dialog chaining and nesting:

Dialogs can start other dialogs. Dialogs that start other dialogs are referred to as calling dialogs and the dialogs that are being started are referred to as called dialogs. Two different options are available to complete this action such as dialog chaining and dialog nesting.

With dialog chaining option, one dialog can start another dialog in a way that the calling dialog ends after the called one starts. Therefore, after the called one starts, there is no way to go back to the calling one anymore. For example, in a travel site, you can configure the Flight booking or the calling dialog to end after the Car booking or the called dialog starts. After the Car booking dialog starts the user cannot go back to the Flight booking dialog.

With the dialog nesting option, one dialog can start another dialog but the calling dialog does not end after the called dialog starts. Therefore, you can go back and forth between these nested dialogs. For example, in a travel site, you can configure the Flight booking or the calling dialog to pause after the Car booking or the called dialog starts. After the Car booking dialog starts the user cannot go back and forth between the Flight booking and car booking dialogs.

In both options, the calling dialog references the dialog to be called as target. The called dialog starts with the event the calling dialog emits.

A called dialog can start only by triggering one of its start transitions and can end only by triggering one of its end transitions. In the following code sample, `dialog2` must have a start transition, which is triggered by the emission of an event `eX`. The `dialog3` must have a start transition, which is triggered by the emission of an event `eY`.

```
...
<transition type="chained">
    <source>
        <transition-endpoint nameref="portlet1">
            <event qname="e1a"/>
        </transition-endpoint>
    </source>
    <target>
        <transition-endpoint nameref="dialog2">
```

```
            <event qname="eX"/>
        </transition-endpoint>
    </target>
</transition>
...
<transition type="nested">
    <source>
        <transition-endpoint nameref="portlet1">
            <event qname="e1b"/>
        </transition-endpoint>
    </source>
    <target>
        <transition-endpoint nameref="dialog3">
            <event qname="eY"/>
        </transition-endpoint>
    </target>
</transition>
```

CF05 "Valid and invalid definitions"
Whether you start a dialog in a chained or a nested fashion, the dialogs can be controlled through an attribute `type`. The attribute `type` is assigned to the transition-endpoint element. Valid values include `chained` and `nested`. If you do not specify the attribute, the default attribute value `chained` is applied.

CF05 "Multiple start transitions" on page 3497
In case the called dialog has multiple start transitions that can be triggered by the emission of an event, the start transition that needs to be triggered can be controlled. To control which of the potentially matching start transitions must be triggered, an attribute `entry-point` is assigned to the transition-endpoint element that references the dialog to be called.

CF05 "Incoming transitions" on page 3498
When the transitions return from the called dialog to the calling dialog, the incoming transitions can control how the transitions must continue.

CF05 "Multiple outgoing transitions" on page 3499
When the calling dialog has multiple outgoing transitions from which the called dialog can be started, you must define how the transition continues. When you return from the called dialog, you must define that the continuation of transitions within the calling dialog differs. The continuation of transitions differs depending on where you exited the calling dialog.

Valid and invalid definitions:

Whether you start a dialog in a chained or a nested fashion, the dialogs can be controlled through an attribute `type`. The attribute `type` is assigned to the transition-endpoint element. Valid values include `chained` and `nested`. If you do not specify the attribute, the default attribute value `chained` is applied.

The following code sample shows the relevant fragments of valid definitions for the calling dialog `dialog1` and the called dialogs `dialog2` and `dialog3`.

Code sample

```
<dialog name="dialog1">
  ...
  <transition type="chained">
      <source>
          <transition-endpoint nameref="portlet1">
              <event qname="e1a"/>
          </transition-endpoint>
      </source>
      <target>
          <transition-endpoint nameref="dialog2">
```

```
            <event qname="eX"/>
        </transition-endpoint>
    </target>
  </transition>
  ...
  <transition type="nested">
      <source>
          <transition-endpoint nameref="portlet1">
              <event qname="e1b"/>
          </transition-endpoint>
      </source>
      <target>
          <transition-endpoint nameref="dialog3">
              <event qname="eY"/>
          </transition-endpoint>
      </target>
  </transition>
  ...
</dialog>
...
<dialog name="dialog2">
  ...
  <transition type="start">
      <source>
          <transition-endpoint nameref="portlet1">
              <event qname="eX"/>
          </transition-endpoint>
      </source>
      <target>
          <transition-endpoint nameref="portlet2">
              <event qname="eX"/>
          </transition-endpoint>
      </target>
  </transition>
  ...
</dialog>
...
<dialog name="dialog3">
  ...
  <transition type="start">
      <source>
          <transition-endpoint nameref="portlet1">
              <event qname="eY"/>
          </transition-endpoint>
      </source>
      <target>
          <transition-endpoint nameref="portlet2">
              <event qname="eY"/>
          </transition-endpoint>
      </target>
  </transition>
  ...
</dialog>
```

The following code sample shows invalid definitions as `dialog2` has no start transition that is triggered by the event `eX`.

Code sample

```
<dialog name="dialog1">
  ...
  <transition type="chained">
      <source>
          <transition-endpoint nameref="portlet1">
              <event qname="e1a"/>
          </transition-endpoint>
      </source>
```

```
      <target>
          <transition-endpoint nameref="dialog2">
              <event qname="eX"/>
          </transition-endpoint>
      </target>
  </transition>
  ...
  <transition type="nested">
      <source>
          <transition-endpoint nameref="portlet1">
              <event qname="e1b"/>
          </transition-endpoint>
      </source>
      <target>
          <transition-endpoint nameref="dialog3">
              <event qname="eY"/>
          </transition-endpoint>
      </target>
  </transition>
  ...
</dialog>
...
<dialog name="dialog2">
  ...
  <transition type="start">
      <source>
          <transition-endpoint nameref="portlet1">
              <event qname="eZ"/>
          </transition-endpoint>
      </source>
      <target>
          <transition-endpoint nameref="portlet2">
              <event qname="eZ"/>
          </transition-endpoint>
      </target>
  </transition>
  ...
</dialog>
...
<dialog name="dialog3">
  ...
  <transition type="start">
      <source>
          <transition-endpoint nameref="portlet1">
              <event qname="eY"/>
          </transition-endpoint>
      </source>
      <target>
          <transition-endpoint nameref="portlet2">
              <event qname="eY"/>
          </transition-endpoint>
      </target>
  </transition>
  ...
</dialog>
```

Note: If the dialogs `dialog2` and `dialog3` are started in a chained or nested fashion, the transition endpoint that is referenced by the start transition's source becomes irrelevant.

Multiple start transitions:

In case the called dialog has multiple start transitions that can be triggered by the emission of an event, the start transition that needs to be triggered can be

controlled. To control which of the potentially matching start transitions must be triggered, an attribute `entry-point` is assigned to the transition-endpoint element that references the dialog to be called.

The following code sample shows an example.

In the sample `dialog1` calls `dialog2`. `Dialog2` has two start transitions, which are both triggered by the emission of the event `eX`. The transition that calls `dialog2` from within `dialog1` uses the `entry-point` attribute. The attribute points to the respective source transition-endpoint `portlet1` of the transition to be triggered.

For example, in a travel site, Flight booking dialog transitions to the Car booking dialog. The Car booking dialog can start either from the Renters information portlet or the Calendar portlet. The transition uses to the `entry-point` to point to the portlet that needs to be start.

```
<dialog name="dialog1">
  ...
  <transition type="nested">
      <source>
          <transition-endpoint nameref="portlet1">
              <event qname="e1a"/>
          </transition-endpoint>
      </source>
      <target>
          <transition-endpoint nameref="dialog2" entry-point="portlet1">
              event qname="eX"/>
          </transition-endpoint>
      </target>
  </transition>
  ...
</dialog>
  ...
  <transition type="start">
      <source>
          <transition-endpoint nameref="portlet1">
              <event qname="eX"/>
          </transition-endpoint>
      </source>
      <target>
          <transition-endpoint nameref="portlet3">
             <event qname="eX"/>
          </transition-endpoint>
      </target>
  </transition>
  <transition type="start">
      <source>
          <transition-endpoint nameref="portlet2">
              <event qname="eX"/>
          </transition-endpoint>
      </source>
      <target>
          <transition-endpoint nameref="portlet3">
             <event qname="eX"/>
          </transition-endpoint>
      </target>
  </transition>
  ...
</dialog>
```

Incoming transitions:

When the transitions return from the called dialog to the calling dialog, the incoming transitions can control how the transitions must continue.

The following code sample shows an example. In the following sample, `dialog1` is the calling dialog that called `dialog2`. After the called dialog `dialog2` ends, the transition returns to the calling dialog `dialog1`. Because `portlet3` emits event `eZ`, the shown incoming transition defines that `dialog1` must continue with `portlet5`. The `portlet5` is initialized with the event `eZ`.

Code sample

```
<dialog name="dialog1">
  ...
  <transition>
      <source>
          <transition-endpoint nameref="dialog2">
              <event qname="eZ"/>
          </transition-endpoint>
      </source>
      <target>
          <transition-endpoint nameref="portlet5">
              <event qname="eZ"/>
          </transition-endpoint>
      </target>
  </transition>
  ...
</dialog>
...
<dialog name="dialog2">
  ...
  <transition type="end">
      <source>
          <transition-endpoint nameref="portlet3">
              <event qname="eZ"/>
          </transition-endpoint>
      </source>
      <target>
          <transition-endpoint nameref="portlet4">
              <event qname="eZ"/>
          </transition-endpoint>
      </target>
  </transition>
  ...
</dialog>
```

The event that is associated with the target of the called dialog's end transition needs to match with the event of the calling dialog's incoming transition. In the sample, `dialog1` needs to have an incoming transition, which is triggered by the emission of an event `eZ` by `dialog2`. The transition endpoint that is referenced by the called dialogs end transitions target becomes irrelevant.

Multiple outgoing transitions:

When the calling dialog has multiple outgoing transitions from which the called dialog can be started, you must define how the transition continues. When you return from the called dialog, you must define that the continuation of transitions within the calling dialog differs. The continuation of transitions differs depending on where you exited the calling dialog.

To control how to continue with such transitions, an attribute `resume-point` is assigned to the transition-endpoint element that references the dialog from which the transition is returned. The `resume-point` attribute needs to reference a transition endpoint of the calling dialog that was active when another dialog was started.

In the following code sample, the dialog1 has two transitions from which dialog2 can be started. One transition references the transition endpoint portlet1 as its source, the other references the transition endpoint portlet2. Furthermore, dialog2 has two distinct incoming transitions. One transition carries the attribute resume-point with value portlet1, the other one with value portlet2.

When you exit dialog1 through the first transition, which references the endpoint portlet1, dialog1 would continue with portlet3 after dialog2 ends. The reason is that the transition that references the resume point with value portlet1 is triggered since portlet1 was active when you exited dialog1.

When you exit dialog1 through the second transition, which references the endpoint portlet2, dialog1 would continue with portlet4 after dialog2 ends. The reason is that the transition that references the resume point with value portlet2 is triggered since portlet2 was active when you exited dialog1.

Code sample

```
<dialog name="dialog1">
   ...
   <transition>
       <source>
           <transition-endpoint nameref="portlet1">
               <event qname="eX"/>
           </transition-endpoint>
       </source>
       <target>
           <transition-endpoint nameref="dialog2">
               <event qname="eX"/>
           </transition-endpoint>
       </target>
   </transition>
   <transition>
       <source>
           <transition-endpoint nameref="portlet2">
               <event qname="eX"/>
           </transition-endpoint>
       </source>
       <target>
           <transition-endpoint nameref="dialog2">
               <event qname="eX"/>
           </transition-endpoint>
       </target>
   </transition>
   ...
   <transition>
       <source>
           <transition-endpoint nameref="dialog2" resumepoint="portlet1">>
               <event qname="eZ"/>
           </transition-endpoint>
       </source>
       <target>
           <transition-endpoint nameref="portlet3">
               <event qname="eZ"/>
           </transition-endpoint>
       </target>
   </transition>
   <transition>
       <source>
           <transition-endpoint nameref="dialog2" resumepoint>
               <event qname="eZ"/>
           </transition-endpoint>
       </source>
       <target>
```

```
          <transition-endpoint nameref="portlet4">
              <event qname="eZ"/>
          </transition-endpoint>
      </target>
  </transition>
  ...
</dialog>
...
```

Rules and restrictions:

When you model a transition certain rules and restrictions apply.

The following are descriptions of the rules:

1. The entire set of dialogs, together with all their transitions, needs to be deterministic. That is the transitions need to produce the same results all times. To achieve this condition, the IBM UX Screen Flow Manager must always clearly determine the transition that is to be triggered. The transition is determined based on the transition endpoint that emits an event and the name (`QName`) of the event. The screenflow is not valid if the definition is not deterministic. For example, the definition of two dialogs as shown in code sample is not valid, as it is not deterministic: The Screen Flow Manager might not decide whether to trigger the first or the second transition after `portlet1` emits event `e1`.

   Code sample

   ```
   <transition>
       <source>
           <transition-endpoint nameref="portlet1">
               <event qname="e1"/>
           </transition-endpoint>
      </source>
       <target>
          <transition-endpoint nameref="portlet2">
               <event qname="e2"/>
           </transition-endpoint>
       </target>
   </transition>
   <transition>
       <source>
           <transition-endpoint nameref="portlet1">
               <event qname="e1"/>
           </transition-endpoint>
      </source>
       <target>
           <transition-endpoint nameref="portlet3">
               <event qname="e3"/>
          </transition-endpoint>
       </target>
   </transition>
   ```

2. A source cannot reference more than one single transition endpoint.
3. A target can reference one of the following portal resources:
   - A single portlet that can receive one or multiple events.
   - Multiple target portlets. All these portlets must be found on the same page. Each single portlet can receive one or multiple events.
   - A single target page. Each single portlet on that page can receive the same event or multiple events that are sent to the page. This action is also called broadcasting.

- One target page and one or multiple target portlets. All these portlets must be found on the target page. Each single portlet on that page can receive the following types of events:
  - The same event or multiple events that are sent to the page.
  - One or multiple dedicated events that are sent to the portlet itself.

Execution priority:

You can and must explicitly model and enforce a deterministic behavior for the transitions. Even if the entire set of dialogs is deterministic, the emission of an event by a particular portlet can trigger two different transitions of two different dialogs. Therefore, you must explicitly model the intended behavior of the transitions.

For example, when a `portlet1` emits an event `e1`, two different transitions T1 and T2 can be triggered. The transition T1 as part of an active dialog D1 and the start transition T2 as part of another dialog D2 both react on the same event `e1`. After `portlet1` emits `e1`, the transitions needs to determine whether to continue with D1 or to suspend D1 and start D2.

You can explicitly model the intended behavior of the transitions by using the optional attribute `priority`. The attribute `priority` is assigned to the element dialog. Valid values that you can use are `preserve` and `suspend`. The default value is set to `suspend` unless you configure the settings differently. For more information about configuring, go to *Configuration Options*.

The value `preserve` defines that the transition that is part of an active dialog always wins. For example, after `portlet1` emits the event `e1`, the transition continues with active dialog D1 instead of switching to another dialog D2.

The value `suspend` defines that start-transitions always win. For example, after `portlet1` emits the event `e1`, the transition suspends the active dialog D1 and starts another dialog D2. If the attribute is not specified, the default value is applied.

The following code samples show the outlined behavior under the assumption that `dialog1` is already active. In the following code sample, after `portlet1` emits the event `e1`, both the transitions shown theoretically fit. Since, for `dailog1` the attribute `priority` is set to `suspend`, the intended behavior is to prefer triggering matching start transitions. Matching start transitions are triggered even if they are part of the active dialog. Hence in this case `dialog1` is suspended and the user must continue with `dialog2` and is redirected to `portlet3`.

Code sample

```
<dialog name="dialog1" priority="suspend">
    ...
    <transition>
        <source>
            <transition-endpoint nameref="portlet1">
                <event qname="e1"/>
            </transition-endpoint>
        </source>
        <target>
            <transition-endpoint nameref="portlet2">
                <event qname="e2"/>
            </transition-endpoint>
        </target>
    </transition>
    ...
```

```
</dialog>
...
<dialog name="dialog2">
    ...
    <transition type="start">
        <source>
            <transition-endpoint nameref="portlet1">
                <event qname="e1"/>
            </transition-endpoint>
        </source>
        <target>
            <transition-endpoint nameref="portlet3">
                <event qname="e3"/>
            </transition-endpoint>
        </target>
    </transition>
    ...
</dialog>
```

In the following code sample, after `portlet1` emits the event `e1`, both transitions shown theoretically fit. Since for `dialog1` the attribute `priority` is set to `preserve`, the intended behavior is to prefer triggering a matching transition that is a part of the active dialog. Thus, in this case the user continues with `dialog1` and is redirected to `portlet2`.

Code sample

```
<dialog name="dialog1" priority="preserve">
    ...
    <transition>
        <source>
            <transition-endpoint nameref="portlet1">
                <event qname="e1"/>
            </transition-endpoint>
        </source>
        <target>
            <transition-endpoint nameref="portlet2">
                <event qname="e2"/>
            </transition-endpoint>
        </target>
    </transition>
    ...
</dialog>
...
<dialog name="dialog2">
    ...
    <transition type="start">
        <source>
            <transition-endpoint nameref="portlet1">
                <event qname="e1"/>
            </transition-endpoint>
        </source>
        <target>
            <transition-endpoint nameref="portlet3">
                <event qname="e3"/>
            </transition-endpoint>
        </target>
    </transition>
    ...
</dialog>
```

Note: In the context of dialog chaining or nesting, the attribute `priority` is not inherited. For example, when a dialog D1 calls another dialog D2, the attribute `priority`, which might be specified for D1 is not propagated to D2.

Related concepts:

CF05 “Configuration options” on page 3525
To change the overall behavior of the IBM UX Screen Flow Manager, several configuration options are available. You specify the options as Resource Environment Provider (REP) properties.

Deploy dialog sets by using the XML configuration interface:

You can use the portal XML configuration interface (XMLAccess) to work with dialog sets.

You can import dialog sets by specifying the value `create` for the `action` attribute.

Code sample

```
<request type="update" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaL

 <portal action="create">
    <dialog-set>
        <dialog name="dialog1">
            ...
        </dialog>
        ...
     </dialog-set>
 </portal>
</request>
```

You can export dialog sets by specifying the value `export` for the `action` attribute. For example, you can use this option for staging or migration purposes. The following code sample shows how you can export a single dialog definition `dialog1`.

Code sample

```
<request type="update" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaL

 <portal action="export">
    <dialog-set>
        <dialog name="dialog1">
    </dialog-set>
 </portal>
</request>
```

When you export dialog sets or definitions, you can use wildcards. The following code sample shows how to export all available dialog definitions.

Code sample

```
<request type="update" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaL

 <portal action="export">
     <dialog-set>
         <dialog name="*">
     </dialog-set>
 </portal>
</request>
```

The following code sample shows how to export all dialog definitions with the name that starts with the string `toBeExported`.

Code sample

```
<request type="update" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLo

    <portal action="export">
```

```
       <dialog-set>
           <dialog name="toBeExported*">
       </dialog-set>
 </portal>
</request>
```

The following code sample shows how to export all dialog definitions with the name that ends with the string `toBeExported`.

Code sample

```
<request type="update" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchem

 <portal action="export">
     <dialog-set>
         <dialog name="*toBeExported">
     </dialog-set>
 </portal>
</request>
```

The following code sample shows how to export all dialog definitions with the name that contains the string `toBeExported`.

Code sample

```
<request type="update" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchem

 <portal action="export">
     <dialog-set>
         <dialog name="*toBeExported*">
     </dialog-set>
 </portal>
</request>
```

You can delete dialog sets by specifying the value `delete` for the `action` attribute. The following code sample shows how to delete a single dialog definition with the name `dialog1`. You can use the same wildcards as for exporting.

Code sample

```
<request type="update" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchem

 <portal action="delete">
     <dialog-set>
         <dialog name="dialog1">
     </dialog-set>
 </portal>
</request>
```

Note: You cannot use the XML configuration interface to merge an updated dialog definition with an existing one that you imported earlier. The XMLAccess request type `update` overwrites the specified existing portal resource with the new one. Therefore, if you want to merge a previous dialog definition with a newer one, you need to manually merge the two XML scripts. To merge the two XML scripts, make sure that the new dialog definition contains both the new sections and the old sections that you want to preserve.

Deploying user interface artifacts

Before you deploy the Screen Flow Manager dialog definitions, you must deploy all portal resources that are part of the dialog definitions, such as pages, and portlets. You can then deploy the dialog definitions by using the portal XML configuration interface (XMLAccess).

You can import, export, update, or delete complete dialog sets by using the following command.

```
xmlaccess.{sh|bat} -in your_dialog_sets_file_name -user user_ID -password password -url http://host_
```

Advanced concepts

Learn about the advanced concepts of the IBM UX Screen Flow Manager.

"Retrieve and store event payload"
In a normal portal environment, the portlets in a single dialog can exchange data through a set of well-defined events. However, you might want to include third-party portlets or older portlets that are not aligned with the normal portal environment. You can include such portlets with the IBM UX Screen Flow Manager.

"Access control" on page 3515
You can use the Portal Access Control (PAC) to control what users can do when they are working with dialogs.

CF05 "Scoping session data and render parameters" on page 3516
The appearance of portlets heavily relies on the portlet session data and the render parameters. To better support the semi-parallel processing of dialogs, the portlet session data and the render parameters are stored in a scoped fashion.

"Dynamic pages and portlets" on page 3516
The IBM UX Screen Flow Manager not only supports redirecting users between static portal resources, but also between dynamic resources. The WebSphere Portal feature Dynamic UI Management is used.

Retrieve and store event payload

In a normal portal environment, the portlets in a single dialog can exchange data through a set of well-defined events. However, you might want to include third-party portlets or older portlets that are not aligned with the normal portal environment. You can include such portlets with the IBM UX Screen Flow Manager.

For example, in a travel site, a passenger information portlet `portlet1` can emit the ID of a passenger. It emits this ID by using a JSR-286 event with the `QName passengerID`. A second portlet, car renters information portlet `portlet2` can receive the ID of the passenger, but it expects this ID to be sent through a JSR-286 event with the `QName userID`. Without extra translation, the two portlets cannot communicate with each other because the events that they exchange use different `QNames`.

The Screen Flow Manager supports two mechanisms that enable such incompatible portlets to exchange data with each other:

- Event mappers. For more information, go to *Event Mappers*.
- The explicit specification of dialog context (DCX) keys when events are associated with referenced transition endpoints.

  CF05 "Dialog context keys" on page 3507
  The dialog context (DCX) acts like the transient memory of a dialog. It maintains contextual information that is passed from one portlet to subsequent portlets and provides the information to all subsequent portlets. Data that is stored in the dialog context (DCX) is stored only for the lifetime of a user session. If a user logs out of the portal or the user session times out, the current dialog instance, all suspended dialog instances, and all related data are lost.

"Event Mappers" on page 3508
In more complex scenarios, adapting only the event names or QNames might not be enough. Sometimes you might need to transform the payload.

CF05 "Dialog chaining and nesting DCX keys and event mappers" on page 3511
In context of dialog chaining or nesting, DCX keys and mappers never influence another dialogs DCX segment. When a transition part of a calling dialog uses a DCX key or a mapper it influences only the data that is stored in the calling dialog's DCX segment and not the called dialog's segment. Similarly, when a transition part of a called dialog uses a DCX key or a mapper it influences only the data that is stored in the called dialog's DCX segment and not the calling dialog's segment.

Related concepts:

"Event Mappers" on page 3508
In more complex scenarios, adapting only the event names or QNames might not be enough. Sometimes you might need to transform the payload.

Dialog context keys:

The dialog context (DCX) acts like the transient memory of a dialog. It maintains contextual information that is passed from one portlet to subsequent portlets and provides the information to all subsequent portlets. Data that is stored in the dialog context (DCX) is stored only for the lifetime of a user session. If a user logs out of the portal or the user session times out, the current dialog instance, all suspended dialog instances, and all related data are lost.

- The DCX is divided into segments where each segment holds data or contextual information of a single dialog instance.
- Normally, when you specify events, the QName defines how and under which key the corresponding payload is stored in the segment of the specific dialog instance.

  For example, the DCX contains the following information after you run the transition:

  – An entry with key = passengerID.
  – A value that matches the payload that was sent with the corresponding event.

  The car renters information portlet portlet2 is then initialized with the data stored in the DCX under the key userID. If no previous transition stored anything under the key userID, portlet2 does not receive any data.

  Code sample

```
<transition>
    <source>
        <transition-endpoint nameref="portlet1">
            <event qname="passengerID"/>
        </transition-endpoint>
    </source>
    <target>
        <transition-endpoint nameref="portlet2">
            <event qname="userID"/>
        </transition-endpoint>
    </target>
</transition>
```

DCX keys make it possible to specify under which key a portlet stores data in the DCX, or under which key a portlet reads data from the DCX.

To enable portlet1 to communicate with portlet2, you have the following two options:

- You can make portlet1 store its payload under the key userID, even though it emits an event with the QName passengerID. The following code sample shows how you can make portlet1 store data under userID.

  Code sample

```
<transition>
    <source>
        <transition-endpoint nameref="portlet1">
            <event qname="passengerID" dcx-key="userID"/>
        </transition-endpoint>
    </source>
    <target>
        <transition-endpoint nameref="portlet2">
            <event qname="userID"/>
        </transition-endpoint>
    </target>
</transition>
```

- You can make portlet2 read the key passengerID, even though it expects an event with the QName userID. The following code sample shows how you can make portlet2 read data under passengerID.

  Code sample

```
<transition>
    <source>
        <transition-endpoint nameref="portlet1">
            <event qname="passengerID"/>
        </transition-endpoint>
    </source>
    <target>
        <transition-endpoint nameref="portlet2">
            <event qname="userID" dcx-key="passengerID"/>
        </transition-endpoint>
    </target>
</transition>
```

Event Mappers:

In more complex scenarios, adapting only the event names or QNames might not be enough. Sometimes you might need to transform the payload.

For example, in a travel site, a passenger information portlet portlet1 emits a passenger ID under the key passengerID. The payload is of type Integer. A second portlet, a car renters information portlet portlet2 expects a passenger ID under the key userID, and sent as payload of type String.

As with DCX keys to enable the passenger information portlet portlet1 to communicate with the car renters information portlet portlet2, you have the following two options:

- You can make the passenger information portlet portlet1 store its payload under the key userID and as of type String, even though it emits an event with the QName employeeID and as of type Integer.
- You can make the car renters information portlet portlet2 read the key employeeID and as of type Integer, even though it expects an event with the QName userID and as of type String.

Mappers have full access to the DCX segment of the currently processed dialog instance and also to the payload that is being processed. Therefore, mappers are powerful tools, as they can run various transformations.

Implement MapperFactory to make the mappers available. You can register the MapperFactory with an Eclipse Extension Point. The advantage of this approach is that you can register new mappers without having to restart the server. This method is also called hot deployment. To use a MapperFactory, you need to implement the interface MapperFactory and return an object instance of type ContextToPayloadMapper or PayloadToContextMapper. This implementation depends on the concrete event mapper that is requested.

The following code sample shows an example of how a MapperFactory can look.

Code sample

```
public class MapperFactory implements com.ibm.portal.pcm.events.MapperFactory {

 public ContextToPayloadMapper getContextToPayloadMapper(String name) {
     ContextToPayloadMapper result = null;
     if (name.equals("myPackage.myMapper")) {
         result = new MyMapper();
     }

     return result;
 }

 public PayloadToContextMapper getPayloadToContextMapper(String name) {
     // ...
 }
}
```

The following code sample shows how you can register a MapperFactory with a plugin.xml file.

Code sample

```
<plugin
    id="com.ibm.wps.portlet.pcm.demo"
    name="Portlet Control Manager Demo"
    version="1.0.0"
    provider-name="IBM">

    <!-- Mapper Factory -->
    <extension point="com.ibm.portal.pcm.MapperFactory" id="PcmMapperFactory">
        <factory class="com.ibm.wps.pcm.demo.mapper.MapperFactory"/>
    </extension>
</plugin>
```

"Event mapper types"
IBM UX Screen Flow Manager supports the following two types of event mappers: PayloadToContextMappers and ContextToPayloadMappers.

"Packaging of event mappers and JAXB serialization" on page 3511
It is good practice to package event mappers in a shared library rather than together with the business portlets.

Event mapper types:

IBM UX Screen Flow Manager supports the following two types of event mappers: PayloadToContextMappers and ContextToPayloadMappers.

Consider the following example to understand how the two types of event mappers function.

Example: In a travel site, a passenger information portlet portlet1 emits a passenger ID under the key passengerID. The payload is of type Integer. A second

portlet, a car renters information portlet portlet2 expects a passenger ID under the key userID, and sent as payload of type String.

- PayloadToContextMappers mappers transform an event that is emitted by a transition source and influence the way that it is stored in the DCX segment. These mappers are started before data is stored in the DCX and therefore can control the key under which data is stored and the data type.

  The following code sample shows how you can use a PayloadToContextMapper to enable the passenger information portlet portlet1 and the car renters information portlet portlet2 in the example to communicate with each other.

  Code sample

  ```
  public void payloadToContext(final QName dcxKey, final Object payload, final DCXData dcxData) {
    // ...

    QName mappedDcxKey = new QName("userID");
    String transformedPayload = ((Integer)payload).toString();

    dcxData.put(mappedDcxKey, transformedPayload);
  }
  ```

  To use this mapper, you need to implement the interface PayloadToContextMapper. The input parameters carry information about the following two items:

  – The DCX key that is used to store the event as defined in association with the source, if it is not changed by the mapper.

  – The payload that is stored under this key, if it is not changed by the mapper.

  The input parameters also provide a reference to the DCX segment that belongs to a specific dialog instance. In this example, the mapper changes the DCX key where the payload is to be stored and the payload itself by transforming it from type Integer to type String.

  The following code sample shows how to register PayloadToContextMappers.

  Code sample

  ```
  <transition>
      <source>
          <transition-endpoint nameref="portlet1">
              <event qname="passengerID" mapper-class="myPackage.myMapper"/>
          </transition-endpoint>
      </source>
      <target>
          <transition-endpoint nameref="portlet2">
              <event qname="userID"/>
          </transition-endpoint>
      </target>
  </transition>
  ```

- ContextToPayloadMappers can transform an event before it is transmitted to a transition target. These mappers are started after data is read from the DCX and before it is transmitted to the target. This action means that these mappers can influence the payload before it is sent.

  The following code sample shows how you can use a ContextToPayloadMapper to enable the passenger information portlet portlet1 and the car renters information portlet portlet2 in the example to communicate with each other.

  Code sample

  ```
  public Serializable contextToPayload(final DCXData dcxData, final QName dcxKey, final Class<?> tar
    // ...
    Object payload = dcxData.get(dcxKey, targetClass);
  ```

```
    String transformedPayload = ((Integer)payload).toString();

    return (Serializable) transformedPayload;
}
```

To use this mapper, you need to implement the interface `ContextToPayloadMapper`. The input parameters carry information about the following two items:

- The DCX key or `QName` that is used to send the event as defined in association with the source.
- The payload is sent under this key - if it is not changed by the mapper.

The input parameters also provide a reference to the DCX segment that belongs to a specific dialog instance. In this example, the mapper changes the type of the payload that is to be sent by transforming it from type `Integer` to type `String`.

The following code sample shows how to register `ContextToPayloadMappers`.

Code sample

```
<transition>
    <source>
        <transition-endpoint nameref="portlet1">
            <event qname="passengerID"/>
        </transition-endpoint>
    </source>
    <target>
        <transition-endpoint nameref="portlet2">
            <event qname="userID" dcx-key="employeeID" mapper-class="myPackage.myMapper"/>
        </transition-endpoint>
    </target>
</transition>
```

Packaging of event mappers and JAXB serialization:

It is good practice to package event mappers in a shared library rather than together with the business portlets.

If you decide to package event mappers with more than one business portlets, the Screen Flow Manager runs JAXB-based marshalling and unmarshaling. This action prevents `ClassCastExceptions` caused by the use of one or more isolated classloaders. You can influence the behavior of the JAXB based marshalling by using specific configuration options. For more information, go to *Configuration Options*.

Related concepts:

"Configuration options" on page 3525
To change the overall behavior of the IBM UX Screen Flow Manager, several configuration options are available. You specify the options as Resource Environment Provider (REP) properties.

Dialog chaining and nesting DCX keys and event mappers:

In context of dialog chaining or nesting, DCX keys and mappers never influence another dialogs DCX segment. When a transition part of a calling dialog uses a DCX key or a mapper it influences only the data that is stored in the calling dialog's DCX segment and not the called dialog's segment. Similarly, when a transition part of a called dialog uses a DCX key or a mapper it influences only the data that is stored in the called dialog's DCX segment and not the calling dialog's segment.

Use of DCX keys and mappers in dialog chaining or dialog nesting is required to properly exchange data between the calling and called dialogs. In many cases, called dialogs are developed independently from calling dialogs, not intended to be part of another dialog.

For example, consider a travel booking screen flow. It consists of steps that allow to book a flight, a hotel, and a car after a particular date of travel, and destination is specified. Before the agent can start booking, the customer's personal data needs to be collected. Assume that another screen flow, a billing screen flow, is already modeled which allows the agent to look up this data through a customer ID. Thus the entire flow requires the booking to be started, the personal data to be collected and updated and finally the actual bookings to be done.

DCX keys

Assume the travel booking screen flow that carries the customer information under the key `travellerID` to be the blue screen flow. Assume the billing screen flow that carries the customer information under the key `customerID` to be the red screen flow.

1. During step 1, the `travellerID` is specified along with the date of travel, and destination. The red screen flow is started in a nested fashion.
2. During step 2, the red screen flow is supposed to display the customer information such as name, and address.
3. During step 3, the red screen flow must allow for updating the customer information. The red screen flow needs to be fed with the `travellerID` but expects it to be sent with an event with `QName customerID`.
4. The entire set of customer information such as name, and address, is packaged in a special customer object. This customer object is emitted by the red screen flow's end transition under the key `customerBean`.
5. The blue screen flow requires this data but expects it to be sent by an event with `QName travellerBean`. The `travellerID` needs to be converted to `customerID` and `travellerBean` to `customerBean`.

The following code sample shows an option to convert the event `Qnames`. The outgoing transition responsible for calling the red dialog assigned the attribute `dcx-key` with value `customerID` to the event element associated with this transition's source transition endpoint. Thus, even though the transition endpoint step_1 emits an event with `QName travellerID`, the information is stored in the blue dialog's DCX under the key `customerID`. Thus it can be sent to the red dialog through an event with `QName customerID` as the corresponding payload can be found in the blue dialog's DCX under exactly that key. The blue dialog's incoming transition receives the customer information through an event with `QName customerBean`, but stores the corresponding payload under the key `travellerBean`. Thus the customer information can be propagated with events with `QName travellerBean` as the corresponding payload can be found in the blue dialog's DCX under exactly that name.

Code sample

```
<dialog name="dialog blue">
  ...
  <transition type="nested">
      <source>
          <transition-endpoint nameref="step_1">
              <event qname="travellerID" dcx-key="customerID"/>
          </transition-endpoint>
```

```
      </source>
      <target>
          <transition-endpoint nameref="dialog red">
              <event qname="customerID"/>
          </transition-endpoint>
      </target>
  </transition>
  ...
  <transition>
      <source>
          <transition-endpoint nameref="dialog red">
              <event qname="customerBean" dcx-key="travellerBean"/>
          </transition-endpoint>
      </source>
      <target>
          <transition-endpoint nameref="step 4">
              <event qname="travellerBean"/>
          </transition-endpoint>
      </target>
  </transition>
  ...
</dialog>
...
<dialog name="dialog red">
 ...
  <transition type="start">
      <source>
          <transition-endpoint nameref="...">
              <event qname="customerID"/>
          </transition-endpoint>
      </source>
      <target>
          <transition-endpoint nameref="step_2">
              <event qname="customerID"/>
          </transition-endpoint>
      </target>
  </transition>
  ...
  <transition type="end">
      <source>
          <transition-endpoint nameref="step 3">
              <event qname="customerBean"/>
          </transition-endpoint>
      </source>
      <target>
          <transition-endpoint nameref="PAGE ORIGIN">
              <event qname="customerBean"/>
          </transition-endpoint>
      </target>
  </transition>
  ...
</dialog>
...
</dialog>
```

Event mappers

Assume the screen flow that carries the customer information under the key `travellerID` to be the blue screen flow. Assume the screen flow that carries the customer information under the key `customerID` to be the red screen flow. Assume the `travellerID` is of type `string` while the `customerID` is of type `Integer`. Also, assume that the beans `travellerBean` and `customerBean` differ not only by name but are also of different object type.

The following code sample shows an option to convert the keys and the bean. The mappers ensure that the events are properly renamed and the data types are properly transformed.

Code sample

```
<dialog name="dialog blue">
  ...
  <transition type="nested">
      <source>
          <transition-endpoint nameref="step 1">
              <event qname="travellerID" mapper-class="B2RMapper"/>
          </transition-endpoint>
      </source>
      <target>
          <transition-endpoint nameref="dialog red">
              <event qname="customerID"/>
          </transition-endpoint>
      </target>
  </transition>
  ...
  <transition>
      <source>
          <transition-endpoint nameref="dialog red">
              <event qname="customerBean" mapper-class="R2BMapper"/>
          </transition-endpoint>
      </source>
      <target>
          <transition-endpoint nameref="step 4">
              <event qname="travellerBean"/>
          </transition-endpoint>
      </target>
  </transition>
  ...
</dialog>
...
<dialog name="dialog red">
 ...
  <transition type="start">
      <source>
          <transition-endpoint nameref="...">
              <event qname="customerID"/>
          </transition-endpoint>
      </source>
      <target>
          <transition-endpoint nameref="step_2">
              <event qname="customerID"/>
          </transition-endpoint>
      </target>
  </transition>
  ...
  <transition type="end">
      <source>
          <transition-endpoint nameref="step 3">
              <event qname="customerBean"/>
          </transition-endpoint>
      </source>
      <target>
          <transition-endpoint nameref="PAGE ORIGIN">
              <event qname="customerBean"/>
          </transition-endpoint>
      </target>
  </transition>
  ...
</dialog>
...
</dialog>
```

Note: In the context of dialog chaining and dialog nesting, the definition of a mapper as part of an outgoing transition's target is invalid. The definition of a mapper as part of the called dialog's start transition's source is also invalid.

Access control

You can use the Portal Access Control (PAC) to control what users can do when they are working with dialogs.

You can assign users the following roles on the virtual resource PCM_DIALOGS:

- Administrator - An administrator has the rights and all tasks that are related to dialogs are done by the administrator.
- Security administrator - A security administrator can grant access on dialogs to other users.
- Manager - A manager can delete dialog definitions.
- User - A user can view a dialog definition and all its transition endpoints and transitions.
- Editor - An editor can create or edit a dialog definition. For example, an editor can add or remove transition endpoints or transitions.

These roles settings are inherited.

To assign users or user groups to these roles with the XML configuration interface (XMLAccess), run a script similar to the one shown in the following code sample.

Code sample

```
01 <virtual-resource action="update" domain="rel" name="PCM_DIALOGS">
02  <access-control externalized="false" owner="undefined" private="false">
03          <role actionset="User" update="set">
04              <mapping subjectid="all authenticated portal users" subjecttype="user_group" updat
05          </role>
06  </access-control>
07 </virtual-resource>
```

Processing a specific dialog depends on whether the user has sufficient rights on all pages and portlets that are part of the dialog. You specify access control settings on a dialog definition level. The following code sample shows an example.

Code sample

```
01 <dialog name="dialog1">
02  <access-control externalized="false" owner="undefined" private="false">
03      <role actionset="User" update="set">
04          <mapping subjectid="uid=wpsadmin,o=defaultWIMFileBasedRealm" subjecttype="USER" update
05      </role>
06      <role actionset="Editor" update="set">
07          <mapping subjectid="uid=wpsadmin,o=defaultWIMFileBasedRealm" subjecttype="USER" update
08      </role>
09  </access-control>
10  <transition-endpoint name="...">
11      ...
12  </transition-endpoint>
13  <transition>
14      ...
15  </transition>
16 </dialog>
```

Note: Assigning access on a specific dialog definition does not automatically assign access to all artifacts that are part of this dialog definition, such as pages, and portlets.

Scoping session data and render parameters

The appearance of portlets heavily relies on the portlet session data and the render parameters. To better support the semi-parallel processing of dialogs, the portlet session data and the render parameters are stored in a scoped fashion.

Storing the portlet session data and the render parameters in a scoped fashion avoids the active dialog from influencing the other dialogs. It also ensures that when the dialogs are resumed they appear exactly the way they appeared when they were suspended.

The data from an active dialog is stored in a different partition than the data that is stored when another dialog is active. Thus the data that is stored by one dialog cannot influence another dialog.

For example, in a travel site, when you start a Flight booking dialog D1, a partition p1 is created. When the Flight booking dialog D1 remains active, all portlet session data and render parameters of the dialog D1 are stored in partition p1. The partition p1 is not accessible or visible to any other partition. When the Flight booking dialog D1 is suspended, the partition P1 and all the data that is stored is also suspended.

When another dialog, a Car booking dialog D2 starts, another partition P2 is created. And when the Car booking dialog D2 remains active all data from dialog D2 is stored in partition P2. When the suspended Flight booking dialog D1 resumes, P1 resumes as well. This action ensures that when the Flight booking dialog D1 is active data is read from and written to P1 and when the Car booking dialog D2 remains active all data is read and written to P2. The partitions are deleted as soon as the dialog they belong to is canceled or ends.

Note: In the context of dialog nesting, a separate partition is created for any nested dialog. In other words, dialogs that call other dialogs read data from and write data to different partitions. This action ensures that the same portlets can be used as part of dialogs that call each other in a nested fashion without causing undesired interferences.

Dynamic pages and portlets

The IBM UX Screen Flow Manager not only supports redirecting users between static portal resources, but also between dynamic resources. The WebSphere Portal feature Dynamic UI Management is used.

You can use the Dynamic UI Management feature to create pages and portlets at run time. You can also use it to modify a user's content model and the navigation model, triggered by a user interaction.

In most cases, a dynamic page is a transient copy of a template page, often referred to as base page. This transient copy behaves like a snapshot of the base page from the time when the copy was created. It contains all portlets of the base page and all its properties. A dynamic portlet is a transient copy of a portlet definition. You can add dynamic portlets only to dynamic pages.

The benefit of using dynamic pages instead of static pages is that you can create multiple copies or instances of the base page. A user can then manually switch or be redirected between these instances.

Dynamic pages are always added to an extension node, a page to which a transformation is assigned.

During the processing of a dialog, single subdialogs can either be static or dynamic. If the subdialog is supposed to be dynamic, you can distinguish between the following two cases:

- If the transition endpoint references a page, the redirect requires a dynamic page. The dynamic page needs to be a transient copy, of the referenced base page that is to be started and to be added under the extension node.
- If the transition endpoint references a portlet, the redirect requires a dynamic portlet. The dynamic portlet needs to be a transient copy, of the referenced portlet definition that is to be started and to be added to an existing dynamic page. This case requires that a dynamic page is always created previously. The Screen Flow Manager ensures this requirement by starting an empty dynamic page. The dynamic portlet can then be added to that page.

You can use the dialog definition to control whether to start a resource dynamically. You can also specify in which extension node you want to add the dynamic copy. The following code sample shows an example.

Code sample

```
<dialog name="dialog1">
 <transition-endpoint name="page2">
     <resource uniquename="uniquename.page2"/>
     <invocation type="dynamic" extension-node="extensionNode1"/>
 </transition-endpoint>
 <transition-endpoint name="portlet1">
     <resource uniquename="uniquename.portlet1"/>
     <invocation type="static" extension-node="extensionNode1"/>
</transition-endpoint>
 <transition-endpoint name="portlet2">
     <resource uniquename="uniquename.portlet2"/>
     <invocation type="dynamic" extension-node="extensionNode1"/>
 </transition-endpoint>
 <transition>
     <source>
         <transition-endpoint nameref="portlet1">
             <event qname="e1-1"/>
         </transition-endpoint>
     </source>
     <target>
         <transition-endpoint nameref="page2">
             <event qname="e2"/>
         </transition-endpoint>
     </target>
 </transition>
 <transition>
     <source>
         <transition-endpoint nameref="portlet1">
             <event qname="e1-2"/>
         </transition-endpoint>
     </source>
     <target>
         <transition-endpoint nameref="portlet2">
             <event qname="e2"/>
         </transition-endpoint>
     </target>
 </transition>
</dialog>
```

In this sample, you can see two transitions: In both cases, `portlet1` exists on a static page.

- If the portlet emits the event `e1-1`, the user is redirected to `page2`. The `page2` is in turn expected to start dynamically in the extension node or page with the

unique name extensionNode1. The dynamic page is a transient copy of the base page with the unique name uniquename.page2.

- If portlet1 emits the event e1-2, the user is redirected to portlet2, which is also expected to start dynamically. A dynamic page is required to which this dynamic portlet can be added. Therefore, the screen flow manager creates an empty dynamic page in the extension node or page with the unique name extensionNode1 to which the portlet is then added.

The portal removes dynamic resources when they are no longer needed. For example, after a transition redirects a user from a dynamic page dpage1 to a dynamic page dpage2, dpage1 is removed and dpage2 is created.

User interface components

In a default WebSphere Portal installation, the Dialog Stack and Dialog State Display portlets are deployed. The following topics describe how these portlets function.

CF05 "Dialog Stack"
The Dialog Stack provides an overview of active and suspended dialogs. It allows for suspending, resuming, and canceling the dialogs.

CF05 "Dialog State Display" on page 3520
The Dialog State Display (DSD) displays a dialogs current state. You can move forward and backward or to jump to a dedicated step that was processed before. It is a generic navigation component that can be easily added to any page that participates in a dialog.

Dialog Stack

The Dialog Stack provides an overview of active and suspended dialogs. It allows for suspending, resuming, and canceling the dialogs.

The Dialog Stack displays the name of the current active dialog and the date and time when the dialog was started.

Consider for example, you are in a travel site and the screen flow is active. You are currently working on the Flight booking dialog. When you look at the Dialog Stack portlet, it would list the Flight booking dialog as active and the other dialogs in the travel site as suspended. For example, the Car booking and Hotel booking dialogs are listed as suspended.

The Dialog Stack also provides the option to return, suspend, or cancel the dialog. The functions of these options are as follows:

Return

When the user clicks Return, they are redirected to the currently active step of the currently active dialog from the Dialog Stack display screen.

For example, the user is redirected from the Dialog Stack portlet to the active Flight booking dialog.

Suspend

When the user clicks Suspend, it causes the active dialog to be suspended. After the dialog is suspended, it is no longer displayed as active and appears under the list of suspended dialogs.

Note: If the active dialog that is suspended is a dialog that is called by another dialog in a nested fashion, then the entire chain is suspended.

For example, the Flight booking dialog is suspended and listed under the suspended dialogs. And if the Flight booking dialog is called by the Car booking dialog and is nested both the dialogs are suspended.

Cancel

When the user clicks `Cancel`, the active dialog is canceled. The dialog is no longer available as active or suspended dialog and cannot be resumed.

Note: If the active dialog that is canceled is a dialog that is called by another dialog in a nested fashion, the entire chain is canceled.

For example, the Flight booking dialog is canceled and cannot be resumed. And if the Flight booking dialog is called by the Car booking dialog and is nested both the dialogs are canceled.

Cancel Nested

This option is only available if enabled.

When the user clicks `Cancel Nested`, the active dialog is canceled. But in contrast to the `Cancel` option, `Cancel Nested` option does not cause the entire chain of dependent dialogs to be canceled.

For example, if the Flight booking dialog called the Car booking dialog and the Car booking dialog is active. When the user clicks `Cancel` both the Flight booking and Car booking dialogs are canceled. When the user clicks `Cancel Nested` only the Car booking dialog is canceled; then, a redirect to the Flight booking dialog would occur. As the Flight booking dialog is the step that was active, when it was exited to call the Car booking dialog.

After the section that displays the current active dialog, the Dialog Stack displays a list of previously suspended dialogs by their name and start date. Each of these suspended dialogs can be canceled or resumed. When the user clicks `Resume` for the suspended dialog becomes active again.

Note: If another dialog is already active when the suspended dialog is resumed, the already active dialog is implicitly suspended.

CF05 "Configuration options for Dialog Stack"
Various configuration options are available for dialog stack. Two menus on the Dialog Stack provide the user with options to influence how dialogs are displayed.

Configuration options for Dialog Stack:

Various configuration options are available for dialog stack. Two menus on the Dialog Stack provide the user with options to influence how dialogs are displayed.

Consider the following example to better understand the configuration options available to you.

Assume a travel site with the following dialogs,

- Flight booking (dialog D1) consists of the following steps Passenger information(p1), Calendar(p2), and Destination(p3).
- Car booking(dialog D2) consists of the following steps Renters information (p4), Calendar (p5), and Vehicle type (p6).

The following are the available configuration options:

Display Initiating Screen Flow

When the user deals with only a single dialog and there is no nesting, the name of the dialog is displayed. For example, when the user works with the Flight booking dialog and the other dialogs are not nested, the name of the Flight booking dialog is displayed.

To understand how things work in a scenario where dialogs call other dialogs, consider the following example:

Assume that The Flight booking dialog D1 usually calls the Car booking dialog D2 from within the Calendar (p2) step and when the user returns from the Car booking dialog D2 to the Flight booking dialog D1 they usually end up in the Destination (p3) step. With the Display Initiating Screen Flow option, the Dialog Stack displays the dialogs as follows:

When in steps Passenger information(p1), Calendar(p2), Renters information (p4), Calend

Display Current Screen Flow

When the user deals with only a single dialog and there is no nesting, the name of the dialog is displayed. For example, when the user works with the Flight booking dialog and the other dialogs are not nested, the name of the Flight booking dialog is displayed.

In the example scenario where dialogs call other dialogs, with the Display Current Screen Flow option the dialog stack portlet displays the dialogs as follows:

When in steps Passenger information(p1), Calendar(p2), Renters information (p4), Calend

Display All Steps in Breadcrumb

When the user deals with only a single dialog and there is no nesting, the name of the dialog is displayed. For example, when the user works with the Flight booking dialog and the other dialogs are not nested, the name of the Flight booking dialog is displayed.

In the example scenario where dialogs call other dialogs, with the Display All Steps in Breadcrumb option the Dialog Stack displays the dialogs as follows:

When in steps Passenger information(p1), and Calendar(p2), the Flight booking dialog
When in step Renters information (p4), Flight booking D1 → Car booking D2 is display
When in steps Calendar (p5),and Vehicle type (p6), Flight booking D1 → Car booking D
When in step Destination(p3), the Flight booking dialog D1 is displayed.

Dialog State Display

The Dialog State Display (DSD) displays a dialogs current state. You can move forward and backward or to jump to a dedicated step that was processed before. It is a generic navigation component that can be easily added to any page that participates in a dialog.

Dialog State Display functions

The Dialog State Display Portlet displays the particular dialogs breadcrumb trail. For example, in a travel booking screen flow the dialog routes a user from a step to the other steps. For example, from a step that is called Flight Booking to steps called Hotel Booking, Car Booking, Insurance Booking, and Travel Summary. The currently active step is displayed in bold.

The Backward and Forward buttons trigger the following actions:

Backward

Clicking `Backward` redirects the user to the step that was active before the currently active one. To be more precise, it redirects to the step from which the currently active one was initially called. For example, if the Car booking dialog is currently active and was called from the Flight booking dialog, clicking `Backward` takes the user to Flight booking dialog which was previously active and called the Car booking dialog.

Forward

Clicking `Forward` redirects the user to the step that was active after the currently active one. To be more precise, it redirects to the step to which the currently active one redirected to last time. For example, the user goes from Car booking dialog to the Hotel booking dialog and returns to the Car booking dialog, which is currently active. Clicking `Forward` takes the user to the Hotel booking dialog.

Already processed steps can also be clicked to directly jump to them.

CF05 "Configuration options for Dialog State Display"
Various configuration options are available for Dialog State Display.

Configuration options for Dialog State Display:

Various configuration options are available for Dialog State Display.

Consider the following example to better understand the configuration options available to you.

Assume a travel site with the following dialogs,

- Flight booking (dialog D1) consists of the following steps Passenger information(p1), Calendar(p2), Destination(p3), Route (p4) and Airlines (p5).
- Car booking(dialog D2) consists of the following steps Renters information (p10), Calendar (p11), and Vehicle type (p12).
- Hotel booking (dialog D3) consists of the following steps Customer information (p20), Calendar (p21), and Room type (p22).
- Billing (dialog D4) consists of the following steps Account information (p30), Shopping cart (p31), and Check out (p32).

Previous Steps display preferences

You can influence how previous steps processed before the currently active step are displayed from the uppermost menu. The following are the available configuration options:

Display All Steps

All steps that are processed before the currently active step, are displayed without omitting one or more of them.

From the example, assume that the Destination (p3) step from the Flight booking dialog D1 is the active step. With this option selected the following steps are displayed.

`Passenger infromation → Calendar → Destination`

Display Custom Number of Steps

Displays the exact number of n steps that was processed most recently before the currently active step.

From the example, assume that the Destination (p3) step from the Flight booking dialog D1 is the active step . With this option selected and with n set to 1, the following steps are displayed.

`Calendar → **Destination**`

Next Steps display preferences

You can also influence how the steps that are to be processed after the currently active step are displayed from the second menu. The following are the available configuration options:

Display All Steps

All steps even those steps that are found on branches that can potentially be processed after the currently active step are being displayed without omitting one or more of them.

From the example,
Assume that the Car booking dialog D2 is called from the Flight booking dialog D1's C
Assume that the Hotel booking dialog D3 and the Billing dialog D4 are called from th
Assume that when you return from the Car booking D2, Hotel booking D3,and the Bil

Furthermore assume that you are currently in the active Passenger information(p1) and
Then, with this option selected the following steps are displayed.

```
Passenger information → Calendar → Destination → Route → Airlines
                                                    Renters information → Calendar→Vehicle
                                                         Customer information → Calendar →
                                                         Account information → Shopping ca
```

Display Custom Number of Steps

A definable number of steps even those steps that are on branches that can potentially be processed after the currently active step are being displayed.

Note: The number of steps is counted independently for each single branch.

With this option selected, two more configuration settings can be made:

Maximum Number of Steps per Branch. ***(s)***

This setting limits the total number of steps to be determined and thus displayed as the potential next steps. For example, if set to 5, the dialog state display shows no more than 5 steps per branch that are determined and displayed as the potential next steps.

Maximum Number of Branches per Step. ***(b)***

This setting limits the total number of branches to be displayed.

For example, the behavior can be described as follows:

- With b set to `unlimited` and with s set to 1 the following steps would be displayed:

  `**Passenger information** → Calendar`

  Reason: Passenger information (p1) is the active step. Potential next steps are Calendar (p2), Destination (p3), and so on. Following the path to Calendar (p2) costs one step (Calendar (p2) itself). Thus the Destination (p3) step that is also found on this branch cannot be displayed anymore as the value set for Maximum Number of Steps per Branch would be exceeded.
- With b set to `unlimited` and with s set to 2, the following steps are displayed

```
Passenger information → Calendar → Destination → Route → Airlines
                                                    Renters information → Calendar
                                                                Customer information → C
                                                                Account information → Sh
```

Reason: Passenger information (p1) is the active step. Potential next steps are Calendar (p2), Destination (p3), and so on. Following the path to Destination (p3) costs two steps Calendar(p2) and Destination (p3). The Destination (p3) step is a step that branches. Starting at Destination (p3) one can either be redirected to Route (p4) or to Renters information (p10). Following the path from Route (p4) to Airlines (p5) costs two steps Route (p4) and Airlines (p5). Thus these two steps can be displayed as it would not cause the value set for Maximum Number of Steps per Branch to be exceeded.

Following the path from Renters information (p10) to Calendar (p11) costs two steps (Renters information (p10) and Calendar (p11). Thus step Vehicle type (p12) also on this branch cannot be displayed anymore as it causes the value set for Maximum Number of Steps per Branch to be exceeded.

Route (p4) is a step that branches. Starting at Route (p4) one can either be redirected to Customer information (p20) or to Account information (p30). Following the path from Customer information (p20) to Calendar (p21) costs two steps (Customer information (p20) and Calendar (p21). Thus step Room type (p22) also on this branch cannot be displayed as it causes the value set for Maximum Number of Steps per Branch to be exceeded.

Following the path from Account information (p30) to Shopping cart (p31) costs 2 steps (Account information (p30) and Shopping cart (p31). Thus step Check out (p32) also on this branch cannot be displayed anymore as it causes the value set for Maximum Number of Steps per Branch to be exceeded.

- With b being set to 1 and with s being set to `unlimited`, the following steps are displayed.

  ```
  Passenger information → Calendar → Destination
  ```

  Reason: Passenger information (p1) is the active step. Potential next steps are Calendar (p2), Destination (p3) and so on. Destination (p3) is a step that branches. Starting at Destination (p3) one can either be redirected to Route (p4) or to Renters information (p10).Thus the number of branches that start from here is two, which does cause the value set for Maximum Number of Branches per Step to be exceeded. Thus the branches that start from Destination (p3) cannot be displayed anymore.

- With b being set to 2 and with s being set to `unlimited`, the following steps are displayed.

  ```
  Passenger information → Calendar → Destination → Route
                                                      Renters information → Calendar →
  ```

  Reason: Passenger information (p1) is the active step. Potential next steps are Calendar (p2), Destination (p3) , and so on. The step Destination (p3) is a step that branches. Starting at Destination (p3) one can either be redirected to Route (p4) or to Renters information (p10). Thus the number of branches that start from here is two, which does not cause the value set for Maximum Number of Branches per Step to be exceeded. Thus the branches that start from Destination (p3) can be displayed.

Route (p4) is a step that branches, too. Starting at Route (p4) one can either be redirected to Airlines (p5), Customer information (p20), or Account information (p30). Thus the number of branches that start from here is 3, which does cause the value set for Maximum Number of Branches per Step to be exceeded. Thus the branches that start from Route (p4) cannot be displayed anymore.

- With b being set to 3 and with s being set to `unlimited`, the following steps are displayed.

  Passenger information → Calendar → Destination → Route → Airlines
  Renters information → Calendar → Veh
  Customer information → Cale
  Account information → Shopp

  Reason: Passenger information (p1) is the active step. Potential next steps are Calendar (p2), Destination (p3) and so on. The step Destination (p3) is a step that branches. Starting at Destination (p3) one can either be redirected to Route (p4) or to Renters information (p10).Thus the number of branches that start from here is two, which does not cause the value set for Maximum Number of Branches per Step to be exceeded. Thus the branches that start from Destination (p3) can be displayed.

  Route (p4) is a step that branches, too. Starting at Route (p4) one can either be redirected to Airlines (p5), Customer information (p20), or Account information (p30). Thus the number of branches that start from here is 3, which does not cause the value set for Maximum Number of Branches per Step to be exceeded. Thus the branches that start from Route (p4) can be displayed.

Nested Dialog Display Preference

With the Nested Dialog Display Preference option, you can control how a nested non-active dialog is displayed. The following options are provided:

Display Nested Dialog Steps in Breadcrumb
With this option selected the single steps part of a nested dialog are represented as they are. They are not replaced by a single node that represents an entire nested dialog and all of its steps. For example, the following steps are displayed.

Passenger information → Calendar → Destination → Route → Airlines
Renters information → Calendar → Vehic
Customer information → Calenda
Account information → Shoppin

Display Nested Dialog as Single Step
With this option selected the single steps part of a nested dialog are represented by a single placeholder node. This single node is supposed to represent an entire nested dialog and all of its steps. For example, the following steps are displayed.

Passenger information → Calendar → Destination → Route → Airlines
Car booking
Hotel booking
Billing

Buttons to Display preference

With the check box underneath the label Buttons to Display, you can control whether extra Suspend and or Cancel buttons are to be displayed. These buttons

provide the option to cancel and suspend the current dialog the same way as in Dialog Stack.

Note: Though Dialog State Display can be used in a production scenario, it has more or less the character of a sample implementation. Depending on various factors such as the complexity of your dialogs, page design and layout, different visualizations or implementations might better satisfy your needs. Therefore, you can create your own custom, DSD implementation based on the APIs tailored to your requirements.

Configuration options

To change the overall behavior of the IBM UX Screen Flow Manager, several configuration options are available. You specify the options as Resource Environment Provider (REP) properties.

You can enable or disable the Screen Flow Manager by using a single configuration switch. The switch can be found in the REP WP_ConfigService for the portal configuration service

Property: `com.ibm.wps.pcm.enabled`
Values: `true, false`
Default: `true`
Description: If you set this property to `true`, the Screen Flow Manager is enabled. If you se

You can find all other properties that are listed here in the resource environment provider WP_PCMConfig.

Property: `com.ibm.wps.pcm.dialog.default.return.uniquename`
Value: A valid unique name of a page or portlet window
Default: None. Specify a unique name.
Description: Use this property to specify the default return target to which a user is redirect

Property: `com.ibm.wps.pcm.dialog.default.priority`
Value: `preserve, suspend`
Default: `suspend`
Description: Determines which default dialog execution priority is used if no priority is expl
The default execution priority implicitly suspends the active dialog, if a matching start trans

Property: `com.ibm.wps.pcm.dialog.default.dialogstep.display.endtransition`
Values: `true, false`
Default: `true`
Description: Use this property to determine whether the end transition is part of the set of

Property: `com.ibm.wps.pcm.dcx.jaxb.serialization.mode`
Values: `on, off, auto`
Default: `auto`
Description: Use this property to determine the JAXB marshalling mode of objects that are p

- The value `on` means that data to be stored in the DCX is always JAXB marshaled.
- The value `auto` means that the data is JAXB marshaled only if required. The decision is computed by analyzing the class loader hierarchy.
- The value `off` means that the data is never JAXB marshaled. For more information, read *Packaging of event mappers and JAXB serialization.*

Related concepts:

"Start transitions and special events" on page 3490
You can define a dialog to starts if a special start event is emitted from a specific source or from an arbitrary source.

"Packaging of event mappers and JAXB serialization" on page 3511
It is good practice to package event mappers in a shared library rather than together with the business portlets.

CF05 "Execution priority" on page 3502
You can and must explicitly model and enforce a deterministic behavior for the transitions. Even if the entire set of dialogs is deterministic, the emission of an event by a particular portlet can trigger two different transitions of two different dialogs. Therefore, you must explicitly model the intended behavior of the transitions.

Staging and migration

For staging or migration purposes, you can use the portal XML configuration interface (XMLAccess) to transfer IBM UX Screen Flow Manager related data from one system to another.

To transfer the data, run a full XMLAccess export of all dialogs that you want to transfer. Then, do an import of the exported data. For more information, go to *Deploy dialog sets by using the XML configuration interface*.

Related concepts:

"Deploy dialog sets by using the XML configuration interface" on page 3504
You can use the portal XML configuration interface (XMLAccess) to work with dialog sets.

Transitions reference

You can configure transitions in multiple ways. For example, with single portlets as source, you can configure it to transition to targets such as single portlets, multiple portlets through single or multiple transition endpoints, single page, or mixed resources. Similarly, you can configure single portlets, multiple portlets, single page, or mixed resources to become the source and transition to the target single portlet. The following reference topics show the code samples for these transitions.

CF05 "Transitions from portlets"
The following code samples show examples of the various transitions that you can configure from a single portlet as the source.

CF05 "Transitions from pages" on page 3531
The following code samples show examples of the various transitions that you can configure with the page as the source.

CF05 "Transitions from dialogs" on page 3532
The following code samples show examples of the various transitions that you can configure with the page as the source.

CF05 "Transitions from multiple resources" on page 3533
The following code sample shows a transition where the source points to multiple resources and the target points to single portlet. The multiple resources that the source points to can be pages or portlets or both pages and portlets that are marked with a particular metadata marker.

Transitions from portlets

The following code samples show examples of the various transitions that you can configure from a single portlet as the source.

Single portlet to single portlet

The transition from a single portlet to single portlet is the most simple transition. The code sample shows an example of such a transition.

The source points to a single transition endpoint a portlet, for example, in a travel site, a passenger information portlet and the target points to another single transition endpoint another portlet, for example, the Calendar portlet. After the source Passenger information portlet `portlet1` emits the event `e1`, the user is routed to the target Calendar portlet `portlet2`, which is fed with event `e2`.

Code sample

```
<transition>
    <source>
        <transition-endpoint nameref="portlet1">
             <event qname="e1"/>
         </transition-endpoint>
     </source>
     <target>
         <transition-endpoint nameref="portlet2">
              <event qname="e2"/>
          </transition-endpoint>
     </target>
</transition>
```

Single portlet to multiple portlets

This code sample shows a transition from a single portlet to multiple portlets where the target points to multiple portlets instead of a single one.

Some portlets are associated with different events than others. In the code sample, both the portlets that are referenced by the transition-endpoints `portlet2a` and `portlet2b` receive the event `e2a`. The portlet that is referenced by the transition endpoint `portlet2c` receives the event `e2c`.

Note: In case multiple portlets are referenced as part of a target, all portlets must be on the same page.

Code sample

```
<transition>
    <source>
        <transition-endpoint nameref="portlet1">
             <event qname="e1"/>
        </transition-endpoint>
    </source>
    <target>
        <transition-endpoint nameref="portlet2a">
              <event qname="e2a"/>
        </transition-endpoint>
        <transition-endpoint nameref="portlet2b">
              <event qname="e2a"/>
        </transition-endpoint>
        <transition-endpoint nameref="portlet2c">
              <event qname="e2c"/>
        </transition-endpoint>
    </target>
</transition>
```

Single portlet to page

This code sample shows a transition from a single portlet to a page instead of a portlet. This transition causes the event to be propagated to all portlets that are found on the page.

All portlets that are found on the page that is referenced by the transition-endpoint `page2` receives the event `e2`.

Code sample

```
<transition>
    <source>
        <transition-endpoint nameref="portlet1">
            <event qname="e1"/>
        </transition-endpoint>
    </source>
    <target>
        <transition-endpoint nameref="page2">
             <event qname="e2"/>
         </transition-endpoint>
     </target>
</transition>
```

Single portlet to a page and multiple portlets

This code sample shows a transition with the target that points to page and multiple portlets. The page and each portlet are associated with a different event. This transition causes each portlet to receive the event that is associated with the page and the event that is associated with the particular portlet.

In code sample, the portlet that is referenced by the transition-endpoint receives the events as follows

- Transition-endpoint `portlet2a` receives the events `e2` and `e2a`.
- Transition-endpoint `portlet2b` receives the events `e2` and `e2a`.
- Transition-endpoint `portlet2c` receives the events `e2` and `e2c`.

Note: If pages and portlets are both referenced as targets, all portlets must be on the referenced page.

Code sample:

```
<transition>
    <source>
        <transition-endpoint nameref="portlet1">
             <event qname="e1"/>
        </transition-endpoint>
     </source>
     <target>
          <transition-endpoint nameref="page2">
              <event qname="e2"/>
          </transition-endpoint>
          <transition-endpoint nameref="portlet2a">
               <event qname="e2a'/>
          </transition-endpoint>
          <transition-endpoint nameref="portlet2b">
               <event qname="e2a'/>
          </transition-endpoint>
          <transition-endpoint nameref="portlet2c">
```

```
            <event qname="e2c'/>
        </transition-endpoint>
     </target>
</transition>
```

The following code sample shows an alternative way to declare what is declared in the previous code sample. Two of the previously shown transition-endpoints are merged into a single one. Both notations lead to the same effect.

Code sample

```
<transition>
    <source>
        <transition-endpoint nameref="portlet1">
            <event qname="e1"/>
        </transition-endpoint>
     </source>
     <target>
          <transition-endpoint nameref="page2">
              <event qname="e2"/>
          </transition-endpoint>
          <transition-endpoint nameref="portlet2a 2b">
               <event qname="e2a'/>
          </transition-endpoint>
          <transition-endpoint nameref="portlet2c">
               <event qname="e2c'/>
          </transition-endpoint>
      </target>
</transition>
```

It is also possible to transmit multiple events to a single target transition-endpoint. In the following code sample, transition-endpoints are associated with events as shown

- The page that is referenced by the transition-endpoint `page2` is associated with the events `e2-1` and `e2-2`.
- The portlet that is referenced by the transition-endpoint `portlet2a` is associated with the events `e2a-1` and `e2a-2`.
- The portlet that is referenced by the transition-endpoint `portlet2b` is associated with the events `e2b-1`.

In this particular sample, `portlet2a` receives the events `e2-1`,`e2-2`, `e2a-1`, and `e2a-2` and `portlet2b` receives the events `e2-1`,`e2-2`, and `e2b-1`.

Note: Transmission of multiple events is supported only for targets.

Code sample

```
<transition>
    <source>
        <transition-endpoint nameref="portlet1">
            <event qname="e1"/>
        </transition-endpoint>
     </source>
     <target>
          <transition-endpoint nameref="page2">
              <event qname="e2-1"/>
              <event qname="e2-2"/>
          </transition-endpoint>
          <transition-endpoint nameref="portlet2a">
               <event qname="e2a-1"/>
               <event qname="e2a-2"/>
          </transition-endpoint>
          <transition-endpoint nameref="portlet2b">
```

```
            <event qname="e2b-1"/>
        </transition-endpoint>
    </target>
</transition>
```

Multiple portlets to single portlet

The following code sample shows a transition where the source points to multiple portlets and the target points to a single portlet.

When any of the referenced source portlets, `portlet1a` or `portlet1b`, emits the event `e1`, the user is routed to the target portlet `portlet2`, which is fed with event `e2`.

Note: For sources, it is not possible to alternatively reference multiple transition-endpoints as it was done with targets.

Code sample

```
<transition>
    <source>
        <transition-endpoint nameref="portlet1a_1b">
            <event qname="e1"/>
        </transition-endpoint>
    </source>
    <target>
        <transition-endpoint nameref="portlet2">
            <event qname="e2"/>
        </transition-endpoint>
    </target>
</transition>
```

Single portlet to another dialog

The following code sample shows a transition where the source points to a single portlet and the target to another dialog.

This transition represents the outgoing transition. When the transition is triggered, it starts the referenced dialog, which is initialized by the defined event. For more information about the outgoing transition, go to *Transition Endpoints*.

In this code sample, when `portlet1` emits the event `e1`, `dialog2` is started and initialized with event `eX`. From here, continuing to a particular step of `dialog2` depends on the transitions that are defined as part of `dialog2`. For more information, go to the topic *Dialog Chaining and Nesting*.

Code sample

```
<transition type="nested">
   <source>
       <transition-endpoint nameref="portlet1">
           <event qname="e1"/>
       </transition-endpoint>
    </source>
    <target>
        <transition-endpoint nameref="dialog2">
            <event qname="eX"/>
        </transition-endpoint>
    </target>
</transition>
```

Related concepts:

CF05 “Dialog chaining and nesting” on page 3494
Dialogs can start other dialogs. Dialogs that start other dialogs are referred to as calling dialogs and the dialogs that are being started are referred to as called dialogs. Two different options are available to complete this action such as dialog chaining and dialog nesting.

CF05 “Transition endpoints” on page 3480
Resources that are part of a screen flow which marks an endpoint, that is, the source or target of a transition are referred to as transition endpoints. Resources can be pages and portlets and they can also wrap widgets or forms. A particular page or portlet that is the active step in a screen flow is referred to as the source and the potential next steps are referred to as targets.

Transitions from pages

The following code samples show examples of the various transitions that you can configure with the page as the source.

Single page to single portlet

The following code sample shows a transition where the source points to a single page and the target points to single portlet.

When any portlet on the referenced page `page1` emits the event `e1`, the user is routed to the target portlet `portlet2`, which is fed with event `e2`.

Code sample

```
<transition>
    <source>
        <transition-endpoint nameref="page1">
            <event qname="e1"/>
         </transition-endpoint>
     </source>
     <target>
         <transition-endpoint nameref="portlet2">
              <event qname="e2"/>
          </transition-endpoint>
      </target>
</transition>
```

Multiple pages to single portlet

The following code sample shows a transition where the source points to multiple pages and the target points to a single portlet.

When any portlet on any of the referenced pages, `page1a` or `page1b`, emits the event `e1`, the user is routed to the target portlet `portlet2`. The target portlet `portlet2` is then fed with event `e2`.

Note: For sources, it is not possible to alternatively reference multiple transition endpoints.

Code sample

```
<transition>
   <source>
       <transition-endpoint nameref="page1a_1b">
            <event qname="e1"/>
       </transition-endpoint>
    </source>
    <target>
```

```
        <transition-endpoint nameref="portlet2">
            <event qname="e2"/>
         </transition-endpoint>
     </target>
</transition>
```

Page hierarchy to single portlet

The following code sample shows a transition where the source points to a page hierarchy and the target points to single portlet.

When any portlet on any page part of the referenced hierarchy emits the event `e1`, the user is routed to the target portlet `portlet2`. The target portlet `portlet2` is then fed with event `e2`.

Code sample

```
<transition>
   <source>
       <transition-endpoint nameref="pageHierarchy1">
            <event qname="e1"/>
       </transition-endpoint>
    </source>
    <target>
        <transition-endpoint nameref="portlet2">
            <event qname="e2"/>
         </transition-endpoint>
     </target>
</transition>
```

Transitions from dialogs

The following code samples show examples of the various transitions that you can configure with the page as the source.

Single dialog to single portlet

The following code sample shows a transition where the source points to a single dialog and the target to a single portlet.

The transition represents the incoming transition. When the transition is triggered, it causes the continuation of the dialog that the transition is returned to. For more information about incoming transitions, go to *Transition Endpoints*.

In this code sample, when `dialog2` emits the event `eX`, `dialog1` is continued and is initialized with event `e2`.

Code sample

```
<transition>
   <source>
       <transition-endpoint nameref="dialog2">
            <event qname="eX"/>
       </transition-endpoint>
    </source>
    <target>
        <transition-endpoint nameref="portlet2">
            <event qname="e2"/>
         </transition-endpoint>
     </target>
</transition>
```

Dialog to dialog

The following code sample shows a transition where the source and the target point to dialogs. This transition represents the dispatching transition.

A dispatching transition is triggered when the transition returns from one dialog and it causes another dialog to start. In this sample when `dialog2` emits the event `eX`, `dialog3` is started and is initialized with the event `eY`. For more information about dispatching transition, go to *Dialog Chaining and Nesting*.

Code sample

```
<transition type="nested">
   <source>
       <transition-endpoint nameref="dialog2">
            <event qname="eX"/>
       </transition-endpoint>
    </source>
    <target>
        <transition-endpoint nameref="dialog3">
            <event qname="eY"/>
         </transition-endpoint>
     </target>
</transition>
```

Related concepts:

CF05 "Dialog chaining and nesting" on page 3494
Dialogs can start other dialogs. Dialogs that start other dialogs are referred to as calling dialogs and the dialogs that are being started are referred to as called dialogs. Two different options are available to complete this action such as dialog chaining and dialog nesting.

CF05 "Transition endpoints" on page 3480
Resources that are part of a screen flow which marks an endpoint, that is, the source or target of a transition are referred to as transition endpoints. Resources can be pages and portlets and they can also wrap widgets or forms. A particular page or portlet that is the active step in a screen flow is referred to as the source and the potential next steps are referred to as targets.

Transitions from multiple resources

The following code sample shows a transition where the source points to multiple resources and the target points to single portlet. The multiple resources that the source points to can be pages or portlets or both pages and portlets that are marked with a particular metadata marker.

Multiple resources to single portlet

When any marked portlet or any portlet on a marked page emits the event `e1`, the user is routed to the target portlet `portlet2`. The target portlet `portlet2` is then fed with the event `e2`.

Code sample

```
<transition>
   <source>
       <transition-endpoint nameref="marked_resources">
            <event qname="e1"/>
       </transition-endpoint>
    </source>
    <target>
        <transition-endpoint nameref="portlet2">
```

```
            <event qname="e2"/>
          </transition-endpoint>
      </target>
</transition>
```

Sample screen flow application

The IBM UX Screen Flow Manager package can be downloaded from the Greenhouse site and includes a sample screen flow application that is ready to use. This sample guides a portal site visitor through a simple travel booking flow. It consists of a set of pages, portlets, and dialog definitions (DDs). For more details about DDs, go to Creating dialog definitions.

The sample includes the following components:

- The following main pages and portlets:
  - Travel Demo. This demonstration page is the root of the sample. The Travel Demo page provides entry fields where a site visitor can enter details about the intended travel. The IBM UX Screen Flow Manager then passes these details to the subsequent pages where the site visitor can book the appropriate flights, hotels, cars, and insurance.
  - Flight Booking
  - Hotel Booking
  - Car Booking
  - Insurance Booking
  - Travel Summary. The page provides a summary of the booking that is done by the site visitor.
- An exemplary Dialog State Display (DSD). It is shown on each page. The site visitor can use this DSD to go back and forth between the individual steps or to suspend or cancel the currently active dialog instance. For more information about DSDs, go to *Dialog State Display*.
- An extra page named Dialog Stack. It provides access to a Dialog Stack (DS). For more information about the Dialog Stack, go to *Dialog Stack*. The site visitor can use the Dialog Stack to view the currently active dialog instance and to return to it or to suspend or cancel it. It also provides an overview of all dialog instances that was previously suspended. The site visitor can resume or cancel these dialog instances.
- Three Dialog Definitions (DDs). The dialog definition that you use determines the order in which the IBM UX Screen Flow Manager routes the site visitor through the flow. By replacing the DD, you can easily change a screen flow without having to change code.

To install the sample, the following are the steps:

1. Deploy the sample application WAR file that is included with the installation package. The file contains the sample portlets that are mentioned before.
2. Create the pages that are mentioned before. To create the pages, run the following XML configuration interface (XMLAccess) command:

   Code sample

   ```
   xmlaccess.{sh|bat} -in deployPagesAndPortlets.xml -user userID  -password password  -url http://h
   ```
3. Import one of the Dialog Definitions included in the installation package. To import, run the appropriate XMLAccess command. The following code sample shows the syntax for the XMLAccess commands for all three dialog definitions.

Select the dialog definition that you import, depending on the order in which you want the IBM UX Screen Flow Manager to route the site visitor through the flow.

Code sample

```
xmlaccess.{sh|bat} -in dialogDefinitions_FlightCarHotel.xml -user userID  -password password -

xmlaccess.{sh|bat} -in dialogDefinitions_FlightHotelCar.xml -user userID -password password -u

xmlaccess.{sh|bat} -in dialogDefinitions_FlightHotelCarInsurance.xml -user userID -password pa
```

Related concepts:

"Creating dialog definitions" on page 3479
With the Screen Flow Manager, different teams or even third-party vendors can develop different types of user interface artifacts. The user puts together the correct set of user interface artifacts and creates the declarative model in XML known as the dialog definition. The dialog definition describes the specific screen flow that is also known as dialog, which consists of multiple steps and single steps referred to as subdialogs. The dialog definition contains all information about the subdialogs that participate and the transitions that route the user from one subdialog to another.

CF05 "Dialog State Display" on page 3520
The Dialog State Display (DSD) displays a dialogs current state. You can move forward and backward or to jump to a dedicated step that was processed before. It is a generic navigation component that can be easily added to any page that participates in a dialog.

CF05 "Dialog Stack" on page 3518
The Dialog Stack provides an overview of active and suspended dialogs. It allows for suspending, resuming, and canceling the dialogs.

Related information:

IBM UX Screen Flow Manager for WebSphere Portal Demo

Chapter 17. Troubleshooting

To help you resolve problems, use diagnostic tools such as IBM Support Assistant and tracing to capture system errors.

"Tools for troubleshooting and diagnostics"
A number of tools and resources are available to help you troubleshoot issues and resolve problems that users might encounter while using IBM WebSphere Portal Express. If you need further assistance, you can use the tools described here to identify and collect information to help IBM Support determine the underlying cause of a problem.

"Logging and tracing" on page 3541
If you are experiencing a problem, you might want to enable tracing and then re-create the problem to capture more log information. You can enable logging and tracing for software that is included with WebSphere Portal Express. Enabling tracing makes log output more verbose. For example, you can enable tracing within WebSphere Application Server to obtain information about application servers and other processes.

"Troubleshooting the Configuration Wizard" on page 3559
The configuration wizard provides tools for troubleshooting and recovering from errors, such as logs, step commands, and reset options.

"Error message codes" on page 3615
Each message code consists of a product identifier, component identifier, a unique number, and a message type identifier. The product identifier is EJP and there are many components. There are three message types: error, information, and warning.

"Contact support" on page 3615
For contact information, see to the IBM Software Support site.

Related information:

Late breaking migration limitations and issues

Technotes for installation and configuration issues

Technotes for database connectivity limitations and issues

Problem determination and troubleshooting technotes

IBM WebSphere Portal Performance Troubleshooting Guide

Tools for troubleshooting and diagnostics

A number of tools and resources are available to help you troubleshoot issues and resolve problems that users might encounter while using IBM WebSphere Portal Express. If you need further assistance, you can use the tools described here to identify and collect information to help IBM Support determine the underlying cause of a problem.

"IBM Support Assistant" on page 3538
IBM Support Assistant (ISA) provides quick access to product, education, and support resources. These resources can help you answer questions and resolve problems with IBM Software products on your own, without needing to contact IBM Support. You can customize IBM Support Assistant for the particular

products that you installed with different product-specific plug-ins. IBM Support Assistant can also collect system data, log files, and other information to help IBM Support determine the cause of a particular problem.

"Data collection and symptom analysis" on page 3539
There is one method to collect data and analyze symptoms for problem determination scenarios. You run a task that can collect and optionally send the data for you. Starting with IBM WebSphere Portal Express version 8.5, there is now a task to collect the configuration wizard logs. This task is only necessary if the wizard fails before the steps to create the `wp_profile/ConfigEngine` instance.

"Portal version and history information" on page 3540
You can use the IBM WebSphere Portal Express version and history information tools to gather information about your portal installation. This information can be useful when you need a snapshot of your portal installation specifics, for example when you contact customer support. This information is automatically included in the automated data collection that is available when you use the IBM Support Assistant Lite for WebSphere Portal Express.

IBM Support Assistant

IBM Support Assistant (ISA) provides quick access to product, education, and support resources. These resources can help you answer questions and resolve problems with IBM Software products on your own, without needing to contact IBM Support. You can customize IBM Support Assistant for the particular products that you installed with different product-specific plug-ins. IBM Support Assistant can also collect system data, log files, and other information to help IBM Support determine the cause of a particular problem.

IBM Support Assistant is a utility to be installed on an administrator's workstation, not directly onto the WebSphere Portal Express or Web Content Manager server system itself. The memory and resource requirements for the Assistant might negatively impact the server's performance. The included portable diagnostic components are designed for minimal impact to the normal operation of a server.

You can use IBM Support Assistant to help you in the following ways:

- To search through IBM and non-IBM knowledge and information sources, across multiple IBM products to answer a question or solve a problem
- To find additional information through product-specific web resources. Including information from product and support home pages, customer news groups and forums, skills and training resources and information about troubleshooting and commonly asked questions
- To extend your ability to diagnose product-specific problems with targeted diagnostic tools available through the Support Assistant
- To simplify collection of diagnostic data. This action helps you and IBM resolve your problems (collecting either general or product/symptom-specific data)
- To help report problem incidents to IBM Support through a customized online interface. You can also attach the previous diagnostic data or any other information to new or existing incidents

Finally, you can use the built-in Updater facility to obtain support for more software products and capabilities as they become available.

More information, the download package, installation instructions, and the latest version of the IBM Support Assistant are available from the IBM Support Assistant web page at `www.ibm.com/software/support/isa/`.

Related information:

IBM Support Assistant

Data collection and symptom analysis

There is one method to collect data and analyze symptoms for problem determination scenarios. You run a task that can collect and optionally send the data for you. Starting with IBM WebSphere Portal Express version 8.5, there is now a task to collect the configuration wizard logs. This task is only necessary if the wizard fails before the steps to create the `wp_profile/ConfigEngine` instance.

`wpcollector` tool

Complete the following steps:

1. If the support team requested tracing, enable it now as instructed and then re-create the problem. If no tracing is requested, skip to the next step.
2. Open a command prompt and change to the *wp_profile_root*`/PortalServer/bin/` directory.

   Attention: You must run the **`wpcollector`** task from the *wp_profile_root*`/PortalServer/bin/` directory. If you run the task from a different directory, the task fails.
3. Run the following script to collect data:
   - Linux : `./wpcollector.sh`
   - IBM i: `wpcollector.sh`
   - Windows: `wpcollector.bat`

   Tip: The `wpcollector` script can automatically FTP the logs to IBM so that you do not need to manually transfer them. To begin the collection and FTP the results to IBM correctly, use the `-Dpmr=`*`pmr_number`* parameter to identify the collection with your PMR (Problem Management Record) number. Format the number either with periods or commas. For example: `wpcollector.bat -Dpmr=`*`12345.xxx.000`*

   To collect files for the Deployment Manager profile, use the `-Ddmgr.root=`*`dmgr_root`* parameter either alone or with `-Dpmr=`*`pmr_number`*.
4. If you did not automatically FTP your results, locate the `wp.mustgather.zip` file or the *`pmr`*`-wp.mustgather-`*`timestamp`*`.zip` file in the *wp_profile_root*`/filesForAutoPD/` directory. Follow the instructions in "Exchanging information with IBM Technical Support for problem determination" to manually FTP your results.

Restriction: If you try to extract the `wp.mustgather.zip` file, some collections might not expand properly if the path name exceeds the 256 character limitation.

`cwcollector` tool

Complete the following steps if the configuration wizard failed before it created the `wp_profile/ConfigEngine` instance:

1. Open a command prompt and change to the *AppServer_root*`/ConfigEngine` directory.
2. Run the following task to collect the configuration wizard logs:

   Tip: The logs are compressed and placed into the *AppServer_root*`/filesForAutoPD` directory.

- Linux : `./ConfigEngine.sh collect-cw-logs -DPortalBinaryLocation=/opt/IBM/WebSphere/PortalServer -DWasPassword=password`
- IBM i: `ConfigEngine.sh collect-cw-logs -DPortalBinaryLocation=/QIBM/ProdData/WebSphere/PortalServer/V85/Server -DWasPassword=password`
- Windows: `ConfigEngine.bat collect-cw-logs -DPortalBinaryLocation=C:/IBM/WebSphere/PortalServer -DWasPassword=password`

Tip: The **collect-cw-logs** script can automatically FTP the logs to IBM so that you do not need to manually transfer them. To begin the collection and FTP the results to IBM correctly, use the `-Dpmr=pmr_number` parameter to identify the collection with your information. Format the number either with periods or commas. **Attention:** If the **collect-cw-logs** task fails, run the **stopserver server1** command from the *AppServer_root*`/bin` directory. Then, rerun the **collect-cw-logs** task.

3. If you did not automatically FTP your results, locate the `cw.mustgather.zip` file or the *pmr*`-cw.mustgather-`*timestamp*`.zip` file in the *AppServer_root*`/filesForAutoPD` directory. Follow the instructions in "Exchanging information with IBM Technical Support for problem determination" to manually FTP your results.

Related information:

Self-Help Central for WebSphere Portal

Exchanging information with IBM Technical Support for problem determination

Portal version and history information

You can use the IBM WebSphere Portal Express version and history information tools to gather information about your portal installation. This information can be useful when you need a snapshot of your portal installation specifics, for example when you contact customer support. This information is automatically included in the automated data collection that is available when you use the IBM Support Assistant Lite for WebSphere Portal Express.

Version information

The portal version information tool is located in the following directory:

- Linux : *wp_profile_root*`/PortalServer/bin`
- IBM i: *wp_profile_root*`/PortalServer/bin`
- Windows: *wp_profile_root*`\PortalServer\bin`

You invoke the tool by using the following command:

- Linux : `./WPVersionInfo.sh`
- IBM i: `WPVersionInfo.sh`
- Windows: `WPVersionInfo.bat`

You can also generate a report in html format by executing the genVersionReport tool

- Linux : `./genVersionReport.sh`

- IBM i: `genVersionReport.sh`
- Windows: `genVersionReport.bat`

History information

The History information tool can be used to gather installation history for the WebSphere Portal Express product. The History information tool is located in the following directory:

- Linux : *wp_profile_root*`/PortalServer/bin`
- IBM i: *wp_profile_root*`/PortalServer/bin`
- Windows: *wp_profile_root*`\PortalServer\bin`

The History information tool can be invoked using the following command:

- Linux : `./WPHistoryInfo.sh`
- IBM i: `WPHistoryInfo.sh`
- Windows: `WPHistoryInfo.bat`

You can also generate a report in HTML format by executing the `genHistoryReport` tool:

- Linux : `./genHistoryReport.sh`
- IBM i: `genHistoryReport.sh`
- Windows: `genHistoryReport.bat`

Related concepts:

"Data collection and symptom analysis" on page 3539
There is one method to collect data and analyze symptoms for problem determination scenarios. You run a task that can collect and optionally send the data for you. Starting with IBM WebSphere Portal Express version 8.5, there is now a task to collect the configuration wizard logs. This task is only necessary if the wizard fails before the steps to create the `wp_profile/ConfigEngine` instance.

Logging and tracing

If you are experiencing a problem, you might want to enable tracing and then re-create the problem to capture more log information. You can enable logging and tracing for software that is included with WebSphere Portal Express. Enabling tracing makes log output more verbose. For example, you can enable tracing within WebSphere Application Server to obtain information about application servers and other processes.

Refer to the MustGather data collection lists used in troubleshooting various problems in WebSphere Portal Express and IBM Web Content Manager. Collecting MustGather data early, even before you open a PMR, helps IBM Product Support quickly determine whether

- Symptoms match known problems (rediscovery).
- A non-defect problem can be identified and resolved.
- A defect identifies a workaround to reduce severity.
- Locating the root cause can speed development of a code fix.

Simplify this process even more by using the IBM Support Assistant Lite for WebSphere Portal Express to automate the collection of the diagnostic data that is needed to troubleshoot most of these situations. You can use the information

gathered to help solve your own problems or to report an issue to IBM Product Support.

Links to important WebSphere Portal Express tracing questions

How do I turn on WebSphere Portal Express trace logging?
See "Trace logging" on page 3551 for information.

What are the different trace settings and where are they logged?
See "WebSphere Portal Express runtime logs" on page 3544 for information.

How do I change the location of my logs?
See "Changing the log file name and location" on page 3552

"Installation and migration logs" on page 3543
Learn about the different log files that WebSphere Portal Express provides to help administrators identify and correct problems with installation and migration.

"WebSphere Portal Express runtime logs" on page 3544
If tracing is enabled, IBM WebSphere Portal Express generates a log file during run time that contains messages and trace information.

"Verbose garbage collection in Java Virtual Machine (JVM) logs" on page 3549
Verbose garbage collection (verbosegc) logging is often required when tuning and debugging many issues, particularly memory problems, and has negligible impact on system performance.

"WebSphere Application Server tracing and log files" on page 3549
Use WebSphere Application Server log files and tracing to troubleshoot problems with WebSphere Portal Express.

"Configuration Wizard log files" on page 3550
The configuration wizard generates log files each time you run it. The log file can help you to debug problems.

"Enabling Virtual Member Manager tracing files" on page 3550
Enable WebSphere Application Server trace facilities to create trace information for Virtual Member Manager.

"System event logging" on page 3551
The system event logging facility of IBM WebSphere Portal Express enables the recording of information about the operation of WebSphere Portal Express.

"Web Content Manager tracing files" on page 3554
Enable the use of WebSphere Application Server trace facilities to create trace information for Web Content Manager. This tracing can be enabled either permanently or for just the current WebSphere Portal Express session.

"Logging and tracing client side rendering" on page 3557
Portal pages that are rendered in client side aggregation mode differ in their logging behavior from portal pages rendered in server side mode. For client side rendered pages, a considerable amount of code is written in JavaScript that is executed in the corresponding browser JavaScript engine rather than on the server. As a result, the corresponding logging and tracing information is collected in the browser and not on the server side.

Related concepts:

"Data collection and symptom analysis" on page 3539
There is one method to collect data and analyze symptoms for problem determination scenarios. You run a task that can collect and optionally send the data for you. Starting with IBM WebSphere Portal Express version 8.5, there is now a task to collect the configuration wizard logs. This task is only necessary if the wizard fails before the steps to create the `wp_profile/ConfigEngine` instance.

Related information:

Collecting Data (MustGather): Read first for IBM WebSphere Portal

Installation and migration logs

Learn about the different log files that WebSphere Portal Express provides to help administrators identify and correct problems with installation and migration.

Installation log files

The IBM Installation Manager controls the WebSphere Portal installation log files. The IBM Installation Manager records any errors or warnings that occur during the installation. The IBM Installation Manager stores all logs in a centralized location and the logs pertaining to WebSphere Portal can be found there. Typically these logs are found in the following directory:

- Linux : `/var/ibm/InstallationManager/logs`
- IBM i: `/QIBM/UserData/InstallationManager/logs`
- Windows: `C:\ProgramData\IBM\Installation Manager\logs`

The IBM Installation Manager retains logs from all installations and uninstallations that it does for all products. You can review your entire history. The IBM Installation Manager has a built-in function that you can use to view the logs. On the IBM Installation Manager main page, click **File** > **View log**. You can also use your web browser to view the `index.xml` file in the log directory.

Migration log files

Unless noted otherwise, migration log files are in the following directory:

- Linux : *wp_profile_root* `/logs`
- IBM i: *wp_profile_root*`/logs`

  Note: Files in this directory are stored only on the local System i5® workstation. You might be able to view them from a Windows workstation by mapping a network drive to the System i5 workstation.
- Windows: *wp_profile_root* `\logs`

The table lists each file, describes the file content, and recommends when to check the file for information that might help troubleshooting migration problems.

Table 505. Migration log files for IBM i Linux Windows

Log file name	Description	Problem symptoms
`ConfigTrace.log`	Contains information that is generated each time a ConfigEngine task is run, including trace information that is generated during migration. This file is in *wp_profile_root*`/ConfigEngine/log`.	Check this log if migration stops before successful completion.

Table 505. Migration log files for IBM i Linux Windows (continued)

Log file name	Description	Problem symptoms
upgradeConfigEngineTrace.log	Contains trace information that is generated when the ConfigEngine tool is upgraded. This file is in *wp_profile_root*/ConfigEngine/log	Check this log if errors occur when you run the upgradeConfigEngine tool.

To set tracing for the migration plug-in com.ibm.wp.was.plugin.jar file, which you use to migrate the Deployment Manager and nodes, specify com.ibm.wp.migration.*=all.

WebSphere Portal Express runtime logs

If tracing is enabled, IBM WebSphere Portal Express generates a log file during run time that contains messages and trace information.

The default runtime log file is shown:

- Linux : *wp_profile_root*/logs/WebSphere_Portal/trace.log
- IBM i (UserData): *wp_profile_root*/logs/WebSphere_Portal/trace.log
- Windows: *wp_profile_root*\logs\WebSphere_Portal\trace.log

See the topic on system event logging for details on how to configure logging and for information on the grammar of the "trace string" configuration key.

The following information describes trace loggers for particular situations and problem symptoms. Enabling the trace loggers can slow down WebSphere Portal Express.

Note: If there are problems with portal administration portlets, the error is not caused by the portlet code itself, but by the underlying function for which the portlet provides the UI. Therefore, the portlet trace strings are not listed here. If there are issues with these portlets, provide the trace strings of the underlying function. If you need traces or logs for portlets, you can learn how to obtain them from support personnel.

Access Control

When to use

Enable this tracer if you want permissions for resources to be explained in detail, need to verify the correctness of a permission, or need to isolate a defect in access control.

Trace String

com.ibm.wps.ac.*=all

Additional comments

The traces are easier to evaluate while WebSphere Portal Express usage is low.

Important: Enabling this logger creates large log files.

Authentication

Trace String

```
com.ibm.wps.services.puma.*=all:
com.ibm.wps.puma.*=all:
com.ibm.wps.auth.*=all:
com.ibm.wps.sso.*=all:
com.ibm.wps.um.*=all:
com.ibm.wps.services.authentication.*=all
```

Command

When to use

Use to turn on all command trace loggers.

Trace String

```
com.ibm.wps.commands.*=all
```

Layout Model

When to use

Enable these messages if you want to get more information on how pages are constructed, need to verify page lists that are displayed on WebSphere Portal Express for correctness, or need to isolate an error in the WebSphere Portal Express aggregation component.

Trace String

```
com.ibm.wps.model.*=all:
com.ibm.wps.composition.*=all
```

Additional comments

The traces are easier to evaluate while WebSphere Portal Express usage is low.

Important: Enabling this logger creates large log files.

Credential Vault

Trace String

```
com.ibm.wps.sso.credentialvault.*=all:
com.ibm.wps.command.credentialvault.*=all:
com.ibm.wps.portletservice.credentialvault.*=all:
com.ibm.wps.services.credentialvault.*=all:
com.ibm.portal.portlet.service.credentialvault.*=all
```

Database

When to use

Deals with generated SQL statements and the internal flow in the WebSphere Portal Express database layer.

Trace String

```
com.ibm.wps.datastore.*=all:
com.ibm.wps.services.datastore.*=all
```

Additional comments

Important: Enabling this logger creates large log files.

Engine

When to use

Use to enable all engine trace loggers.

Trace string

```
com.ibm.wps.engine.*=all
```

General

Trace string

```
com.ibm.wps.*=all
```

Note: If you want to use general tracing but do not want render times to be displayed for such portlets, you must selectively disable tracing by using the following trace string:

```
com.ibm.wps.pe.PortletRenderTimeLoggingHelper=info
```

Mail Service

When to use

Use to diagnose problems with the Mail Service.

Trace string

```
com.ibm.wps.services.mail.*=all
```

Mapping URLs

When to use

Use to diagnose problems with the user-defined mappings of URLs.

Trace string

```
com.ibm.wps.mappingurl.*=all:
com.ibm.wps.command.mappingurl.*=all
```

Personalization

Trace string

```
com.ibm.websphere.personalization.*=all:
com.ibm.dm.pzn.ui.*=all
```

Additional comments

When Personalization is installed outside of a WebSphere Portal Express server, Personalization logs by using WebSphere Application Server tracing with the same trace strings.

Portlet Container

Trace string

```
com.ibm.wps.pe.pc.*=all:
org.apache.jetspeed.portlet.Portlet=all:
javax.portlet.Portlet=all
```

Portlet Environment

Trace string

```
com.ibm.wps.pe.ext.*=all:
com.ibm.wps.pe.factory.*=all:
com.ibm.wps.pe.om.*=all:
com.ibm.wps.pe.util.*=all
```

Portlet Load Monitoring

When to use

Use to diagnose problems with Portlet Load Monitoring (PLM).

Trace string

```
com.ibm.wps.pe.pc.waspc.plm.*=all:
com.ibm.wps.command.plm.*=all
```

Deployment

Trace string

```
com.ibm.wps.pe.mgr.*=all:
com.ibm.wps.services.deployment.*=all:
com.ibm.wps.command.applications.*=all:
com.ibm.wps.command.portlets.*=all
```

Portlets

When to use

Use to diagnose problems with portlets.

Trace string

```
com.ibm.wps.portlets.*=all:
org.apache.jetspeed.portlet.PortletLog=all
```

Additional comments

Enables tracing for all portlets. Therefore, place the suspect portlet on a separate page for testing.

Scripting Interface

When to use

Use this trace string to diagnose problems with the Portal Scripting Interface, or with application interface scripting, and the execution of such scripts.

Trace string

```
com.ibm.wps.scripting.*=all
```

Additional comments

The traces are easier to evaluate while portal usage is low.

Note: Enabling this logger can create large log files fast.

Selfcare

When to use

Use to diagnose problems with user registration and profile editing.

Trace string

```
com.ibm.wps.services.puma.*=all:
com.ibm.wps.puma.*=all:
com.ibm.wps.um.*=all
```

Additional comments

Use this logger if there are errors in the sign-up, Edit My Profile, and the Manage Users and Groups portlets.

Services: EventBroker

Trace string

```
com.ibm.wps.services.registry.EventHandlerRegistry=all:
com.ibm.wps.services.events.*=all
```

Services: Finder

When to use

Use for debugging the resolution of file names.

Trace string

```
com.ibm.wps.services.finder.*=all
```

Services: Loader

When to use

Use to trace the dynamic class loading that is done by this service.

Trace string

```
com.ibm.wps.services.ServiceManager=all
```

ServicesNaming

When to use

Use to debug the lookup of objects by the naming service.

Trace string

```
com.ibm.wps.services.naming.*=all
```

ServicesNavigator

When to use

Use to diagnose problems with parts of page aggregation and display.

Trace string

```
com.ibm.wps.services.navigator.*=all
```

ServicesRegistry

When to use

Use to view the policies of the internal portlet object caching and watch it reload its content.

Trace string

```
com.ibm.wps.services.registry.*=all
```

Services

When to use

Use for turning on tracing for all services.

Trace string

```
com.ibm.wps.services.*=all
```

SSO

When to use

Use to turn on all SSO tracer loggers.

Trace string

```
com.ibm.wps.sso.*=all
```

Additional comments

Use this logger if errors occur when you use the Security Vault task on the Security page of the Administration pages.

WSRP administration

When to use

Use to diagnose problems that occur during the administration of Web Services for Remote Portlets (WSRP) with WebSphere Portal Express.

Trace string

```
com.ibm.wps.command.wsrp.*=all:
com.ibm.wps.wsrp.cmd.*=all
```

WSRP Consumer

When to use

Use to diagnose problems that occur during the use of WSRP with WebSphere Portal Express as a Consumer.

Trace string

```
com.ibm.wps.wsrp.consumer.*=all
```

WSRP Producer

When to use

Use to diagnose problems that occur during the use of WSRP with WebSphere Portal Express as a Producer.

Trace string

```
com.ibm.wps.wsrp.producer.*=all
```

XML configuration interface

When to use

Use to diagnose problems with the XML import/export of WebSphere Portal Express configurations.

Trace string

```
com.ibm.wps.command.xml.*=all
```

Verbose garbage collection in Java Virtual Machine (JVM) logs

Verbose garbage collection (verbosegc) logging is often required when tuning and debugging many issues, particularly memory problems, and has negligible impact on system performance.

The default WebSphere Portal Express installation enables verbosegc logging and configures the following generic JVM argument:

```
-Xverbosegclog:${SERVER_LOG_ROOT}/verbosegc.m%d.5/10/13M%S.
%pid.txt,20,10000
```

The verbosegc log file name is `verbosegc.m%d.5/10/13M%S.%pid.txt`. It includes a date/time stamp and the process ID (PID) of the WebSphere Portal Express instance.

The default WebSphere Portal Express installation redirects the verbosegc output to 20 rotating historical log files, each containing 10000 garbage collection (GC) cycles.

For more information about configuring the JVM through WebSphere Application Server, see the IBM WebSphere Application Server information centers at `www.ibm.com/software/webservers/appserv/was/library`.

WebSphere Application Server tracing and log files

Use WebSphere Application Server log files and tracing to troubleshoot problems with WebSphere Portal Express.

WebSphere Application Server has log files and a tracing function; however, whenever possible use the IBM Installation Manager installation logs to determine whether WebSphere Application Server was successfully configured for WebSphere Portal Express and whether WebSphere Portal Express was successfully started on WebSphere Application Server.

Configuration Wizard log files

The configuration wizard generates log files each time you run it. The log file can help you to debug problems.

The trace string for the Configuration Wizard is:
`com.ibm.wplc.config.wizard.*=all`.

The log files for the configuration wizard are located in the following directory:

- Windows: *AppServer_root*`\logs\server1`
- Linux : *AppServer_root*`/logs/server1`
- IBM i: *AppServer_root*`/logs/cw_profile`

Note: On IBM i, the *AppServer_root* refers to the UserData path.

You can view log files from the referenced directory. Log files are also generated that reflect the name of the task ran. These files are generated after the task has completed. They contain task-specific copies of the output in the configwizard log file and can be used to track output for one specific task when multiple tasks are run. The configwizard log file is generated at the task runs and remains in the `PortalServer/logs` directory. Each log file is backed up if a task is run again.

Enabling Virtual Member Manager tracing files

Enable WebSphere Application Server trace facilities to create trace information for Virtual Member Manager.

About this task

Virtual Member Manager uses the WebSphere Application Server trace facilities to create trace information. Complete the following steps to enable the Virtual Member Manager trace output to debug a problem:

Procedure

1. Log on to the WebSphere Integrated Solutions Console.
2. Go to section **Troubleshooting** > **Logs and Traces** > **WebSphere_Portal** > **Diagnostic Trace**.
3. Ensure the **File** radio button under **Trace Output** is selected.
4. Under the **Additional Properties** section, click **Change Log Detail Levels**. Enter the following trace string in the text box:

   ```
   com.ibm.ws.security.*=all:com.ibm.websphere.wim.*=all:
   com.ibm.wsspi.wim.*=all:com.ibm.ws.wim.*=all
   ```
5. Click **OK** and save the changes to the master configuration.
6. Restart WebSphere Portal Express.

Results

The resulting traces of Virtual Member Manager are written to:

- Windows: *wp_profile_root*\logs\WebSphere_Portal\trace.log
- Linux : *wp_profile_root*/logs/WebSphere_Portal/trace.log
- IBM i: *wp_profile_root*/logs/WebSphere_Portal/trace.log

System event logging

The system event logging facility of IBM WebSphere Portal Express enables the recording of information about the operation of WebSphere Portal Express.

Event logs provide administrators with information about important or abnormal events, especially errors that occur during the operation of the product. In addition, event logs gather debugging information that helps IBM support to resolve problems.

WebSphere Portal Express provides two types of logging: logging of messages and logging of debugging messages called traces.

For information about how to use log files and a list of trace logger strings refer to the topic about WebSphere Portal Express logs.

Message logging

Messages for WebSphere Portal Express are logged in the following files:

SystemOut.log
: Contains information that is useful to monitor the health of the WebSphere Portal Express server and all running processes.

System.err
: Contains exception stack trace information that is useful when problem analysis is done.

Locating the log files: Log files for WebSphere Portal Express, including SystemOut.log and System.err are in the following directory: *wp_profile_root*/logs/WebSphere_Portal

Trace logging

WebSphere Portal Express provides the logging of debugging messages called traces. These traces are useful for fixing problems. However, to save system resources, they are turned off by default.

Traces can be set for different durations:

Temporary
: Traces can be set for a temporary period by using the administration portlet **Enable Tracing** or the WebSphere Integrated Solutions Console. To set traces by using the portlet, complete the following steps:

1. Log in as the administrator.
2. Click the **Administration menu** icon. Then, click **Portal Analysis** > **Enable Tracing**. The Enable Tracing portlet displays.

3. Type the required trace string into the field **Append these trace settings:** For example, this string can be `com.ibm.wps.command.credentialvault.*=finest`
4. Click the **Add** icon. **Enable Tracing** updates the **Current trace settings** field.

Note: Restarting WebSphere Portal Express removes traces that were set by using the Enable Tracing Administration portlet.
To disable tracing, use either of the following methods:

- Select the current trace settings under **Current trace settings:** and click the **Remove** icon. For example, the current setting can be `com.ibm.wps.command.credentialvault.*=finest`.
- Type the trace string `*=info` into the field **Append these trace settings:** and click the **Add** icon. This trace string overwrites all settings that are listed under **Current trace settings:** and resets it to the default.

Extended

To enable trace settings for a longer period, that is, for more than one session, switch them on in the WebSphere Application Server configuration. Proceed by the following steps:

1. Access the WebSphere Integrated Solutions Console by using this URL: `http://hostname:`*`port_number`*`/ibm/console`
2. Go to **Servers** > **Server Types** > **WebSphere application servers**.
3. Select the application server.
4. Click **Troubleshooting** > **Change Log Detail Levels**.
5. Specify the required trace settings. For example, this setting can be `com.ibm.wps.command.credentialvault.*=finest`
6. Save your updates.
7. Restart the WebSphere_Portal server.
8. To disable tracing, specify `tracestring: *=info` and restart the WebSphere_Portal server.

Changing the log file name and location

You can change the locations of the log files by configuring them in the WebSphere Integrated Solutions Console. Go to **Troubleshooting** > **Logs and Trace** > **server_name** and select the logger type that you want to change. In the configuration dialog, change the path for the log file as required.

Changing the language used in the log file

By default, information in the log file is written in the language that was used for the WebSphere Portal Express installation. However, because WebSphere Portal Express supports a number of languages, you can choose to have the log file information that is written in a language other than that language used during installation.

To change the language that is used for the log file, edit the file `log.properties`. This file is in the following WebSphere Portal Express directory:

- Linux : *wp_profile_root*`/PortalServer/config/config`
- IBM i: *wp_profile_root*`/PortalServer/config/config`
- Windows: *wp_profile_root*`\PortalServer\config\config`

Add the following line:

`locale=xx`

Where *xx* is the two-letter abbreviation for the locale. For a list of the locale abbreviations that are used with WebSphere Portal Express, refer to the topic about Directory structure and go to the section about Directories for languages. For example, to have log information that is generated in English, you would add the following line:

```
locale=en
```

Reference: Log file format

If the logs are written to the log file of WebSphere Portal Express and not redirected to the logging facility WebSphere Application Server, the log file consists of a sequence log records that are separated by blank lines.

The log records have the following format:

```
timestamp classification classname method threadID
messagecode: logmessage
```

Where:

- The *timestamp* is the time (to the millisecond) when the log record was created.
- The *classification* is one of the following letters:

  E For error messages

  W For warning messages

  I For informational messages

  l For traces (low details)

  m For traces (medium details)

  h For traces (high details)
- The *classname* is the Java class that contains the code that triggered the log event.
- The *method* is the name of the Java method that contains the code that triggered the log event.
- The *messagecode* is a unique identifier for this message, to uniquely identify the specific message and refer to it when you are consulting documentation or support. The message code is only available for error, warning, or informational messages, and not for traces. It consists of:
  - A four-character identifier for the component that defines the message.
  - A four-digit number that identifies the message in the component.
  - A one-letter classification code, which can be E, W, or I.
- The *logmessage* is the actual log message that describes the logged event. Error, warning, and informational messages are translated into the system locale. Trace messages are not translated.
- The *threadID* is the identification of the thread that triggered the log event.

Note:

1. Traces are written only if the specific tracing facility is enabled; all other messages are written unconditionally.

2. The system locale is part of the general globalization features of WebSphere Portal Express and can be configured by using `LocalizerService`. For more information, see the topics about Setting service configuration properties and about the Portal configuration services.

Here is an example of a log record:

```
2011.05.16 13:36:14.449 W com.ibm.wps.services.datastore.DataStoreServiceImpl init 0000003a
DSTO0063W: The transaction isolation level is not set to READ_COMMITTED.

The current value is TRANSACTION_REPEATABLE_READ.
```

Related tasks:

"Starting and stopping servers, deployment managers, and node agents" on page 1216
Various installation and configuration tasks require you to start and stop IBM WebSphere Application Server and the IBM WebSphere Portal Express application servers, deployment managers, and node agents.

"Setting service configuration properties" on page 283
IBM WebSphere Portal Express comprises a framework of services to accommodate the different scenarios that portals need to address. Services are available for both WebSphere Portal Express and IBM Web Content Manager. You can configure some of these services.

Related reference:

"Portal service configuration" on page 289
IBM WebSphere Portal Express comprises a framework of configuration services to accommodate the different scenarios that portals of today need to address. You can configure some of these services.

"Localizer Service" on page 328
The portal Localizer Service provides access to the configured default locale and the system default locale. It also provides a list of supported bidirectional languages. Giving the system default locale is necessary because `Locale.getDefault()` is set to the default.

"Directory structure" on page 3618
The topic shows the naming conventions that are used to denote the location of files on the servers and the types of resources you can find in those directories.

Related information:


WebSphere Application Server 8.5.:Log and trace settings

Web Content Manager tracing files

Enable the use of WebSphere Application Server trace facilities to create trace information for Web Content Manager. This tracing can be enabled either permanently or for just the current WebSphere Portal Express session.

IBM Web Content Manager uses the IBM WebSphere Application Server trace facilities to create trace information. If you need detailed trace output of Web Content Manager to debug a problem, follow these steps:

Permanently enable tracing

Procedure

1. Start WebSphere Application Server.
2. Open the WebSphere Integrated Solutions Console.
3. Go to section **Troubleshooting** > **Logs and Traces** > **WebSphere_Portal** > **Diagnostic Trace**.

4. Make sure that the check box **Enable Trace** is selected.
5. Enter any of the following in the **TraceSpecification** field:
   - com.ibm.workplace.wcm.*
   - com.aptrix.*
   - com.presence.*

   For example, to trace all events, enter the following value:

   ```
   com.ibm.workplace.wcm.*=all:com.aptrix.*=all:com.presence.*=all
   ```
6. Save the changes.
7. Restart IBM WebSphere Portal Express.

Enable tracing just for the current WebSphere Portal Express session

Procedure

1. Click the **Administration menu** icon. Then, click **Portal Analysis** > **Enable Tracing**.
2. Enter any of the following values in the **Append these trace settings** field:
   - com.ibm.workplace.wcm.*
   - com.aptrix.*
   - com.presence.*

   For example, to trace all events, enter the following value:

   ```
   com.ibm.workplace.wcm.*=all:com.aptrix.*=all:com.presence.*=all
   ```

   Here is a list of advanced trace settings:

   com.ibm.workplace.wcm.services.content.*
   This setting enables low level tracing for every item.

   com.ibm.workplace.wcm.domain.transformers.control. Controltype
   You can enable tracing for any of the following control types:
   - HistoryControlTransformer
   - IdentityControlTransformer
   - ProfileControlTransformer
   - SecurityControlTransformer
   - WorkflowControlTransformer

   com.ibm.workplace.wcm.domain.transformers.control.*
   This setting enables the tracing for all control types for all items.

   com.ibm.workplace.wcm.domain.transformers.controllable. Controllabletype
   You can enable tracing for any of the following controllable types:
   - AbstractControllableTransformer
   - AlternateDesignCmpntTransformer
   - AlternateLinkCmpntTransformer
   - ArrayCmpntTransformer
   - AttributeReferenceCmpntTransformer
   - BasePathCmpntTransformer
   - BaseReferenceCmpntTransformer
   - CategoryTransformer
   - CmpntReferenceTransformer
   - CmpntTransformer

- ConfigParamCmpntTransformer
- ContentLinkTransformer
- ContentSpotCmpntTransformer
- ContentTransformer
- ContextPathCmpntTransformer
- ControllableNodeValueTransformer
- ControllableTransformer
- DateCmpntTransformer
- EmailActionTransformer
- ExpireActionTransformer
- ExternalLinkTransformer
- FEDCmpntReferenceTransformer
- FEDCmpntTransformer
- FileResourceCmpntTransformer
- HistoryCmpntTransformer
- HTMLCmpntTransformer
- IDCmpntTransformer
- ImageResourceCmpntTransformer
- IndentCmpntTransformer
- IndexCmpntTransformer
- InlineEditCmpntTransformer
- InlineEditReferenceCmpntTransformer
- JSPCmpntTransformer
- LinkCmpntTransformer
- MenuCmpntTransformer
- NavigatorCmpntTransformer
- NoPrefixBasePathCmpntTransformer
- NoPrefixServletPathCmpntTransformer
- NumericCmpntTransformer
- ObjectSummaryTransformer
- OptionSelectionCmpntTransformer
- PageInfoCmpntTransformer
- PagingCmpntTransformer
- PDMCmpntReferenceTransformer
- PDMCmpntTransformer
- PlaceholderCmpntTransformer
- PlutoSubscriberTransformer
- PlutoSyndicatorTransformer
- PrefixPathCmpntTransformer
- ProfileCmpntTransformer
- PublishActionTransformer
- ResourceCmpntTransformer
- ScheduledMoveActionTransformer
- SearchCmpntTransformer
- SecurityCmpntTransformer

- ServletPathCmpntTransformer
- SiteAreaTransformer
- SiteTransformer
- StyleSheetCmpntReferenceTransformer
- StyleSheetCmpntTransformer
- StyleTransformer
- TaxonomyCmpntTransformer
- TaxonomyTransformer
- TemplateTransformer
- TextCmpntTransformer
- UsernameCmpntTransformer
- UserSelectionCmpntTransformer
- WCMURLCmpntTransformer
- WorkflowActionTransformer
- WorkflowCmpntTransformer
- WorkflowStageTransformer
- WorkflowTransformer

com.ibm.workplace.wcm.domain.transformers.controllable.*
This setting enables the tracing for all controllable types for all items.

Results

The resulting traces of Virtual Member Manager are written here.

- Linux : *wp_profile_root*/logs/WebSphere_Portal/trace.log
- IBM i: *wp_profile_root*/logs/WebSphere_Portal/trace.log
- Windows: *wp_profile_root*\logs\WebSphere_Portal\trace.log

Logging and tracing client side rendering

Portal pages that are rendered in client side aggregation mode differ in their logging behavior from portal pages rendered in server side mode. For client side rendered pages, a considerable amount of code is written in JavaScript that is executed in the corresponding browser JavaScript engine rather than on the server. As a result, the corresponding logging and tracing information is collected in the browser and not on the server side.

"Enabling client side logging and tracing"
To enable client-side logging and tracing for console loggers in client web browsers, modify the custom properties cc.isDebug and cc.traceConfig in the **WP CommonComponentConfigService**. This resource environment provider manages many of the configurable options for the common component.

"Capturing the log statements" on page 3558
Client side log statements are written to the JavaScript console of your browser.

"Enabling module tracing" on page 3559
Enable tracing to debug your module information to improve performance. You can enable a trace string so that debugging is enabled for all users. Or you can set a specific cookie so that debugging is only enabled for that user's cookie.

Enabling client side logging and tracing

To enable client-side logging and tracing for console loggers in client web browsers, modify the custom properties cc.isDebug and cc.traceConfig in the **WP**

CommonComponentConfigService. This resource environment provider manages many of the configurable options for the common component.

Procedure

1. Access the WebSphere Integrated Solutions Console.
2. Go to **Resources** > **Resource environment** > **Resource environment providers** > **WP CommonComponentConfigService**.
3. To view and edit the custom properties for this resource environment provider, click the **Custom Properties** link.
4. Set the value of the property `cc.isDebug` to `true`.
5. Set the value of the property `cc.traceConfig` property to a value that represents a correctly formatted JavaScript array of strings. Each string in the array is the name of the component that you want to trace. You can use the wildcard character asterisk (`*`) for multiple matching. Example:

   ```
   ["com.ibm.mashups.enabler.*","com.ibm.mashups.builder.model.ContextMenu"]
   ```

   This value adds client-side trace-logging for all components in the namespace `com.ibm.mashups.enabler` and the component `com.ibm.mashups.builder.model.ContextMenu`.
6. Save and persist the changes to the master configuration.
7. Restart the portal server.

Example

Examples: To activate all iWidget related logging and tracing, you use the following line:

```
traceConfig: ["com.ibm.mm.iwidget.*"]
```

To set multiple patterns, you separate them by commas like this example:

```
traceConfig: ["com.ibm.mm.iwidget.*", "com.ibm.portal.*", "com.ibm.portal.wps.*"]
```

What to do next

By alternative, you can also achieve the same result by using scripting tools that WebSphere Application Server provides. For more information, refer to the WebSphere Application Server information center under topic *configuring custom properties for resource environment providers by using wsadmin scripting*.

Related information:

Configuring custom properties for resource environment providers using wsadmin scripting

Capturing the log statements

Client side log statements are written to the JavaScript console of your browser.

About this task

There are various ways of accessing the data written to the JavaScript console, depending on the type of browser that you use. For example, in Microsoft Internet Explorer Version 8 you access the console by selecting **Tools** > **Developer Tools**. For FireFox, various plug-ins are available that provide access to the console, for example the Firebug plug-in.

Enabling module tracing

Enable tracing to debug your module information to improve performance. You can enable a trace string so that debugging is enabled for all users. Or you can set a specific cookie so that debugging is only enabled for that user's cookie.

Procedure

1. Enable tracing through trace string.
   a. To debug a module or theme, enable portal tracing with the following trace string.

      ```
      com.ibm.wps.resourceaggregator.CombinerDataSource.RemoteDebug=all
      ```

      This string loads the modules by using separate links and script tags, isolating each one independently. If the module definition defines a debug version, it also loads debug versions of each contribution. Typically, the debug version is an uncompressed version of the <script> tag that contains the same data as the normal version.

      Both using separate links and by using uncompressed <script> tags makes it easier to debug a running WebSphere Portal Express environment from the browser.
2. Enable tracing with cookies.
   a. Open WebSphere Integrated Solutions Console.
   b. Select **Resources** > **Resource Environment** > **Resource Environment Providers**.
   c. Select the **WP ConfigService** resource environment provider.
   d. Click **custom properties**.
   e. Change the **resourceaggregation.client.debug.mode.allowed** entry to `true`.
   f. Save the changes.
   g. Restart the WebSphere Portal Express server.

Results

When a user sets a cookie that is named **com.ibm.portal.resourceaggregator.client.debug.mode** to `true`, debug versions of module contributions are loaded if they are defined. Modules are loaded without using separate links and script tags. All resources are downloaded as a combined unit in that case.

Troubleshooting the Configuration Wizard

The configuration wizard provides tools for troubleshooting and recovering from errors, such as logs, step commands, and reset options.

If you encounter a failure during a configuration, do not go back to the Answer Questions or Customize Values pages. To recover from the failure and continue the configuration:

1. Click **View Results** for the step that failed to determine what caused the failure.
2. Save your selections in case you must reset the configuration process or start over. Click **Download Wizard Selections**.
3. Recovery steps vary based on the cause of the failure.
   - Update your configuration environment and run the step that failed again.
   - Reset the configuration steps and start over.

View Results to determine cause

When you encounter a problem, it is typically recorded in a log file. There are a number of log files. The two most frequently used log files are `ConfigTrace.log` and `SystemOut.log`. On the Configure page, use the **View Results** link to open the `ConfigTrace.log`. To inspect the `ConfigTrace.log`, use the following approach:

1. Search the log file for `BUILD FAILED`.
2. After you find `BUILD FAILED`, search for an error message that occurred before that message.
3. Most errors have a code and message. The code includes an E to indicate error, such as EJPXX1234E. Ideally the message indicates what failed, why, and what you need to do to recover. Some errors, especially from third-party applications, do not have error codes. For these errors, consult the application documentation.

Run Step to run a single step again

Depending on the failure and the action that you took, you might be able to run the step again. You can run the step again if the failure resulted from:

- Not being able to connect to the LDAP or database server because the server was not online. Start the server and when it is online, run the failed step again.
- Network communications instability. When network becomes stable, run the failed step again.

Reset Steps

The **Reset Steps** option forces you to start over. If you must reset all the steps, download your wizard selections to save time. After you click **Reset Steps**, return to the home page by clicking Cancel. Select your configuration options, and click **Upload Saved Selections**. After the XML file uploads, you can correct the settings that led to the error.

You need to reset the steps and start the configuration again if you:

- Provided an incorrect value for the profile path, administrator credentials, or another repeatedly used property.
- Completed manual steps only before the failure occurred.

Resetting the steps does not undo any tasks that the wizard completed before the failure occurred. Steps that you need to take to clean up depend on the configuration steps that the wizard ran before the failure occurred. See individual configuration troubleshooting topics for guidance.

After you correct the values, on the Configure page, click **Skip Step** for any steps that ran that you do not need to run again. If you completed a step in the previous attempt that did not use values that you changed, then you can skip the step.

"Troubleshooting: Database Transfer" on page 3561
Database transfer is part of setting up a stand-alone and cluster server topologies. Learn how to troubleshoot each step in your configuration for your target database.

"Troubleshooting: Enable federated security option" on page 3582
Enabling federated security is part of many environment setups. If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

"Troubleshooting: Create a deployment manager" on page 3587
Create a deployment manager for clustered environments. If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

"Troubleshooting: Create a cluster option" on page 3589
Creating a cluster is part of setting up a clustered environment.If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

"Troubleshooting: Create an additional cluster node" on page 3591
Adding a node to a cluster is part of setting up a clustered environment. If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

"Troubleshooting: Create a WebSphere Portal profile" on page 3594
View troubleshooting information for creating a WebSphere Portal Express profile.

"Troubleshooting: Remove a WebSphere Portal profile" on page 3598
View troubleshooting information for creating a WebSphere Portal Express profile.

"Troubleshooting: Migrate a stand-alone server" on page 3599
If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

"Troubleshooting: Migrate the deployment manager profile for a cluster environment" on page 3605
If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

"Troubleshooting: Migrate node profiles for a cluster environment" on page 3608
If you encounter a failure during the migration of the node profiles for a cluster environment, learn how to correct the issue and recover from the failure.

"Troubleshooting: Upgrade node profiles for a cluster environment" on page 3612
If you encounter a failure while upgrading the node profiles for a cluster environment, learn how to correct the issue and recover from the failure.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

Troubleshooting: Database Transfer

Database transfer is part of setting up a stand-alone and cluster server topologies. Learn how to troubleshoot each step in your configuration for your target database.

"DB2: Troubleshooting Database Transfer" on page 3562
If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

"IBM DB2 for i: Troubleshooting Database Transfer" on page 3567
If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

"DB2 for z/OS: Troubleshooting Database Transfer" on page 3570
If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

"SQL Server: Troubleshooting Database Transfer" on page 3574
If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

"Oracle: Troubleshooting Database Transfer" on page 3578
If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

DB2: Troubleshooting Database Transfer

If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

Each potential step in the configuration is included. Because the steps vary depending on your selections, the steps are not numbered. Find the step that failed to learn more about correcting and recovering from the failure. If you need to change a value that you entered in the wizard, then you must run the configuration again.

Tip: If you must go through the wizard again, download the wizard selections that you made to save time. Then, cancel the configuration. Start the process over and upload your saved selections. Correct or enter values for the parameters that caused the failure.

Back up the properties files that the wizard uses during the configuration

About this task

During this step, the wizard attempts to back up the `wkplc.properties`, `wkplc_dbdomain.properties`, and `wkplc_dbtype.properties` files.

Table 506. Appropriate actions for step: Back up the properties files that the wizard uses during the configuration

Actions	Notes
Run the step again	You can run the step again without causing any harm. Alternatively, you can manually back up the properties files instead of running the step again.
Skip the step	If you already backed up your properties files before you started this configuration, you can skip this step. If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Manual Step: Create the database users and groups

About this task

During this step, you create the database configuration users and groups on the database server.

If you need a runtime user for day-to-day operations, you must create the runtime database users and groups on the database server.

Use the database user IDs and group names that are entered in the wizard when you create the database users and groups.

Table 507. Appropriate actions for step: Manual Step: Create the database users and groups

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm. If you realize later in the configuration that the database users and groups were not created correctly, you can repeat these instructions. For example, you might encounter a failure in the setup database step if the database users and groups are created incorrectly. You can repeat these instructions to correct the issue and then perform the setup database step again.
Skip the step	If you successfully created the database users and groups before you started this configuration, you can skip this step.If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Create your databases
About this task

Table 508. Appropriate actions for step: Create your databases

Actions	Notes
Run the step again	The type of failure that you encounter determines whether you can run the step again. Use the ERRORCODE and SQLSTATE associated with the error message in the `ConfigTrace.log` to determine the reason for the failure. You can run the step again if the reason for the failure does not affect any values that you entered in the wizard. For example, if you forgot to copy the JDBC JAR files from your database to the portal server, you can copy the files to the portal server. Then, you can run the step again. If you need to update wizard values to correct the issue, do not run the step again. You must create new scripts in the wizard. Upload your saved selections, update your values, and create scripts. Examples of issues that result in you updating your values include entering the wrong port number or entering an incorrect host name into the wizard.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required. If the step fails because you entered incorrect values in the wizard, remove the database manually, enter correct values to create your scripts, then run the configuration again.

Manual Step: Download the script and run it on the database server to create your database

About this task

During this step, you download a script to create your database.

Table 509. Appropriate actions for step: Manual Step: Download the script and run it on the database server to create your database

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm. In this step, you download a script to run. The type of failure you encounter determines whether you can run the script again. Use the ERRORCODE and SQLSTATE associated with the error message in the `ConfigTrace.log` to determine the reason for the failure. You can run the step again if the reason for the failure does not affect any values that you entered in the wizard. For example, if you forgot to copy the JDBC JAR files from your database to the portal server, you can copy the files to the portal server. Then, you can run the step again. If you need to update wizard values to correct the issue, do not run the step again. You must create new scripts in the wizard. Upload your saved selections, update your values, and create scripts. Examples of issues that result in you updating your values include entering the wrong port number or entering an incorrect host name into the wizard.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	Not applicable

Set up your database

About this task

Use the ERRORCODE and SQLSTATE associated with the error message in the `ConfigTrace.log` to determine the source of the problem.

Table 510. Appropriate actions for step: Set up your database

Actions	Notes
Run the step again	The type of failure that you encounter determines whether you can run the step again. You can run the step again if the issue for the failure does not affect any values that you entered in the wizard. If you need to update values, you must create new scripts in the wizard. Upload your saved selections, change the affected values, and create new scripts to run. For example, if you entered the wrong path to your database library, you must upload your saved selections, correct the path, and then configure your system with new scripts.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Manual Step: Download the script and run it on the database server to set up your database

About this task

During this step, you set permissions on your database.

Table 511. Appropriate actions for step: Manual Step: Download the script and run it on the database server to set up your database

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm. In this step, you download a script to run. The type of failure you encounter determines whether you can run the script again. Use the ERRORCODE and SQLSTATE associated with the error message in the `ConfigTrace.log` to determine the reason for the failure. You can run the downloaded script again if the issue for the failure does not affect any values that you entered in the wizard. For example, if you created database users and groups incorrectly on your database server, you can correct the issue and run the step again. If you need to update values, you must create new scripts in the wizard. Upload your saved selections, change the affected values, and create new scripts to run.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Manual Step: Set up JCR collation for correct language locale order

About this task

Table 512. Appropriate actions for step: Manual Step: Set up JCR collation for correct language locale order

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	You must remove the database and create the database again.

Manual Step: Restart the DB2 server

About this task

During this step, you set global database default values that are required for the database that you created. These global values do not take effect until you restart your database server.

Table 513. Appropriate actions for step: Manual Step: Restart the DB2 server

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again. If you already restarted your database server, you can skip this step.
Clean up step	None required

Validate the database connection and environment

About this task

During this step, you are validating that the database is still available and that you can successfully connect to the database.

Table 514. Appropriate actions for step: Validate the database connection and environment

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Stop the portal server
About this task

Table 515. Appropriate actions for step: Stop the portal server

Actions	Notes
Run the step again	You can run the step again without causing any harm.
Skip the step	If you already stopped the portal server, you can skip this step.
Clean up step	None required

Transfer the database
About this task

Table 516. Appropriate actions for step: Transfer the database

Actions	Notes
Run the step again	The type of failure that you encounter determines whether you can run the step again. Use the ERRORCODE and SQLSTATE associated with the error message in the `ConfigTrace.log` to determine the reason for the failure. You might have a portal failure that does not have a SQLSTATE error code that is associated with the error message. You can run the step again if the issue for the failure does not affect any values that you entered in the wizard. If you need to update values in the wizard, you must create new scripts in the wizard. Upload your saved selections, change the affected values, and create new scripts to run. You do not need to run previous successful steps.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Grant privileges to the database runtime users.
About this task

Table 517. Appropriate actions for step: Grant privileges to the database runtime users.

Actions	Notes
Run the step again	The type of failure that you encounter determines whether you can run the step again. You can run the step again if the issue for the failure does not affect any values that you entered in the wizard. If you need to update values in the wizard, you must create new scripts in the wizard. Upload your saved selections, change the affected values, and create new scripts to run. You do not need to run previous successful steps.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Manual Step: Download the script and run it on the database server to grant privileges to the runtime user

About this task

Table 518. Appropriate actions for step: Manual Step: Download the script and run it on the database server to grant privileges to the runtime user

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm. In this step, you download a script to run. The type of failure you encounter determines whether you can run the script again. You can run the downloaded script again if the issue for the failure does not affect any values that you entered in the wizard. For example, if you created runtime users and groups incorrectly on your database server, you can correct the issue and run the script again. If you need to update values in the wizard, you must create new scripts in the wizard. Upload your saved selections, change the affected values, and create new scripts to run. You do not need to run previous successful steps.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Configure the JCR domain to support large files

About this task

Table 519. Appropriate actions for step: Configure the JCR domain to support large files

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Start the portal server

About this task

Table 520. Appropriate actions for step: Start the portal server

Actions	Notes
Run the step again	You can run the step again without causing any harm
Skip the step	If you already started the portal server, you can skip this step.
Clean up step	None required

IBM DB2 for i: Troubleshooting Database Transfer

If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

Each potential step in the configuration is included. Because the steps vary depending on your selections, the steps are not numbered. Find the step that failed to learn more about correcting and recovering from the failure. If you need to change a value that you entered in the wizard, then you must run the configuration again.

Tip: If you must go through the wizard again, download the wizard selections that you made to save time. Then, cancel the configuration. Start the process over and upload your saved selections. Correct or enter values for the parameters that caused the failure.

Back up the properties files that the wizard uses during the configuration

About this task

During this step, the wizard attempts to back up the `wkplc.properties`, `wkplc_dbdomain.properties`, and `wkplc_dbtype.properties` files.

Table 521. Appropriate actions for step: Back up the properties files that the wizard uses during the configuration

Actions	Notes
Run the step again	You can run the step again without causing any harm. Alternatively, you can manually back up the properties files instead of running the step again.
Skip the step	If you already backed up your properties files before you started this configuration, you can skip this step. If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Manual Step: Create the database user profile on IBM DB2 for i

About this task

During this step, you create the database user profile on the database server. Use the database user ID that you entered in the database configuration user field in the Configuration Wizard for the database user profile.

Table 522. Appropriate actions for step: Manual Step: Create the database user profile on IBM DB2 for i

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm. If you realize in the configuration that the database user profile was not created correctly, you can repeat these instructions.
Skip the step	If you successfully created the database user profile before you started this configuration, you can skip this step.If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Manual Step: Create the database runtime users and groups

About this task

During this step, you create the database runtime users on the database server. If you create database runtime users, you must also create database runtime groups.

Table 523. Appropriate actions for step: Manual Step: Create the database runtime users and groups

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm. If you realize later in the configuration that the database runtime user was not created correctly, you can repeat these instructions. For example, you might encounter a failure when you set up your database if the database runtime user is created incorrectly. You can repeat these instructions to correct the issue. Then, you can repeat the setup database step again.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Create your databases
About this task

Table 524. Appropriate actions for step: Create your databases

Actions	Notes
Run the step again	The type of failure that you encounter determines whether you can run the step again. Use the ERRORCODE and SQLSTATE associated with the error message in the `ConfigTrace.log` to determine the reason for the failure. You can run the step again if the reason for the failure does not affect any values that you entered in the wizard. For example, if you forgot to copy the JDBC JAR files from your database to the portal server, you can copy the files to the portal server. Then, you can run the step again. If you need to update wizard values to correct the issue, do not run the step again. You must create new scripts in the wizard. Upload your saved selections, update your values, and create scripts. Examples of issues that result in you updating your values include entering the wrong port number or entering an incorrect host name into the wizard.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required. If the step fails because you entered incorrect values in the wizard, remove the database manually, enter correct values to create your scripts, then run the configuration again.

Validate the database connection and environment
About this task

During this step, you are validating that the database is still available and that you can successfully connect to the database.

Table 525. Appropriate actions for step: Validate the database connection and environment

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Stop the portal server
About this task

Table 526. Appropriate actions for step: Stop the portal server

Actions	Notes
Run the step again	You can run the step again without causing any harm.
Skip the step	If you already stopped the portal server, you can skip this step.
Clean up step	None required

Transfer the database
About this task

Table 527. Appropriate actions for step: Transfer the database

Actions	Notes
Run the step again	The type of failure that you encounter determines whether you can run the step again. Use the ERRORCODE and SQLSTATE associated with the error message in the `ConfigTrace.log` to determine the reason for the failure. You might have a portal failure that does not have a SQLSTATE error code that is associated with the error message. You can run the step again if the issue for the failure does not affect any values that you entered in the wizard. If you need to update values in the wizard, you must create new scripts in the wizard. Upload your saved selections, change the affected values, and create new scripts to run. You do not need to run previous successful steps.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Grant privileges to the database runtime users.
About this task

Table 528. Appropriate actions for step: Grant privileges to the database runtime users.

Actions	Notes
Run the step again	The type of failure that you encounter determines whether you can run the step again. You can run the step again if the issue for the failure does not affect any values that you entered in the wizard. If you need to update values in the wizard, you must create new scripts in the wizard. Upload your saved selections, change the affected values, and create new scripts to run. You do not need to run previous successful steps.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Start the portal server
About this task

Table 529. Appropriate actions for step: Start the portal server

Actions	Notes
Run the step again	You can run the step again without causing any harm
Skip the step	If you already started the portal server, you can skip this step.
Clean up step	None required

DB2 for z/OS: Troubleshooting Database Transfer

If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

Each potential step in the configuration is included. Because the steps vary depending on your selections, the steps are not numbered. Find the step that failed to learn more about correcting and recovering from the failure. If you need to change a value that you entered in the wizard, then you must run the configuration again.

Tip: If you must go through the wizard again, download the wizard selections that you made to save time. Then, cancel the configuration. Start the process over and upload your saved selections. Correct or enter values for the parameters that caused the failure.

Back up the properties files that the wizard uses during the configuration

About this task

During this step, the wizard attempts to back up the `wkplc.properties`, `wkplc_dbdomain.properties`, and `wkplc_dbtype.properties` files.

Table 530. Appropriate actions for step: Back up the properties files that the wizard uses during the configuration

Actions	Notes
Run the step again	You can run the step again without causing any harm. Alternatively, you can manually back up the properties files instead of running the step again.
Skip the step	If you already backed up your properties files before you started this configuration, you can skip this step. If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Manual Step: Create the database configuration users on DB2 for z/OS

About this task

During this step, you download RACF commands. Review the RACF commands with your RACF administrator. If you encounter a failure, you might need to edit the RACF commands before you run the RACF commands again.

Table 531. Appropriate actions for step: Manual Step: Create the database configuration users on DB2 for z/OS

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again. If you have database IDs that are already defined, review the RACF commands before you skip this step. You might need to edit the RACF commands. For example, the RACF administrator might need to remove the ADDUSER commands if you already have database user IDs that are defined in your RACF.
Clean up step	None required

Manual Step: Download the script and view instructions to delete existing databases

About this task

Table 532. Appropriate actions for step: Manual Step: Download the script and view instructions to delete existing databases

Actions	Notes
Run the step again	You can run this step only under certain conditions. • If you have previously run the database transfer option, you need to delete the databases that you created before you rerun the create database and transfer database steps. • You can run the commands in this step only once when you use the database transfer option. If you run this step multiple times, you might delete databases that must exist for your configuration.
Skip the step	If you have never run the transfer database step, you can skip this step.
Clean up step	None required

Manual Step: Download the script and view instructions to create your databases

About this task

Table 533. Appropriate actions for step: Manual Step: Download the script and view instructions to create your databases

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm. Use the SQLSTATE associated with the error message to determine the source of the problem.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	If you need to update wizard values to correct the issue, do not run the commands in this step again. You must create new scripts in the wizard. Upload your saved selections, update your values, and create scripts. In the next configuration attempt, you must delete existing databases before you create databases. Do not skip the following step: Manual Step: Download the script and view instructions to delete existing databases

Validate the database connection and environment

About this task

During this step, you are validating that the database is still available and that you can successfully connect to the database.

Table 534. Appropriate actions for step: Validate the database connection and environment

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Stop the portal server

About this task

Table 535. Appropriate actions for step: Stop the portal server

Actions	Notes
Run the step again	You can run the step again without causing any harm.
Skip the step	If you already stopped the portal server, you can skip this step.
Clean up step	None required

Transfer the database

About this task

Table 536. Appropriate actions for step: Transfer the database

Actions	Notes
Run the step again	The type of failure that you encounter determines whether you can run the step again. Use the ERRORCODE and SQLSTATE associated with the error message in the `ConfigTrace.log` to determine the reason for the failure. You might have a portal failure that does not have a SQLSTATE error code that is associated with the error message. You can run the step again if the issue for the failure does not affect any values that you entered in the wizard. If you need to update values in the wizard, you must create new scripts in the wizard. Upload your saved selections, change the affected values, and create new scripts to run. You do not need to run previous successful steps.

Table 536. Appropriate actions for step: Transfer the database (continued)

Actions	Notes
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	If you need to update wizard values to correct the issue, do not run the commands in this step again. You must create new scripts in the wizard. Upload your saved selections, update your values, and create scripts. In the next configuration attempt, you must delete existing databases before you create databases. Do not skip the following step: Manual Step: Download the script and view instructions to delete existing databases.

Grant privileges to the database runtime users.

About this task

Table 537. Appropriate actions for step: Grant privileges to the database runtime users.

Actions	Notes
Run the step again	The type of failure that you encounter determines whether you can run the step again. You can run the step again if the issue for the failure does not affect any values that you entered in the wizard. If you need to update values in the wizard, you must create new scripts in the wizard. Upload your saved selections, change the affected values, and create new scripts to run. You do not need to run previous successful steps.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Connect to your databases

About this task

Table 538. Appropriate actions for step: Connect to your databases

Actions	Notes
Run the step again	The type of failure that you encounter determines whether you can run the step again. Use the ERRORCODE and SQLSTATE associated with the error message in the `ConfigTrace.log` to determine the reason for the failure. You can run the step again if the issue for the failure does not affect any values that you entered in the wizard. If you need to update values in the wizard, you must create new scripts in the wizard. Upload your saved selections, change the affected values, and create new scripts to run. You do not need to run previous successful steps.
Skip the step	If you need to run the configuration again, you must repeat this step if you need to run the transfer database step again. If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again. If this step is successful, you can skip it if you encounter a failure in a later step.
Clean up step	None required

Manual Step: Download the script and view instructions to reset the check pending status on portal table spaces

About this task

Table 539. Appropriate actions for step: Manual Step: Download the script and view instructions to reset the check pending status on portal table spaces

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Reset the web content manager event log
About this task

Table 540. Appropriate actions for step: Reset the web content manager event log

Actions	Notes
Run the step again	You can run this step repeatedly without causing harm.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Start the portal server
About this task

Table 541. Appropriate actions for step: Start the portal server

Actions	Notes
Run the step again	You can run the step again without causing any harm
Skip the step	If you already started the portal server, you can skip this step.
Clean up step	None required

SQL Server: Troubleshooting Database Transfer

If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

Each potential step in the configuration is included. Because the steps vary depending on your selections, the steps are not numbered. Find the step that failed to learn more about correcting and recovering from the failure. If you need to change a value that you entered in the wizard, then you must run the configuration again.

Tip: If you must go through the wizard again, download the wizard selections that you made to save time. Then, cancel the configuration. Start the process over and upload your saved selections. Correct or enter values for the parameters that caused the failure.

Back up the properties files that the wizard uses during the configuration
About this task

During this step, the wizard attempts to back up the `wkplc.properties`, `wkplc_dbdomain.properties`, and `wkplc_dbtype.properties` files.

Table 542. Appropriate actions for step: Back up the properties files that the wizard uses during the configuration

Actions	Notes
Run the step again	You can run the step again without causing any harm. Alternatively, you can manually back up the properties files instead of running the step again.
Skip the step	If you already backed up your properties files before you started this configuration, you can skip this step. If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Create your databases
About this task

Table 543. Appropriate actions for step: Create your databases

Actions	Notes
Run the step again	The type of failure that you encounter determines whether you can run the step again. Use the ERRORCODE and SQLSTATE associated with the error message in the `ConfigTrace.log` to determine the reason for the failure. You can run the step again if the reason for the failure does not affect any values that you entered in the wizard. For example, if you forgot to copy the JDBC JAR files from your database to the portal server, you can copy the files to the portal server. Then, you can run the step again. If you need to update wizard values to correct the issue, do not run the step again. You must create new scripts in the wizard. Upload your saved selections, update your values, and create scripts. Examples of issues that result in you updating your values include entering the wrong port number or entering an incorrect host name into the wizard.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required. If the step fails because you entered incorrect values in the wizard, remove the database manually, enter correct values to create your scripts, then run the configuration again.

Manual Step: Download the script and run it on the database server to create your database
About this task

Table 544. Appropriate actions for step: Manual Step: Download the script and run it on the database server to create your database

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm. In this step, you download a script to run. The type of failure you encounter determines whether you can run the script again. Use the ERRORCODE and SQLSTATE associated with the error message in the `ConfigTrace.log` to determine the reason for the failure. You can run the step again if the reason for the failure does not affect any values that you entered in the wizard. For example, if you forgot to copy the JDBC JAR files from your database to the portal server, you can copy the files to the portal server. Then, you can run the step again. If you need to update wizard values to correct the issue, do not run the step again. You must create new scripts in the wizard. Upload your saved selections, update your values, and create scripts. Examples of issues that result in you updating your values include entering the wrong port number or entering an incorrect host name into the wizard.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Set up your database
About this task

Use the ERRORCODE and SQLSTATE associated with the error message in the `ConfigTrace.log` to determine the source of the problem.

Table 545. Appropriate actions for step: Set up your database

Actions	Notes
Run the step again	The type of failure that you encounter determines whether you can run the step again. You can run the step again if the issue for the failure does not affect any values that you entered in the wizard. If you need to update values, you must create new scripts in the wizard. Upload your saved selections, change the affected values, and create new scripts to run. For example, if you entered the wrong path to your database library, you must upload your saved selections, correct the path, and then configure your system with new scripts.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Manual Step: Download the script and run it on the database server to set up your database

About this task

During this step, you set permissions on your database.

Table 546. Appropriate actions for step:

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm. In this step, you download a script to run. The type of failure you encounter determines whether you can run the script again. Use the ERRORCODE and SQLSTATE associated with the error message in the `ConfigTrace.log` to determine the reason for the failure. You can run the downloaded script again if the issue for the failure does not affect any values that you entered in the wizard. For example, if you created database users and groups incorrectly on your database server, you can correct the issue and run the step again. If you need to update values, you must create new scripts in the wizard. Upload your saved selections, change the affected values, and create new scripts to run.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Validate the database connection and environment

About this task

During this step, you are validating that the database is still available and that you can successfully connect to the database.

Table 547. Appropriate actions for step: Validate the database connection and environment

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Stop the portal server

About this task

Table 548. Appropriate actions for step: Stop the portal server

Actions	Notes
Run the step again	You can run the step again without causing any harm.
Skip the step	If you already stopped the portal server, you can skip this step.
Clean up step	None required

Transfer the database
About this task

Table 549. Appropriate actions for step: Transfer the database

Actions	Notes
Run the step again	The type of failure that you encounter determines whether you can run the step again. Use the ERRORCODE and SQLSTATE associated with the error message in the `ConfigTrace.log` to determine the reason for the failure. You might have a portal failure that does not have a SQLSTATE error code that is associated with the error message. You can run the step again if the issue for the failure does not affect any values that you entered in the wizard. If you need to update values in the wizard, you must create new scripts in the wizard. Upload your saved selections, change the affected values, and create new scripts to run. You do not need to run previous successful steps.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Grant privileges to the database runtime users.
About this task

Table 550. Appropriate actions for step: Grant privileges to the database runtime users.

Actions	Notes
Run the step again	The type of failure that you encounter determines whether you can run the step again. You can run the step again if the issue for the failure does not affect any values that you entered in the wizard. If you need to update values in the wizard, you must create new scripts in the wizard. Upload your saved selections, change the affected values, and create new scripts to run. You do not need to run previous successful steps.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Manual Step: Download the script and run it on the database server to grant privileges to database runtime users
About this task

Table 551. Appropriate actions for step: Manual Step: Download the script and run it on the database server to grant privileges to database runtime users

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm. In this step, you download a script to run. The type of failure you encounter determines whether you can run the script again. You can run the downloaded script again if the issue for the failure does not affect any values that you entered in the wizard. For example, if you created runtime users and groups incorrectly on your database server, you can correct the issue and run the script again. If you need to update values in the wizard, you must create new scripts in the wizard. Upload your saved selections, change the affected values, and create new scripts to run. You do not need to run previous successful steps.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Start the portal server
About this task

Table 552. Appropriate actions for step: Start the portal server

Actions	Notes
Run the step again	You can run the step again without causing any harm
Skip the step	If you already started the portal server, you can skip this step.
Clean up step	None required

Oracle: Troubleshooting Database Transfer
If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

Each potential step in the configuration is included. Because the steps vary depending on your selections, the steps are not numbered. Find the step that failed to learn more about correcting and recovering from the failure. If you need to change a value that you entered in the wizard, then you must run the configuration again.

Tip: If you must go through the wizard again, download the wizard selections that you made to save time. Then, cancel the configuration. Start the process over and upload your saved selections. Correct or enter values for the parameters that caused the failure.

Back up the properties files that the wizard uses during the configuration
About this task

During this step, the wizard attempts to back up the `wkplc.properties`, `wkplc_dbdomain.properties`, and `wkplc_dbtype.properties` files.

Table 553. Appropriate actions for step: Back up the properties files that the wizard uses during the configuration

Actions	Notes
Run the step again	You can run the step again without causing any harm. Alternatively, you can manually back up the properties files instead of running the step again.
Skip the step	If you already backed up your properties files before you started this configuration, you can skip this step. If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Manual Step: Create your Oracle database
About this task

Table 554. Appropriate actions for step: Manual Step: Create your Oracle database

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm. The type of failure that you encounter determines whether you can run the step again. You can repeat the instructions again if the issue for the failure does not affect the values that you entered in the wizard. For example, if you forgot to copy the JDBC JAR files from your database to the portal server, you can copy the files to the portal server. Then, repeat the step without harm. If you need to update wizard values to correct the issue, you must create new scripts in the wizard. Upload your saved selections, update your values, and create scripts. Examples of issues that result in you updating your values include entering the wrong port number or entering an incorrect host name into the wizard.

Table 554. Appropriate actions for step: Manual Step: Create your Oracle database (continued)

Actions	Notes
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Manual Step: Create the data directory, `data`, and the index directory, `index`, on your database server

About this task

Table 555. Appropriate actions for step: Manual Step: Create the data directory, `data`, and the index directory, `index`, on your database server

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm.
Skip the step	If you already created these directories, skip this step. If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Set up your database

About this task

Use the ERRORCODE and SQLSTATE associated with the error message in the `ConfigTrace.log` to determine the source of the problem.

Table 556. Appropriate actions for step: Set up your database

Actions	Notes
Run the step again	The type of failure that you encounter determines whether you can run the step again. You can run the step again if the issue for the failure does not affect any values that you entered in the wizard. If you need to update values, you must create new scripts in the wizard. Upload your saved selections, change the affected values, and create new scripts to run. For example, if you entered the wrong path to your database library, you must upload your saved selections, correct the path, and then configure your system with new scripts.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Manual Step: Download the script and run it on the database server to set up your database

About this task

During this step, you set permissions on your database.

Table 557. Appropriate actions for step: Manual Step: Download the script and run it on the database server to set up your database

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm. In this step, you download a script to run. The type of failure you encounter determines whether you can run the script again. Use the ERRORCODE and SQLSTATE associated with the error message in the `ConfigTrace.log` to determine the reason for the failure. You can run the downloaded script again if the issue for the failure does not affect any values that you entered in the wizard. If you need to update values, you must create new scripts in the wizard. Upload your saved selections, change the affected values, and create new scripts to run. For example, if you entered the wrong path to your database library, you must upload your saved selections, correct the path, and then configure your system with new scripts.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Validate the database connection and environment

About this task

During this step, you are validating that the database is still available and that you can successfully connect to the database.

Table 558. Appropriate actions for step: Validate the database connection and environment

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Stop the portal server

About this task

Table 559. Appropriate actions for step: Stop the portal server

Actions	Notes
Run the step again	You can run the step again without causing any harm.
Skip the step	If you already stopped the portal server, you can skip this step.
Clean up step	None required

Transfer the database
About this task

Table 560. Appropriate actions for step: Transfer the database

Actions	Notes
Run the step again	The type of failure that you encounter determines whether you can run the step again. Use the ERRORCODE and SQLSTATE associated with the error message in the `ConfigTrace.log` to determine the reason for the failure. You might have a portal failure that does not have a SQLSTATE error code that is associated with the error message. You can run the step again if the issue for the failure does not affect any values that you entered in the wizard. For example, if your data and index directories are not created in the correct location, this step might fail. (Automatic Storage Management Users only): After you run the setup database script, you must manually create JCR table spaces to prevent a database transfer failure . Go to "Oracle: Creating JCR table spaces (Automatic Storage Management)" on page 554 to perform additional manual instructions. After you perform these instructions, run the transfer database step again. (Automatic Storage Management Users only): If you manually downloaded a script to set up your database, you must edit the script for your Automatic Storage Management environment to prevent a database transfer failure. After you run the edited script, you can run the transfer database step again. If you need to update values in the wizard, you must create new scripts in the wizard. Upload your saved selections, change the affected values, and create new scripts to run. You do not need to run previous successful steps.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Grant privileges to the database runtime users.
About this task

Table 561. Appropriate actions for step: Grant privileges to the database runtime users.

Actions	Notes
Run the step again	The type of failure that you encounter determines whether you can run the step again. You can run the step again if the issue for the failure does not affect any values that you entered in the wizard. If you need to update values in the wizard, you must create new scripts in the wizard. Upload your saved selections, change the affected values, and create new scripts to run. You do not need to run previous successful steps.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Manual Step: Download the script and run it to grant the database runtime user the appropriate privileges to work with database tables

About this task

Table 562. Appropriate actions for step: Manual Step: Download the script and run it to grant the database runtime user the appropriate privileges to work with database tables

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm. In this step, you download a script to run. The type of failure you encounter determines whether you can run the script again. You can run the downloaded script again if the issue for the failure does not affect any values that you entered in the wizard. For example, if you created runtime users and groups incorrectly on your database server, you can correct the issue and run the script again. If you need to update values in the wizard, you must create new scripts in the wizard. Upload your saved selections, change the affected values, and create new scripts to run. You do not need to run previous successful steps.
Skip the step	If this step is successful, you can skip it if you encounter a failure in a later step and need to run the configuration again.
Clean up step	None required

Manual Step: Improve database response time for your database that contains the JCR domain

About this task

Table 563. Appropriate actions for step: Manual Step: Improve database response time for your database that contains the JCR domain

Actions	Notes
Run the step again	You cannot rerun a manual step, but you can perform the instructions for the step again without harm.
Skip the step	Not applicable.
Clean up step	None required

Start the portal server

About this task

Table 564. Appropriate actions for step: Start the portal server

Actions	Notes
Run the step again	You can run the step again without causing any harm
Skip the step	If you already started the portal server, you can skip this step.
Clean up step	None required

Troubleshooting: Enable federated security option

Enabling federated security is part of many environment setups. If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

Each potential step in the configuration is included. Because the steps vary depending on your selections, the steps are not numbered. Find the step that failed to learn more about correcting and recovering from the failure. If you need to change a value that you entered in the wizard, then you must run the configuration again.

Tip: If you must go through the wizard again, download the wizard selections that you made to save time. Then, cancel the configuration. Start the process over and

upload your saved selections. Correct or enter values for the parameters that caused the failure.
Attention: The Enable Federated Security option modifies the `wimconfig.xml` file. Make a backup copy of this file before you run any of the configuration tasks.

wp_profile_root`/config/cells/`*CellName*`/wim/config/wimconfig.xml`

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

Manual Step: Retrieve the SSL certificate from the SSL port

About this task

Table 565. Appropriate actions for step: Manual Step: Retrieve the SSL certificate from the SSL port

Actions	Notes
Run the step again	Not applicable
Skip the step	Yes, if you completed this manual step successfully, you can skip the step in subsequent configuration attempts
Clean up step	None required

Create a backup of the WebSphere Portal Express profile before modifying cell security

CF07

About this task

Actions	Notes
Run step again	You can run the step repeatedly without causing any harm.
Skip step	If this step is successful, you can skip it if you run the configuration process again.
Clean up step	None required

Validate your LDAP server settings

During this step, the wizard attempts to connect to your LDAP server and authenticate by using the provided credentials and LDAP information.

About this task

Table 566. Appropriate actions for step: Validate your LDAP server settings

Actions	Notes
Run step again	You can run the step repeatedly without causing any harm.
Skip step	If this step is successful, you can skip it if you run the configuration process again.
Clean up step	None required

Procedure

Verify that the values used to connect with the LDAP were entered correctly. Click **View Step Command** to see which values are used.

Add an LDAP user registry to the default federated repository

During this step, the wizard attempts to add your LDAP to the federated repository. This step uses the same parameters as the step that validates the LDAP server settings.

About this task

Table 567. Appropriate actions for step: Add an LDAP user registry to the default federated repository

Actions	Notes
Run step again	You can run the step repeatedly without causing any harm.
Skip step	If this step is successful, you can skip it if you run the configuration process again.
Clean up step	Complete the following steps from the WebSphere Integrated Solutions Console to remove the configured repository: 1. Go to **Security** > **Global Security** > **Configure**. 2. Remove the repository from the realm. 3. Go to **Manage repositories** and delete the repository configuration.

Register the WebSphere Application Server scheduler tasks

About this task

Table 568. Appropriate actions for step: Register the WebSphere Application Server scheduler tasks

Actions	Notes
Run step again	You can run this step again after you clean up the issue.
Skip step	If this step is successful, you can skip it if you run the configuration process again.
Clean up step	Log in to the WebSphere Integrated Solutions Console. Go to **Resources** > **Schedulers** and delete the **WPSTaskScheduler**. If this task fails because of the administrator ID, change the **federated.ldap.bindDN** and optionally the **newAdminId** value. These values must be unique. Then, rerun this task. If this action does not resolve the issue, run the **wp-change-portal-admin-user** and **wp-change-was-admin-user** tasks. These tasks change the **PortalAdminId** and **WasUserId** so that the file system administrators are different from the LDAP users.

Replace the file-based WebSphere Portal and WebSphere Application Server users and groups with users and groups from your LDAP server

During this step, the wizard attempts to configure the portal to use the administrative user and user group that is stored in your LDAP server. The administrative ID and group must exist in your LDAP server. If the ID and group do not exist, create them and try the step again.

About this task

Table 569. Appropriate actions for step: Replace the file-based WebSphere Portal and WebSphere Application Server users and groups with users and groups from your LDAP server

Actions	Notes
Run step again	You can run the step repeatedly without causing any harm.
Skip step	If this step is successful, you can skip it if you run the configuration process again.

Table 569. Appropriate actions for step: Replace the file-based WebSphere Portal and WebSphere Application Server users and groups with users and groups from your LDAP server (continued)

Actions	Notes
Clean up step	You can log in to the WebSphere Integrated Solutions Console. However, the portal administrative user does not work as expected. You do not need to deactivate security with the file-based repository. If the WebSphere Application Server administrative user is not functional, it is likely that the WebSphere Integrated Solutions Console is not accessible. If you cannot log in to the WebSphere Integrated Solutions Console, disable security in the `security.xml` file in the *wp_profile_root*`/config/cells/`*`cellname`* directory. Restart WebSphere Application Server and log in. Then, complete the following steps: 1. Go to **Users and Groups** > **Administrative user roles**. 2. Validate the current administrative user ID or set a new user. 3. Go to **Resources** > **Resource Environment** > **Resource Environment Providers** > **WP AccessControlDataManagementService** > **Custom properties**. 4. Validate the values for the administrative users and groups of the different domains. If necessary, update the values to a valid user. **Valid users:** To find valid users, go to **Users and Groups** > **Manage Users** to search for valid users.

Update the user registry where new users and groups are stored

About this task

Table 570. Appropriate actions for step: Update the user registry where new users and groups are stored

Actions	Notes
Run step again	You can run the step repeatedly without causing any harm.
Skip step	If the current user repository is correct for new users and groups, you can skip this step.
Clean up step	Complete the following steps from the WebSphere Integrated Solutions Console to change the repository: 1. Go to **Security** > **Global Security** > **Federated repositories** > **Supported entity types**. 2. Click one of the following options to edit the **Base Entry for the Default Parent** to the specific Base Entry for your target repository: • **Group** • **OrgContainer** • **PersonAccount**

Recycle the servers after a security change

During this step, the wizard stops and starts the portal server.

About this task

Table 571. Appropriate actions for step: Recycle the servers after a security change

Actions	Notes
Run step again	Yes, run this step again under the following conditions. • If this step fails, run the step again. • If you are running the configuration again.
Skip step	If you are running the configuration again, you can skip this step only if you skipped all the previous steps.
Clean up step	None required

Update the search administration user

The wizard updates the user ID that is used to manage the search collections.

About this task

Table 572. Appropriate actions for step: Update the search administration user

Actions	Notes
Run step again	Yes, run this step again under the following conditions. • If this step fails, run the step again. • If you are running the configuration again.
Skip step	If you are running the configuration again, you can skip this step only if you skipped all the previous steps.
Clean up step	Log in to the WebSphere Integrated Solutions Console. Go to **Security** > **Global security** > **Java Authentication and Authorization Service** > **J2C authentication data**. Change the user ID and password for the **SearchAdminUser** and the alias.

After you change the security model, the servers need to be restarted

During this step, the wizard stops and starts the portal server.

About this task

Table 573. Appropriate actions for step: After you change the security model, the servers need to be restarted

Actions	Notes
Run step again	Yes, run this step again under the following conditions. • If this step fails, run the step again. • If you are running the configuration again.
Skip step	If you are running the configuration again, you can skip this step only if you skipped all the previous steps.
Clean up step	None required

Verify that all defined attributes are available in the configured LDAP user registry

About this task

Table 574. Appropriate actions for step: Verify that all defined attributes are available in the configured LDAP user registry

Actions	Notes
Run step again	Yes, run this step again under the following conditions: • If this step fails, run the step again. • If you are running the configuration again.
Skip step	If you are running the configuration again, you can skip this step if both of the following conditions are true: • The step completed successfully before • You did not change any attributes when you corrected other failures
Clean up step	None required

Manual Step: Update the appropriate `MemberFixerModule.properties` file with the values for your LDAP users

About this task

Table 575. Appropriate actions for step: Manual Step: Update the appropriate `MemberFixerModule.properties` file with the values for your LDAP users

Actions	Notes
Run step again	Not applicable
Skip step	Yes, if you previously modified the properties file, you can skip this step.
Clean up step	None required

Run the member fixer tool

During this step, the wizard runs the member fixer tool to clean up the entries in the portal server.

About this task

Table 576. Appropriate actions for step: Run the member fixer tool

Actions	Notes
Run step again	Yes, run this step again under the following conditions. • If this step fails, run the step again. • If you are running the configuration again.
Skip step	If you are running the configuration again, you can skip this step only if you skipped all the previous steps.
Clean up step	None required

Manual Step: Map attributes to ensure proper communication between WebSphere Portal and the LDAP server

About this task

Table 577. Appropriate actions for step: Manual Step: Map attributes to ensure proper communication between WebSphere Portal and the LDAP server

Actions	Notes
Run step again	Not applicable
Skip step	If you successfully completed the step before, then skip this step.
Clean up step	None required

Troubleshooting: Create a deployment manager

Create a deployment manager for clustered environments. If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

Each potential step in the configuration is included. Because the steps vary depending on your selections, the steps are not numbered. Find the step that failed to learn more about correcting and recovering from the failure. If you need to change a value that you entered in the wizard, then you must run the configuration again.

Tip: If you must go through the wizard again, download the wizard selections that you made to save time. Then, cancel the configuration. Start the process over and upload your saved selections. Correct or enter values for the parameters that caused the failure.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

Manual Step: Install the deployment manager software

This is a manual step; any errors that occur are outside the context of the wizard.

About this task

Table 578. Appropriate actions for Manual Step: Install the deployment manager software

Actions	Notes
Run the step again	Not applicable
Skip the step	If you successfully completed the step before, then skip this step.
Clean up step	None required

Create the deployment manager profile

About this task

If the step fails, see the logs for the **manageprofiles** command to determine why the step failed. The wizard uses the portal profile templates to create the deployment manager profile. An error might result from a problem with the profile templates. The error message in the log provides more information.

The log files are in the *app_server_root*/logs/manageprofiles directory.

Table 579. Appropriate actions for step: Create the deployment manager profile

Actions	Notes
Run the step again	Run the step again, if it did not complete successfully before.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	If an unrecoverable error occurs and the create deployment manager profile step fails, remove the profile. 1. Use the **manageprofiles** command to remove the profile. The command file is in the app_server_root/bin directory. The command file is a script that is named manageprofiles.sh|bat. Example: `/opt/IBM/WebSphere/AppServer/bin/manageprofiles.sh -delete -profileName dmgr01` 2. Delete the profile directory. 3. Then, run the **Create a Deployment Manager Profile** step again.

Start the deployment manager server

About this task

If the step fails, check the systemout.log for the deployment manager. The log file is in the dmgr_profile/log/*dmgr01* directory.

Table 580. Appropriate actions for step: Start the deployment manager server

Actions	Notes
Run the step again	Run this step again if the deployment manager is not running.
Skip the step	Do not skip this step if the deployment manager is not running. You cannot successfully run the step to augment the profile unless the deployment manager is running.
Clean up step	None required

Augment the deployment manager profile with the portal profile template

About this task

If the step fails, see the logs for the **manageprofiles** command to determine why the step failed. The wizard uses the portal profile templates to create the deployment manager profile. An error might result from a problem with the profile templates. The error message in the log provides more information.

The log files are in the *app_server_root*/logs/manageprofiles directory.

Table 581. Appropriate actions for step: Augment the deployment manager profile with the portal profile template

Actions	Notes
Run the step again	Run the step again, if it did not complete successfully before.
Skip the step	If you successfully completed the step before, then skip this step.
Clean up step	If an unrecoverable error occurs and the augment deployment manager profile step fails, remove the profile. 1. Use the **manageprofiles** command to remove the profile. The command file is in the app_server_root/bin directory. The command file is a script that is named manageprofiles.sh|bat. Example: `/opt/IBM/WebSphere/AppServer/bin/manageprofiles.sh -delete -profileName dmgr01` 2. Delete the profile directory. 3. Then, run the **Create a Deployment Manger Profile** step again.

Stop the deployment manager

About this task

If the step fails, check the systemout.log for the deployment manager. The log file is in the dmgr_profile/log/*dmgr01* directory.

Table 582. Appropriate actions for step: Stop the deployment manager

Actions	Notes
Run the step again	You can run this step again.
Skip the step	Do not skip this step.
Clean up step	None required

Start the deployment manager after the profile augmentation is complete

About this task

If the step fails, check the systemout.log for the deployment manager. The log file is in the dmgr_profile/log/*dmgr01* directory.

Table 583. Appropriate actions for step: Start the deployment manager after the profile augmentation is complete

Actions	Notes
Run the step again	You can run this step again.
Skip the step	Do not skip this step.
Clean up step	None required

Troubleshooting: Create a cluster option

Creating a cluster is part of setting up a clustered environment.If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

Each potential step in the configuration is included. Because the steps vary depending on your selections, the steps are not numbered. Find the step that failed to learn more about correcting and recovering from the failure. If you need to change a value that you entered in the wizard, then you must run the configuration again.

Tip: If you must go through the wizard again, download the wizard selections that you made to save time. Then, cancel the configuration. Start the process over and upload your saved selections. Correct or enter values for the parameters that caused the failure.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

Manual Step: Verify that the portal node and deployment manager system clocks are within 5 minutes of each other

Manual step errors occur outside the context of the wizard.

About this task

Table 584. Appropriate actions for Manual Step: Verify that the portal node and deployment manager system clocks are within 5 minutes of each other

Actions	Notes
Run the step again	Not applicable
Skip the step	If you successfully completed the step before, then skip this step.
Clean up step	None required

Federate the node

When you federate the node, it becomes a managed node in the deployment manager cell.

About this task

If this step fails see the `addNode.log` to determine why the step failed. In most cases, you can correct the error condition and run the step again. You do not have to cancel or reset the configuration steps.

The log is in the `/wp_profile/logs` directory

Table 585. Appropriate actions for step: Federate the node

Actions	Notes
Run the step again	If the step did not complete successfully during the previous configuration, run it again.
Skip the step	If you successfully completed the step before, then skip this step.
Clean up step	Based on where the step failed during the **addNode** task, you might need to remove the node and run the step again. Run the **removeNode** command from the `wp_profile/bin` directory. Example: `wp_profile/bin/removeNode.sh`

Configure the dynamic cluster node

Applies to dynamic clusters only.

About this task

Table 586. Appropriate actions for step: Configure the dynamic cluster node

Actions	Notes
Run the step again	You can run this step repeatedly without causing harm.
Skip the step	If you successfully completed the step before, then skip this step.

Table 586. Appropriate actions for step: Configure the dynamic cluster node (continued)

Actions	Notes
Clean up step	None required

Prepare the node for clustering

If this step fails, click **View Results** to see the applicable section of the `ConfigTrace.log`.

About this task

Table 587. Appropriate actions for step: Prepare the node for clustering

Actions	Notes
Run the step again	You can run this step repeatedly without causing harm.
Skip the step	If you successfully completed the step before, then skip this step.
Clean up step	None required

Complete the cluster setup

About this task

Table 588. Appropriate actions for step: Complete the cluster setup

Actions	Notes
Run the step again	You can run this step repeatedly without causing harm.
Skip the step	If you successfully completed the step before, then skip this step.
Clean up step	None required

Troubleshooting: Create an additional cluster node

Adding a node to a cluster is part of setting up a clustered environment. If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

Each potential step in the configuration is included. Because the steps vary depending on your selections, the steps are not numbered. Find the step that failed to learn more about correcting and recovering from the failure. If you need to change a value that you entered in the wizard, then you must run the configuration again.

Tip: If you must go through the wizard again, download the wizard selections that you made to save time. Then, cancel the configuration. Start the process over and upload your saved selections. Correct or enter values for the parameters that caused the failure.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

Manual Step: Install profile templates

Manual step errors occur outside the context of the wizard.

About this task

Table 589. Appropriate actions for Manual Step: Install profile templates

Actions	Notes
Run the step again	Not applicable
Skip the step	If you successfully completed the step before, then skip this step.
Clean up step	None required

Manual Step: Install portal binary files on the server where you plan to add a node to your cluster

Manual step errors occur outside the context of the wizard.

About this task

Table 590. Appropriate actions for Manual Step: Install portal binary files on the server where you plan to add a node to your cluster

Actions	Notes
Run the step again	Not applicable
Skip the step	If you successfully completed the step before, then skip this step.
Clean up step	None required

Manual Step: Copy the database drivers from the primary node to the additional node

Manual step errors occur outside the context of the wizard.

About this task

Table 591. Appropriate actions for Manual Step: Copy the database drivers from the primary node to the additional node

Actions	Notes
Run the step again	Not applicable
Skip the step	If you successfully completed the step before, then skip this step.
Clean up step	None required

Manual Step: Verify that the portal node and deployment manager system clocks are within 5 minutes of each other

Manual step errors occur outside the context of the wizard.

About this task

Table 592. Appropriate actions forManual Step: Verify that the portal node and deployment manager system clocks are within 5 minutes of each other

Actions	Notes
Run the step again	Not applicable
Skip the step	If you successfully completed the step before, then skip this step.
Clean up step	None required

Create the profile for the secondary portal node

About this task

If the step fails, see the logs for the **manageprofiles** command to determine why the step failed. The wizard uses the portal profile templates to create the deployment manager profile. An error might result from a problem with the profile templates. The error message in the log provides more information.

The log files are in the *app_server_root*/logs/manageprofiles directory.

Use the following table for manual instructions on recovering from a failure. If you prefer to use the Configuration Wizard option, **Remove the WebSphere Portal profile**, to remove the profile in the event of a failure, go to **More options** > **Remove the WebSphere Portal profile** and remove the profile. When the profile is removed successfully, you can run the **Create an Additional Cluster Node** configuration again.

Table 593. Appropriate actions for step: Create the profile for the secondary portal node

Actions	Notes
Run the step again	Run the step again, if it did not complete successfully before.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	If an unrecoverable error occurs and the create deployment manager profile step fails, remove the profile. 1. Use the **manageprofiles** command to remove the profile. The command file is in the app_server_root/bin directory. The command file is a script that is named manageprofiles.sh|bat. Example: /opt/IBM/WebSphere/AppServer/bin/manageprofiles.sh -delete -profileName *dmgr01* 2. Delete the profile directory. 3. Then, run the **Create an Additional Cluster Node** again.

Federate the node

About this task

If this step fails see the addNode.log to determine why the step failed. In most cases, you can correct the error condition and run the step again. You do not have to cancel or reset the configuration steps.

The log is in the /wp_profile/logs directory

Table 594. Appropriate actions for step: Federate the node

Actions	Notes
Run the step again	If the step did not complete successfully during the previous configuration, run it again.
Skip the step	If you successfully completed the step before, then skip this step.
Clean up step	Based on where the step failed during the **addNode** task, you might need to remove the node and run the step again. Run the **removeNode** command from the wp_profile/bin directory. Example: wp_profile/bin/removeNode.sh

Configure the dynamic cluster node

If this step fails, click **View Results** to see the applicable section of the ConfigTrace.log.

About this task

Table 595. Appropriate actions for step: Configure the dynamic cluster node

Actions	Notes
Run the step again	You can run this step repeatedly without causing harm.
Skip the step	If you successfully completed the step before, then skip this step.
Clean up step	None required

Add a secondary node to the cluster

About this task

Table 596. Appropriate actions for step: Add a secondary node to the cluster

Actions	Notes
Run the step again	You can run this step repeatedly without causing harm.
Skip the step	If you successfully completed the step before, then skip this step.
Clean up step	None required

Start the portal server

About this task

Table 597. Appropriate actions for step: Start the portal server

Actions	Notes
Run the step again	You can run this step repeatedly without causing harm.
Skip the step	Not applicable
Clean up step	None required

Troubleshooting: Create a WebSphere Portal profile

View troubleshooting information for creating a WebSphere Portal Express profile.

Each potential step in the configuration is included. Because the steps vary depending on your selections, the steps are not numbered. Find the step that failed to learn more about correcting and recovering from the failure. If you need to change a value that you entered in the wizard, then you must run the configuration again.

Tip: If you must go through the wizard again, download the wizard selections that you made to save time. Then, cancel the configuration. Start the process over and upload your saved selections. Correct or enter values for the parameters that caused the failure.

If you are on WebSphere Portal Express Version 8.5 without a combined cumulative fix applied, then you can use this option in the Configuration Wizard to create an additional profile.

Important: You cannot complete this configuration option, if you have CF01 or a later combined cumulative fix applied.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

Create the target profile for WebSphere Portal Express in the WebSphere Application Server

About this task

If the step fails, see the logs for the **manageprofiles** command to determine why the step failed. The wizard uses the portal profile templates to create the deployment manager profile. An error might result from a problem with the profile templates. The error message in the log provides more information.

The log files are in the *app_server_root*/logs/manageprofiles directory.

Important: WebSphere Application Server Version 8.5.5.5 requires that fix PI37248 is applied when creating the managed portal profile. This step fails if PI37248 is not installed.

Table 598. Appropriate actions for step: Create the target profile for WebSphere Portal Express in the WebSphere Application Server

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	If an unrecoverable error occurs and this step fails, remove the profile. Use the **manageprofiles** command to remove the profile. The command file is in the app_server_root/bin directory. The command file is a script that is named manageprofiles.sh|bat. Example: `/opt/IBM/WebSphere/AppServer/bin/manageprofiles.sh -delete -profileName dmgr01` Delete the profile directory only if the **manageprofiles** command completes successfully. Then, run **Create the target profile for WebSphere Portal** again.

Install the ConfigEngine into the target WebSphere Portal Express profile

About this task

If you completed a binary installation, this is your first step.

Table 599. Appropriate actions for step: Install the ConfigEngine into the target WebSphere Portal profile

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	If this step fails, look in configtrace.log for any failures.

Register the WebSphere Portal Express components with the ConfigEngine

About this task

Table 600. Appropriate actions for step: Register the components with the ConfigEngine

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again provided you keep the same profile name.
Clean up step	If this step fails, look in configtrace.log for any failures.

Consolidate the properties files for WebSphere Portal Express components used in this configuration into a single properties file

About this task

Table 601. Appropriate actions for step: Consolidate the properties files for WebSphere Portal components used in this configuration into a single properties file

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	If this step fails, look in `configtrace.log` for any failures.

Prepare the profile for basic configuration

About this task

Table 602. Appropriate actions for step: Prepare the profile for basic configuration

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	If this step fails, look in `configtrace.log` for any failures.

Validate the database connection and environment

About this task

Table 603. Appropriate actions for step: Validate the database connection and environment

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	Contact support.

Deploy applications into the portal profile

About this task

Table 604. Appropriate actions for step: Deploy applications into the portal profile

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	If this step fails, look in `configtrace.log` for any failures.

Configure the JCR, theme, and core runtime components of your portal server

About this task

Table 605. Appropriate actions for step: Configure the JCR theme, and core runtime components of your portal server

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	If this step fails, look in `configtrace.log` for any failures.

Deploy the administration portlets and pages to the portal

About this task

Table 606. Appropriate actions for step: Deploy the administration portlets and pages to the portal

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	If this step fails, look in `configtrace.log` for any failures.

Deploy the out-of-box pages and portlets to the portal

About this task

Table 607. Appropriate actions for step: Deploy the out-of-box pages and portlets to the portal

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	If this step fails, look in `configtrace.log` for any failures.

Remove the application server (server1) from the profile

About this task

Table 608. Appropriate actions for step: Remove the application server (server1) from the profile

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If you already removed server1, you do not need to run this step again.
Clean up step	If this step fails, look in `configtrace.log` for any failures.

Stop the portal server

About this task

Table 609. Appropriate actions for step: Stop the portal server

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	Do not skip this step, if you are running the configuration again.
Clean up step	Check for failures in the `systemout` and `systemerror` logs for starting or stopping the portal server.

Collect the deployment manager augmentation files and profile templates that are required to build a cell

About this task

Table 610. Appropriate actions for step: Collect the deployment manager augmentation files and profile templates that are required to build a cell

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	If this step fails, look in `configtrace.log` for any failures.

Restart the WebSphere Portal Express server

About this task

Table 611. Appropriate actions for step: Restart the WebSphere Portal Server

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	Do not skip this step, if you are running the configuration again.
Clean up step	Check for failures in the `systemout` and `systemerror` logs for starting or stopping the portal server.

Troubleshooting: Remove a WebSphere Portal profile

View troubleshooting information for creating a WebSphere Portal Express profile.

Each potential step in the configuration is included. Because the steps vary depending on your selections, the steps are not numbered. Find the step that failed to learn more about correcting and recovering from the failure. If you need to change a value that you entered in the wizard, then you must run the configuration again.

Tip: If you must go through the wizard again, download the wizard selections that you made to save time. Then, cancel the configuration. Start the process over and upload your saved selections. Correct or enter values for the parameters that caused the failure.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

Manual Step: Prepare your system

About this task

Table 612. Appropriate actions for step: Prepare your system

Actions	Notes
Run the step again	Not applicable
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	None required

Remove portal node from cluster

About this task

Table 613. Appropriate actions for step: Remove portal node from cluster

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	Review `removeNode.log` in `wp_profile/logs` for any failures that are indicated by the words "exception" or "error."

Remove portal profile

About this task

Table 614. Appropriate actions for step: Remove portal profile

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	1. Use the **manageprofiles** command to remove the profile. The command file is in the `app_server_root/bin` directory. The command file is a script that is named `manageprofiles.sh|bat`. Example: `/opt/IBM/WebSphere/AppServer/bin/manageprofiles.sh -delete -profileName profile_name` 2. Manually delete the profile directory only if the `manageprofiles -delete` command completes successfully. The **manageprofiles** command creates a log for every profile that it creates, deletes, or augments. If the `manageprofiles -delete` command does not complete successfully, review the logs that are named `profile_name_create.log` and `profile_name_augment.log` in `install_root/logs/manageprofiles`.

Stop the deployment manager

About this task

Table 615. Appropriate actions for step: Stop the deployment manager

Actions	Notes
Run the step again	Run this step again, only if you are running the configuration again and removing the deployment manager profile.
Skip the step	Do not skip this step, if you are running the configuration again to remove the deployment manager profile. The deployment manager must be stopped to remove the deployment manager profile.
Clean up step	None required You can check the status of the deployment manager by running `server-status.sh|bat`.

Remove the deployment manager profile

About this task

Table 616. Appropriate actions for step: Remove the deployment manager profile

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	1. Use the **manageprofiles** command to remove the profile. The command file is in the `app_server_root/bin` directory. The command file is a script that is named `manageprofiles.sh|bat`. Example: `/opt/IBM/WebSphere/AppServer/bin/manageprofiles.sh -delete -profileName profile_name` 2. Manually delete the profile directory only if the `manageprofiles -delete` command completes successfully. The **manageprofiles** command creates a log for every profile that it creates, deletes, or augments. If the `manageprofiles -delete` command does not complete successfully, review the log named `delete.log` in `install_root/logs/dmgr_01`.

Troubleshooting: Migrate a stand-alone server

If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

Each potential step in the configuration is included. Because the steps vary depending on your selections, the steps are not numbered. Find the step that failed to learn more about correcting and recovering from the failure. If you need to change a value that you entered in the wizard, then you must run the configuration again.

Tip: If you must go through the wizard again, download the wizard selections that you made to save time. Then, cancel the configuration. Start the process over and upload your saved selections. Correct or enter values for the parameters that caused the failure.

Related concepts:

"Migrate a stand-alone server" on page 842
Use the Configuration Wizard to migrate a stand-alone server environment. Use the following information to get familiar with the information that you must provide in the wizard and the configuration procedure that it generates.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

Manual Step: Install the latest fix packs

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 617. Appropriate actions for step: Install the latest fix packs

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	If you encounter an issue when you are installing the fix, refer to the documentation for the fix.

Generate the files for remote migration

About this task

Table 618. Appropriate actions for step: Generate files for remote migration

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	If this step fails, review the parameters and values that you entered in the Configuration Wizard, specifically the target temporary path and the application server path. If the parameter and values that you entered are correct, and the step fails again, use the wp-collector tool to gather the files that are needed to contact support for help. See "Data collection and symptom analysis" on page 3539 for information about using the wp-collector tool.

Manual Step: Copy the remote migration package to the source environment

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 619. Appropriate actions for step: Copy the remote migration package to the source environment

Actions	Notes
Run the step again	Not applicable
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	None required

Create a backup of the remote source portal profile

About this task

Table 620. Appropriate actions for step: Create a backup of the remote source portal profile

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	If this step fails, delete the path to the temporary backup profile, and run the step again. For more information about troubleshooting the WASPreUpgrade command, see the WebSphere Application Server documentation on Troubleshooting migration.

Create a backup profile of the source portal profile

About this task

Table 621. Appropriate actions for step: Create a backup of the source portal profile

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	If this step fails, delete the path to the temporary backup profile, and run the step again. For more information about troubleshooting the WASPreUpgrade command, see the WebSphere Application Server documentation on Troubleshooting migration.

Manual Step: If the backup profile is larger than 2 GB, clean up the backup profile

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 622. Appropriate actions for step: If the backup profile is larger than 2 GB, clean up the backup profile

Actions	Notes
Run the step again	Not applicable
Skip the step	If you are running the configuration again, you can skip this step only if you skipped all the previous steps.
Clean up step	None required

Create a default profile

About this task

Table 623. Appropriate actions for step: Create a default profile

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	If an unrecoverable error occurs and the create default profile step fails, remove the profile. 1. Use the **manageprofiles** command to remove the profile. The command file is in the `app_server_root/bin` directory. The command file is a script that is named `manageprofiles.sh|bat`. Example: `/opt/IBM/WebSphere/AppServer/bin/manageprofiles.sh -delete -profileName profile_name` 2. Delete the profile directory. 3. Then, run the **Create a default profile** step again.

Import backup profile

About this task

Table 624. Appropriate actions for step: Import backup profile

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	Do not skip this step, if you are running the configuration again. If you re-create the default profile from the **Create a default profile** step, then you must run this step again to import the new default profile.
Clean up step	If an unrecoverable error occurs and the import backup profile step fails, remove the profile. 1. Use the **manageprofiles** command to remove the profile. The command file is in the `app_server_root/bin` directory. The command file is a script that is named `manageprofiles.sh|bat`. Example: `/opt/IBM/WebSphere/AppServer/bin/manageprofiles.sh -delete -profileName profile_name` 2. Delete the profile directory. 3. Then, run the **Create a default profile** step again before you rerun this step.

Manual Step: If you cleaned up the backup profile, restore the JCR content

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 625. Appropriate actions for step: If you cleaned up the backup profile, restore the JCR content

Actions	Notes
Run the step again	Not applicable
Skip the step	Do not skip this step, if you are running the configuration again. You must restore the JCR content, if you completed the previous manual step to clean up the backup profile that is over 2 GB.
Clean up step	None required

Upgrade the ConfigEngine

About this task

Table 626. Appropriate actions for step: Upgrade the ConfigEngine

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	Do not skip this step, if you are running the configuration again.
Clean up step	If this step fails, review the parameters and values that you entered in the Configuration Wizard, specifically the new host name, passwords, port numbers, and the Portal server path. If the parameter and values that you entered are correct, and the step fails again, use the wp-collector tool to gather the files that are needed to contact support for help. See "Data collection and symptom analysis" on page 3539 for information about using the wp-collector tool.

Manual Step: Update the ports on the target environment

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 627. Appropriate actions for step: Update the ports on the target environment

Actions	Notes
Run the step again	Not applicable
Skip the step	Do not skip this step, if you are running the configuration again.
Clean up step	None required

Manual Step: Update database settings

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 628. Appropriate actions for step: Update database settings

Actions	Notes
Run the step again	Not applicable
Skip the step	Do not skip this step, if you are running the configuration again.
Clean up step	None required

Validate database settings

About this task

Table 629. Appropriate actions for step: Validate database settings

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	Do not skip this step, if you are running the configuration again.
Clean up step	Check your properties files to make sure that you have all of your parameters and values set correctly before you run the step again.

Connect to new database copies

About this task

Table 630. Appropriate actions for step: Connect to new database copies

Actions	Notes
Run the step again	If this step fails, you can run the step again.
Skip the step	Do not skip this step, if you are running the configuration again.
Clean up step	None required.

Manual Step: Review database schema changes

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 631. Appropriate actions for step: Review database schema changes

Actions	Notes
Run the step again	If this step fails, you can run this step again.
Skip the step	This step is optional. You can skip this step if you do not want to review the database schema changes.
Clean up step	None required

Upgrade the base portal database component

About this task

Table 632. Appropriate actions for step: Upgrade the base portal database component

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	Do not skip this step, if you are running the configuration again.
Clean up step	If this step fails, delete the database, create a new copy, and run the step again.

Manual Step: Remove check pending statuses from table spaces

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 633. Appropriate actions for step: Remove check pending statuses from table spaces

Actions	Notes
Run the step again	If this step fails, clean up the issue and start back with the **Upgrade the base portal database components** step, and then run this step again.
Skip the step	Do not skip this step, if you are running the configuration again.
Clean up step	If this step fails, delete the database, create a new copy, and rerun the **Upgrade the portal database component** step before you rerun this step again.

Upgrade the remaining portal databases

About this task

Table 634. Appropriate actions for step: Upgrade the remaining portal databases

Actions	Notes
Run the step again	If this step fails, clean up the issue and start back with the **Upgrade the base portal database components**.

Table 634. Appropriate actions for step: Upgrade the remaining portal databases (continued)

Actions	Notes
Skip the step	Do not skip this step, if you are running the configuration again. You must complete the **Upgrade the base portal database components** and **Remove check pending statuses from table spaces** steps before you run this step again.
Clean up step	If this step fails, delete the database, create a new copy, and rerun the **Upgrade the portal database component** and **Remove check pending statuses from table spaces** steps before you run this step again.

Upgrade the portal profile

About this task

Table 635. Appropriate actions for step: Upgrade the portal profile

Actions	Notes
Run the step again	If this step fails, you must contact support. **Note:** Contact support before you start the Portal server.
Skip the step	Do not skip this step, if you are running the configuration again.
Clean up step	Contact support.

When you run this step, the sub task that is named **`action-deploy-portlets-applyMIGStatic-wp.oob.full`** runs and completes successfully. However, the following error messages are shown. You can ignore these error messages:

- `EJPXA0161W: The web module ContactList could not be activated. Please see previous messages for reasons and possible corrective actions.`
- `EJPPH0048W: The synchronization mode of all nodes in the portal cluster is not consistently set. The portlet application PA_ContactList will not be started in the Application Server. Manual synchronization is assumed for all nodes. Manually start the application after all nodes were synchronized.`
- `EJPXA0067E: The following configuration data is needed to create a content-node resource: content-parentref.`

Troubleshooting: Migrate the deployment manager profile for a cluster environment

If you encounter a failure during the configuration process, determine whether you can run the step again, skip the step, or if you must clean up the step. For some failed steps, learn how to correct the issue and recover from the failure.

Each potential step in the configuration is included. Because the steps vary depending on your selections, the steps are not numbered. Find the step that failed to learn more about correcting and recovering from the failure. If you need to change a value that you entered in the wizard, then you must run the configuration again.

Tip: If you must go through the wizard again, download the wizard selections that you made to save time. Then, cancel the configuration. Start the process over and upload your saved selections. Correct or enter values for the parameters that caused the failure.

Related concepts:

"Cluster: Migrate the deployment manager profile" on page 847
Use the Configuration Wizard to migrate the deployment manager profile for a cluster environment. Use the following information to get familiar with the information that you must provide in the wizard and the configuration procedure that it generates.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

Manual Step: Disable automatic synchronization on all nodes in the cluster

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 636. Appropriate actions for step: Disable automatic synchronization on all nodes in the cluster

Actions	Notes
Run the step again	You can run this step again if it was not successful.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	None required

Manual Step: Install the latest fix packs

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 637. Appropriate actions for step: Install the latest fix packs

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	If you encounter an issue when you are installing the fix, refer to the documentation for the fix.

Manual Step: Install the Portal and WebSphere binary files

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 638. Appropriate actions for step: Install the Portal and WebSphere binary files

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, do not run this step again.
Clean up step	Complete an uninstall of the product, and delete the remaining file structure. Then, start the configuration from the beginning.

Manual Step: Copy required portal binary files to the target deployment manager

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 639. Appropriate actions for step: Copy required portal binary files to the target deployment manager

Actions	Notes
Run the step again	Not applicable
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	None required

Manual Step: Generate files for remote migration on the deployment manager

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 640. Appropriate actions for step: Generate files for remote migration on the deployment manager

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again. This step is optional. You do not need to complete this step, if the deployment manager is in the same binary as the primary node.
Clean up step	If this step fails, review the parameters and values that you entered in the Configuration Wizard, specifically the target temporary path and the application server path. If the parameter and values that you entered are correct, and the step fails again, use the wp-collector tool to gather the files that are needed to contact support for help. See "Data collection and symptom analysis" on page 3539 for information about using the wp-collector tool.

Manual Step: Copy the remote migration package to the source environment

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 641. Appropriate actions for step: Copy the remote migration package to the source environment

Actions	Notes
Run the step again	Not applicable
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	None required

Manual Step: Create a backup of the source deployment manager

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 642. Appropriate actions for step: Create a backup of the source deployment manager

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.

Table 642. Appropriate actions for step: Create a backup of the source deployment manager (continued)

Actions	Notes
Clean up step	If this step fails, delete the path to the temporary backup profile, and run the step again. For more information about troubleshooting the WASPreUpgrade command, see the WebSphere Application Server documentation on Troubleshooting migration. **IBMi only:** If this step fails, remove the **oldProfile** parameter and run the step again.

Manual Step: Create a default deployment manager profile

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 643. Appropriate actions for step: Create a default deployment manager profile

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	If an unrecoverable error occurs and the create default profile step fails, remove the profile. 1. Use the **manageprofiles** command to remove the profile. The command file is in the app_server_root/bin directory. The command file is a script that is named manageprofiles.sh|bat. Example: `/opt/IBM/WebSphere/AppServer/bin/manageprofiles.sh -delete -profileName profile_name` 2. Delete the profile directory. 3. Then, run the **Create a default deployment manager profile** step again.

Manual Step: Import the backup profile

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 644. Appropriate actions for step: Import the backup profile

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	Do not skip this step, if you are running the configuration again. If you re-create the default profile from the **Create a default deployment manager profile** step, then you must run this step again to import the new default profile.
Clean up step	If an unrecoverable error occurs and the import backup profile step fails, remove the profile. 1. Use the **manageprofiles** command to remove the profile. The command file is in the app_server_root/bin directory. The command file is a script that is named manageprofiles.sh|bat. Example: `/opt/IBM/WebSphere/AppServer/bin/manageprofiles.sh -delete -profileName profile_name` 2. Delete the profile directory. 3. Then, run the **Create a default deployment manager profile** step again before you rerun this step.

Troubleshooting: Migrate node profiles for a cluster environment

If you encounter a failure during the migration of the node profiles for a cluster environment, learn how to correct the issue and recover from the failure.

Each potential step in the migrate node profiles option is included. Since the steps vary, depending on your selections, the steps are not numbered. Find the step that failed to learn more about correcting and recovering from the failure.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

Manual Step: Stop the source deployment manager and node agents

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 645. Appropriate actions for step: Stop the source deployment manager and node agents

Actions	Notes
Run the step again	You can run this step again if it was not successful.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	None required

Manual Step: Start the target deployment manager

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 646. Appropriate actions for step: Start the target deployment manager

Actions	Notes
Run the step again	You can run this step again if it was not successful.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	None required

Generate the files for remote migration

About this task

Table 647. Appropriate actions for step: Generate the files for remote migration

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	If this step fails, review the parameters and values that you entered in the Configuration Wizard, specifically the target temporary path and the application server path. If the parameter and values that you entered are correct, and the step fails again, use the wp-collector tool to gather the files that are needed to contact support for help. See "Data collection and symptom analysis" on page 3539 for information about using the wp-collector tool.

Manual Step: Copy the remote migration package to the source environment

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 648. Appropriate actions for step: Copy the remote migration package to the source environment

Actions	Notes
Run the step again	Not applicable
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	None required

Create a backup of the source portal profile

About this task

Table 649. Appropriate actions for step: Create a backup of the source portal profile

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	If this step fails, delete the path to the temporary backup profile, and run the step again. For more information about troubleshooting the WASPreUpgrade command, see the WebSphere Application Server documentation on Troubleshooting migration.

Manual Step: Create a backup of the remote source portal profile

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 650. Appropriate actions for step: Create a backup of the remote source portal profile

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	If this step fails, delete the path to the temporary backup profile, and run the step again. For more information about troubleshooting the WASPreUpgrade command, see the WebSphere Application Server documentation on Troubleshooting migration.

Manual Step: Update the deployment manager settings

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 651. Appropriate actions for step: Update the deployment manager settings

Actions	Notes
Run the step again	Not applicable
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	None required

Manual Step: If the backup profile is larger than 2 GB, clean up the backup profile

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 652. Appropriate actions for step: If the backup profile is larger than 2 GB, clean up the backup profile

Actions	Notes
Run the step again	Not applicable
Skip the step	If you are running the configuration again, you can skip this step only if you skipped all the previous steps.
Clean up step	None required

Create a default profile

About this task

Table 653. Appropriate actions for step: Create a default profile

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	If an unrecoverable error occurs and the create default profile step fails, remove the profile. 1. Use the **manageprofiles** command to remove the profile. The command file is in the `app_server_root/bin` directory. The command file is a script that is named `manageprofiles.sh|bat`. Example: `/opt/IBM/WebSphere/AppServer/bin/manageprofiles.sh -delete -profileName profile_name` 2. Delete the profile directory. 3. Then, run the **Create a default profile** step again.

Import the backup profile

About this task

Table 654. Appropriate actions for step: Import the backup profile

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	Do not skip this step, if you are running the configuration again. If you re-create the default profile from the **Create a default profile** step, then you must run this step again to import the new default profile.
Clean up step	If an unrecoverable error occurs and the import backup profile step fails, remove the profile. 1. Use the **manageprofiles** command to remove the profile. The command file is in the `app_server_root/bin` directory. The command file is a script that is named `manageprofiles.sh|bat`. Example: `/opt/IBM/WebSphere/AppServer/bin/manageprofiles.sh -delete -profileName profile_name` 2. Delete the profile directory. 3. Then, run the **Create a default profile** step again before you rerun this step.

Manual Step: If you cleaned up the backup profile, restore the JCR content

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 655. Appropriate actions for step: If you cleaned up the backup profile, restore the JCR content

Actions	Notes
Run the step again	Not applicable
Skip the step	Do not skip this step, if you are running the configuration again. You must restore the JCR content, if you completed the previous manual step to clean up the backup profile that is over 2 GB.
Clean up step	None required

Troubleshooting: Upgrade node profiles for a cluster environment

If you encounter a failure while upgrading the node profiles for a cluster environment, learn how to correct the issue and recover from the failure.

Each potential step in the upgrade node profiles option is included. Since the steps vary, depending on your selections, the steps are not numbered. Find the step that failed to learn more about correcting and recovering from the failure.

Related concepts:

"Cluster: Upgrade node profiles" on page 853
Use the Configuration Wizard to upgrade the nodes profiles for a cluster environment. Use the following information to get familiar with the information you must provide in the wizard and the configuration procedure that it generates.

Related tasks:

"Accessing the Configuration Wizard" on page 235
The home page of the Configuration Wizard provides access to common configuration tasks. Supported tasks include setting up a stand-alone server, setting up a cluster, transferring from Apache Derby to another supported database, migrating your server, installing add-ons, and more.

Manual Step: Update the ports for the deployment manager and nodes

About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 656. Appropriate actions for step: Update the ports for the deployment manager and nodes

Actions	Notes
Run the step again	You can run this step again if it fails.
Skip the step	If this step was successful, you can skip it if you run the configuration process again.
Clean up step	None required

Upgrade the ConfigEngine
About this task

Table 657. Appropriate actions for step: Upgrade the ConfigEngine

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again. If you re-create your profile for any reason, you must run this step again.
Clean up step	If this step fails, review the parameters and values that you entered in the Configuration Wizard, specifically the new host name, passwords, port numbers, and the Portal server path. If the parameter and values that you entered are correct, and the step fails again, use the wp-collector tool to gather the files that are needed to contact support for help. See "Data collection and symptom analysis" on page 3539 for information about using the wp-collector tool.

Update database settings
About this task

Table 658. Appropriate actions for step: Update database settings

Actions	Notes
Run the step again	Not applicable
Skip the step	If this step was successful, you can skip it if you run the configuration process again. If you re-create your profile for any reason, you must run this step again.
Clean up step	None required

Validate the database settings
About this task

Table 659. Appropriate actions for step: Validate the database settings

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	If this step was successful, you can skip it if you run the configuration process again. If you re-create your profile for any reason, you must run this step again.
Clean up step	Check your properties files to make sure that you have all of your parameters and values set correctly before you run the step again.

Connect to new databases
About this task

Table 660. Appropriate actions for step: Connect to new databases

Actions	Notes
Run the step again	If this step fails, you can run the step again.
Skip the step	If this step was successful, you can skip it if you run the configuration process again. If you re-create your profile for any reason, you must run this step again.
Clean up step	None required

Manual Step: Review database schema changes
About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 661. Appropriate actions for step: Review database schema changes

Actions	Notes
Run the step again	If this step fails, you can run this step again.
Skip the step	This step is optional. You can skip this step if you do not want to review the database schema changes.
Clean up step	None required

Upgrade the base portal database component
About this task

Table 662. Appropriate actions for step: Upgrade the base portal database component

Actions	Notes
Run the step again	If this step fails, you can run this step again after you clean up the issue.
Skip the step	Do not skip this step, if you are running the configuration again.
Clean up step	If this step fails, delete the database, create a new copy, and run the step again.

Manual Step: Remove check pending statuses from table spaces
About this task

Since this is a manual step, any error that occurs is outside the context of the wizard.

Table 663. Appropriate actions for step: Remove check pending statuses from table spaces

Actions	Notes
Run the step again	If this step fails, clean up the issue and start back with the **Upgrade the base portal database components** step, and then run this step again.
Skip the step	Do not skip this step, if you are running the configuration again.
Clean up step	If this step fails, delete the database, create a new copy, and run the **Upgrade the portal database component** step again before you rerun this step again.

Upgrade the remaining portal databases
About this task

Table 664. Appropriate actions for step: Upgrade the remaining portal databases

Actions	Notes
Run the step again	If this step fails, clean up the issue and start back with the **Upgrade the base portal database components**.
Skip the step	Do not skip this step, if you are running the configuration again. You must complete the **Upgrade the base portal database components** and **Remove check pending statuses from table spaces** steps before you run this step again.
Clean up step	If this step fails, delete the database, create a new copy, and rerun the **Upgrade the portal database component** and **Remove check pending statuses from table spaces** steps before you run this step again.

Upgrade the portal profile
About this task

Table 665. Appropriate actions for step: Upgrade the portal profile

Actions	Notes
Run the step again	If this step fails, you must contact support. **Note:** Contact support before you start the Portal server.
Skip the step	Do not skip this step, if you are running the configuration again.
Clean up step	Contact support.

When you run this step, the sub task that is named **`action-deploy-portlets-applyMIGStatic-wp.oob.full`** runs and completes successfully. However, the following error messages are shown. You can ignore these error messages:

- EJPXA0161W: The web module ContactList could not be activated. Please see previous messages for reasons and possible corrective actions.
- EJPPH0048W: The synchronization mode of all nodes in the portal cluster is not consistently set. The portlet application PA_ContactList will not be started in the Application Server. Manual synchronization is assumed for all nodes. Manually start the application after all nodes were synchronized.
- EJPXA0067E: The following configuration data is needed to create a content-node resource: content-parentref.

Error message codes

Each message code consists of a product identifier, component identifier, a unique number, and a message type identifier. The product identifier is EJP and there are many components. There are three message types: error, information, and warning.

Example

For the following message code EJPIC0001E:

- EJP is the product identifier
- IC is the component identifier, installation user interface
- 0001 is the unique message number
- E is the message type, error

During the last session, you clicked Start Configuration and the configuration was running automatically. To return to the automatic process, click Resume Configuration. Otherwise, click Cancel to run each remaining step manually.

Contact support

For contact information, see to the IBM Software Support site.

Related information:

IBM Software Support

Chapter 18. Reference

View information that helps you use the information center including directory conventions, terms of use, trademarks, a glossary, and more.

"Conventions"
Understand the conventions that are used in this documentation to use it more effectively.

"Directory structure" on page 3618
The topic shows the naming conventions that are used to denote the location of files on the servers and the types of resources you can find in those directories.

"Supported languages" on page 3622
The different user interfaces used to install, configure, and use IBM WebSphere Portal Express might not be available in all languages. Refer to the following table for the language codes and to see which areas of the product are available in your language. If your language is not supported currently, you might see the English version of the user interface.

"Updates using ReleaseBuilder" on page 3624
After setting up your initial staging and production servers, you can use ReleaseBuilder to make updates to your production server.

"Staging Personalization rules to production" on page 3629
Use the steps in this file to move Personalization rules from a staging system to a production system. There are a number of methods for moving rules between servers, each suitable for different situations.

"CF04 and earlier: Using friendly URLs without state information" on page 3631
By default, WebSphere Portal Express URLs include navigational state information. If you configure pages for friendly URLs, the portal appends the state information to the friendly URLs. Some scenarios require short and fully human readable URLs that omit the state information. For such scenarios, you can configure friendly URLs so that the portal does not show that state information.

"Terms of use" on page 3634
Understand the terms and conditions before you use this publication.

"Notices" on page 3634
This information was developed for products and services offered in the U.S.A. This material may be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

"Glossary" on page 3637
This glossary includes terms and definitions for IBM WebSphere Portal Express.

Conventions

Understand the conventions that are used in this documentation to use it more effectively.

The following section provides conventions that can help you interpret the information that is provided in this documentation:

- File names, directories, and commands appear in `Courier` font. For example:
  - File name: `xmlaccess.bat` or `xmlaccess.sh`

  - Directory: `/opt/WebSphere/PortalServer`
  - Command: `startServer WebSphere_Portal`
- Variables are either italicized, enclosed in brackets, or both. For example: `http://`*`hostname.example.com`*`:`*`10039`*`/wps/portal`, where *hostname.example.com* is the fully qualified host name of the server where Portal is running and *10039* is the default transport port that is created by WebSphere Application Server. The port number might be different for your environment.
- Variables are used to indicate root installation directories. For more information, see "Directory structure."
- Directories are shown with forward slashes (/), unless operating-system specific information is provided. On Windows systems, you must use backward slashes (\) when typing at a command line, unless otherwise noted.
- Operating system-specific information is provided, for example:
  - Linux : **`./ConfigEngine.sh`** *task_name*
  - IBM i: **`ConfigEngine.sh`** *task_name*
  - Windows: **`ConfigEngine.bat`** *task_name*
- Links to reference information and external links are marked with the **For your reference** icon: .
- Most topics include a Related information section that links to other relevant topics.

Directory structure

The topic shows the naming conventions that are used to denote the location of files on the servers and the types of resources you can find in those directories.

PortalServer_root

Throughout this documentation, the installation location for the portal server component of IBM WebSphere Portal Express is noted as *PortalServer_root*.

For the IBM i operating system, an extra variable is used to indicate the user data directory. The user data directory is noted as *PortalServer_root_user*.

The following information shows the default location if it is not otherwise specified during installation:

IBM i

- `portal_server_root` (ProdData)
  - `/QIBM/ProdData/WebSphere/PortalServer/V85/<product offering>`

  Where *product offering* is Server or Express
- `PortalServer_root_user` (UserData)
  - WebSphere Application Server Version 8.5.5 for Network Deployment:

- /QIBM/UserData/WebSphere/AppServer/V85/ND/profiles/wp_profile/PortalServer

Linux /opt/IBM/WebSphere/PortalExpress/PortalServer

Windows
C:\Program Files\IBM\WebSphere\PortalExpress\PortalServer

wp_profile_root

Throughout this documentation, the profile location is noted as *wp_profile_root*. The following information shows the default profile location if another location is not specified during installation:

IBM i

- WebSphere Application Server Version 8.5 for Network Deployment:
  - /QIBM/UserData/WebSphere/AppServer/V85/ND/profiles/wp_profile

The *wp_profile* is the default profile name but is used here as an example since there can be multiple profiles with self described or incremental names (for example, wp_profile1, wp_profile2).

Linux /opt/IBM/WebSphere/PortalExpress/AppServer/profiles/wp_profile

Windows
C:\Program Files\IBM\WebSphere\wp_profile

ConfigEngine_root

Throughout this documentation, the installation location for the Configuration Engine component is noted as *ConfigEngine_root*.

IBM i /QIBM/ProdData/WebSphere/PortalServer/V85/ConfigEngine

Linux /opt/IBM/WebSphere/PortalExpress/AppServer/profiles/ConfigEngine

Windows
C:\Program Files\IBM\WebSphere\ConfigEngine

WebSphere Portal Express Configuration Engine profile directory

The Configuration Engine profile directory is the location of the **ConfigEngine** task.

IBM i

- WebSphere Application Server Version 8.5.5 for Network Deployment:
  - /QIBM/UserData/WebSphere/AppServer/V85/ND/profiles/wp_profile/ConfigEngine

The *wp_profile* is the default profile name but is used here as an example since there can be multiple profiles with self described or incremental names (for example, wp_profile1, wp_profile2).

Linux /opt/IBM/WebSphere/PortalExpress/AppServer/profiles/wp_profile/ConfigEngine

Windows
C:\Program Files\IBM\WebSphere\wp_profile\ConfigEngine

WebSphere Portal Express directory structure after installation

WebSphere Portal Express has the following directory structure after installation:

Note: On the Linux and IBM i operating systems, all directories are r/o.

```
PortalServer_root          Root directory for WebSphere Portal Express
 |
 +-- ap
 |
 +-- base
 |
 +-- bin                    WebSphere Portal Express tools
 |
 +-- bp
 |
 +-- doc                    Javadoc and sample XMLAccess input files
 |
 +-- ext
 |
 +-- filesForDmgr
 |
 +-- installer
 |
 +-- jcr                    Resources for the Content Repository
 |
 +-- license                WebSphere Portal Express license agreement
 |
 +-- lwo
 |
 +-- lwp04.infra
 |
 +-- people
 |
 +-- prereq
 |
 +-- prereqs.infra
 |
 +-- profileTemplates
 |
 +-- properties
 |
 +-- pzn
 |
 +-- pzn.ext
 |
 +-- search
 |
 +-- shared                 Shared resources, including runtime JARs, TLDs,
 |                          and other resources.
 |                          The /app subdirectory is the application server's
 |                          WPSLib shared library for WebSphere Portal Express
 |
 +-- solutionInstaller
 +-- theme
 |
 +-- ui
 |
 +-- version                Version information for various components
 |
 +-- wcm                    Source Web application files for web content manager
 |
 +-- wps.properties
```

AppServer_root

The following information shows the WebSphere Application Server installation directory:

WebSphere Portal Configuration profile directory *cw_profile_root*

Throughout this documentation, the configuration wizard profile location is noted as cw_profile_root. The following information shows the profile location:

IBM i WebSphere Application Server Version 8.5 for Network Deployment:/QIBM/UserData/WebSphere/AppServer/V8/ND/profiles/cw_profile

Linux /opt/IBM/WebSphere/AppServer/profiles/cw_profile

Windows
C:\Program Files\IBM\WebSphere\AppServer\profiles\cw_profile

IBM i The installation location for WebSphere Application Server is noted as *app_server_root* and refers to the UserData path, unless otherwise specified in the topic where you see it. The *profile_root* following variable refers to the name given to the WebSphere Application Server profile in use.

The following information shows the default WebSphere Application Server installation location if it is not otherwise specified during installation:

IBM i The installation location for WebSphere Application Server is noted as *app_server_root* and refers to the UserData path, unless otherwise specified in the topic where you see it. The *profile_root* following variable refers to the name given to the WebSphere Application Server profile in use.

Linux /opt/IBM/WebSphere/PortalExpress/AppServer

Windows
C:\Program Files\IBM\WebSphere\PortalExpress\AppServer

Directories for languages

The following table shows the languages that are supported by WebSphere Portal Express and the directories that are used for storing locale-specific resources. These directories are used in portlet web application directories and in the WebSphere Portal Express enterprise application (themes, skins, and other web application resources).

Table 666. Languages supported by WebSphere Portal Express

Language (locale)	Directory
Arabic	/ar
Danish	/da
English	/en
French	/fr
Italian	/it
Kazakh	/kk
Korean	/ko
Polish	/pl

Table 666. Languages supported by WebSphere Portal Express (continued)

Language (locale)	Directory
Romanian	/ro
Slovenian	/sl
Turkish	/tr
Traditional Chinese	/zh_TW
Catalan	/ca
German	/de
Spanish	/es
Croatian	/hr
Hebrew	/iw
Dutch	/nl
Portuguese	/pt
Russian	/ru
Swedish	/sv
Ukrainian	/uk
Czech	/cs
Greek	/el
Finnish	/fi
Hungarian	/hu
Japanese	/ja
Norwegian	/no
Brazilian Portuguese	/pt_BR
Slovak	/sk
Thai	/th
Simplified Chinese	/zh

Supported languages

The different user interfaces used to install, configure, and use IBM WebSphere Portal Express might not be available in all languages. Refer to the following table for the language codes and to see which areas of the product are available in your language. If your language is not supported currently, you might see the English version of the user interface.

Table 667. Supported languages for the WebSphere Portal user interfaces

Language	Language code	First Steps	Launchpad	Installation Manager	Configuration Wizard	WebSphere Portal Express
Arabic	ar	Available	Available	Available	Available	Available
Catalan	ca	Available	Available	Not available	Available	Available
Czech	cs	Available	Available	Available	Available	Available
Danish	da	Not available	Not available	Not available	Not available	Available

Table 667. Supported languages for the WebSphere Portal user interfaces (continued)

Language	Language code	First Steps	Launchpad	Installation Manager	Configuration Wizard	WebSphere Portal Express
Dutch	nl	Not available	Not available	Not available	Not available	Available
English	en	Available	Available	Available	Available	Available
Finnish	fi	Not available	Not available	Not available	Not available	Available
French	fr	Available	Available	Available	Available	Available
German	de	Available	Available	Available	Available	Available
Greek	el	Available	Available	Available	Available	Available
Hebrew	iw	Not available	Not available	Not available	Not available	Available
Croatian	hr	Available	Available	Available	Available	Available
Hungarian	hu	Available	Available	Available	Available	Available
Italian	it	Available	Available	Available	Available	Available
Japanese	ja	Available	Available	Available	Available	Available
Kazakh	kk	Available	Available	Not available	Available	Available
Korean	ko	Available	Available	Available	Available	Available
Norwegian	no	Not available	Not available	Not available	Not available	Available
Polish	pl	Available	Available	Available	Available	Available
Portuguese	pt	Available	Available	Not available	Available	Available
Brazilian Portuguese	pt_BR	Available	Available	Available	Available	Available
Romanian	ro	Available	Available	Not available	Available	Available
Russian	ru	Available	Available	Available	Available	Available
Slovak	sk	Available	Available	Available	Available	Available
Slovenian	sl	Available	Available	Available	Available	Available
Spanish	es	Available	Available	Available	Available	Available
Simplified Chinese	zh	Available	Available	Available	Available	Available
Traditional Chinese	zh_TW	Available	Available	Available	Available	Available
Swedish	sv	Not available	Not available	Not available	Not available	Available
Thai	th	Available	Available	Available	Available	Available
Turkish	tr	Available	Available	Available	Available	Available
Ukrainian	uk	Available	Available	Not available	Available	Available

Updates using ReleaseBuilder

After setting up your initial staging and production servers, you can use ReleaseBuilder to make updates to your production server.

"ReleaseBuilder"
To generate or stage follow-on releases of IBM WebSphere Portal Express portals, configurations, and artifacts need to be moved between systems. ReleaseBuilder enables management of release configurations independent of user configurations.

"Making updates with ReleaseBuilder" on page 3625
You can use ReleaseBuilder to compare the XML configuration files that describe your staging server (REV1) and your updated staging server (REV2). You can also use it to create an XML configuration file that contains the differences between the two servers. You can then use this output file to import only the differences from your staging server (REV2) onto the production server. Features that are unchanged from (REV1) on the staging server (REV2) are not affected by the import.

"Reference: ReleaseBuilder command syntax" on page 3628
The ReleaseBuilder command file is in the *wp_profile_root*`/PortalServer/bin` directory and has the following syntax:

ReleaseBuilder

To generate or stage follow-on releases of IBM WebSphere Portal Express portals, configurations, and artifacts need to be moved between systems. ReleaseBuilder enables management of release configurations independent of user configurations.

Release configuration data are exported to XML files that can be imported using the XML configuration interface (XmlAccess). Using ReleaseBuilder it is possible to stage release configurations between two portals. This allows you to track which configuration entities were removed, added, or changed compared to the previous release generated from a given portal and to apply these differential updates to another portal. Detecting the differences between one configuration and another of the same portal server creates differential updates. A third configuration or "diff", generated by ReleaseBuilder, represents the changes made between the two configurations. The third configuration can be used to apply not only addition and update modifications but also deletions to the target server. This allows two portal servers, for example, a staging server and a production server, to remain in synch. ReleaseBuilder is designed to eliminate the need to generate complete XmlAccess exports to move a partial configuration or to manually create XML response files to export a partial configuration. ReleaseBuilder also helps to prevent the problem of configuration bloat on the target server.

For staging virtual portals, ReleaseBuilder supports a virtual portal mode that allows the generation of difference configurations for virtual portal scoped resources only. This mode allows you to stage virtual portals.

Note: ReleaseBuilder is a configuration management tool. Do not use it for migration purposes.

Performance benefits

Massively parallel portal configuration tasks, for example hundreds of administrators working in parallel, can affect the user experience and portal performance. Such tasks can be distributed to independent portal systems.

ReleaseBuilder, together with the XML configuration interface, allows you to integrate the resulting configurations.

Usage notes

- ReleaseBuilder is installed when WebSphere Portal Express is installed along with the XML configuration interface. You can run ReleaseBuilder on the production server, although it can impact the WebSphere Portal Express performance.
- Unlike the XML configuration interface, ReleaseBuilder does not interact with the portal server runtime. You should run ReleaseBuilder on a separate, standalone machine. This system can be the staging system or a completely separate system where WebSphere Portal Express is installed.
- To maximize use of system resources, WebSphere Portal Express should not be running when executing ReleaseBuilder.

Making updates with ReleaseBuilder

You can use ReleaseBuilder to compare the XML configuration files that describe your staging server (REV1) and your updated staging server (REV2). You can also use it to create an XML configuration file that contains the differences between the two servers. You can then use this output file to import only the differences from your staging server (REV2) onto the production server. Features that are unchanged from (REV1) on the staging server (REV2) are not affected by the import.

Before you begin

Ensure that your development, test, and production environments are configured to allow all of the required artifacts and the configuration to be moved.

ReleaseBuilder is preferably run on an integration or staging server. Running ReleaseBuilder on the production server is not advised because ReleaseBuilder uses resources and affects portal services to users.

ReleaseBuilder uses XML configuration files to create an XML configuration file of the differences between the two servers. This XML configuration file is used to transfer the new portal configuration from your staging server to your production server. To export the configurations of the servers, use the XML configuration interface.

About this task

The XML files from your staging server (REV1) and from your updated staging server (REV2) refer to two exports that are taken from the SAME portal server. Building a release means to generate an XML file that contains the same modifications that are made to the staging server between REV1 and REV2. ReleaseBuilder is not designed to determine the differences between or changes that are made to two separate Portal servers.

Note: The following instructions describe the building of a release in installations, which do not use virtual portals. For instructions for staging virtual portals, refer to the topic about Building a Release for virtual portal installations.

Procedure

1. Complete the following steps to export the staging server configuration REV1:

Remember: Export the entire portal configuration REV1 of the staging server; do not include users, users' access control, or any other user configurations.

a. On the staging server change to the *wp_profile_root*/PortalServer/bin directory. This directory contains the portal tools.

b. Use the XML configuration interface to export the staging server REV1 configuration. A sample file is available, ExportRelease.xml.

   - HP-UX Linux : `./xmlaccess.sh -in ExportRelease.xml -user wpsadmin_user_ID -password wpsadmin_pwd -url "http://stagingserver.example.com:port/wps/config" -out stagingserverREV1_config.xml`
   - IBM i: `xmlaccess.sh -in ExportRelease.xml -user wpsadmin_user_ID -password wpsadmin_pwd -url "http://stagingserver.example.com:port/wps/config" -out stagingserverREV1_config.xml`
   - Windows: `xmlaccess.bat -in ExportRelease.xml -user wpsadmin_user_ID -password wpsadmin_pwd -url "http://stagingserver.example.com:port/wps/config" -out stagingserverREV1_config.xml`

   The exported configuration is stored in the stagingserverREV1_config.xml file.

2. Develop and test new functions and portlets on the staging server. This phase is where you add or delete functions. This phase can last for a long time. Ensure that the staging server is fully tested and the portal is ready.

3. Complete the following steps to export the staging server configuration REV2:

   Remember: Export the entire portal configuration REV2 of the staging server; do not include users, users' access control, or any other user configurations.

   a. On the staging server change to the *wp_profile_root*/PortalServer/bin directory. This directory contains the portal tools.

   b. Run the following task to export the staging server configuration REV2:

      Note: Use the XML configuration interface and the provided sample file that is called ExportRelease.xml.

      - HP-UX Linux : `./xmlaccess.sh -in ExportRelease.xml -user wpsadmin_user_ID -password wpsadmin_pwd -url "http://stagingserver.example.com:port/wps/config" -out stagingserverREV2_config.xml`
      - Windows: `xmlaccess.bat -in ExportRelease.xml -user wpsadmin_user_ID -password wpsadmin_pwd -url "http://stagingserver.example.com:port/wps/config" -out stagingserverREV2_config.xml`
      - IBM i: `xmlaccess.sh -in ExportRelease.xml -user wpsadmin_user_ID -password wpsadmin_pwd -url "http://stagingserver.example.com:port/wps/config" -out stagingserverREV2_config.xml`

      The exported configuration is stored in the stagingserverREV2_config.xml file.

4. If you installed extra portlets or applications, copy the necessary WAR files from the staging server to the *wp_profile_root*/PortalServer/deployed/archive installation directory on the production server.

   Note: The deployed/archive directory is always in the original *wp_profile_root* installation path, even if the production server is using an extra profile that is created after installation.

Note: As Windows limits the maximum path length to 260 characters, the name of the WAR file must be 25 characters or less. Installing a WAR file with a name that is more than 25 characters results in an error.

5. Complete the following steps to generate the differences between staging server configurations REV1 and REV2:

   Note: Complete these steps on the staging server and not on the production server. Computing a release difference produces a high load on the system and uses much memory for large releases. Therefore, do not complete them on the production system.

   a. Optional: Stop the portal server on the staging system. This option frees resources for computing the release difference.
   b. On the server where you just stopped the portal server, change to the *wp_profile_root*/PortalServer/bin directory.
   c. To generate the differences file containing the additions and deletions configuration file, enter one of the following commands:

Table 668. Differences file commands

Operating system	Command
Linux	./releasebuilder.sh -inOld *stagingserverREV1_config.xml* -inNew *stagingserverREV2_config.xml* -out *outputfile.xml*
Windows:	releasebuilder.bat -inOld *stagingserverREV1_config.xml* -inNew *stagingserverREV2_config.xml* -out *outputfile.xml*
IBM i:	releasebuilder.sh -inOld *stagingserverREV1_config.xml* -inNew *stagingserverREV2_config.xml* -out *outputfile.xml*

   The resulting output configuration file contains the additions and deletions to be imported onto your production server.

   d. Optional: Restart the portal server on the staging system, if you stopped it before.

6. Use the *outputfile*.xml, which contains the differences between REV1 and REV2 portal server to import these differences onto the production server.

Table 669. Task to import differences onto the production server

Operating system	Command
Linux	./xmlaccess.sh -in outputfile.xml -user *wpsadmin_user_ID* -password *wpsadmin_pwd* -url "http://*productionserver.example.com:port*/wps/config"
Windows:	xmlaccess.bat -in outputfile.xml -user *wpsadmin_user_ID* -password *wpsadmin_pwd* -url "http://*productionserver.example.com:port*/wps/config"

Table 669. Task to import differences onto the production server (continued)

Operating system	Command
IBM i:	`xmlaccess.sh -in outputfile.xml -user` *`wpsadmin_user_ID`* `-password` *`wpsadmin_pwd`* `-url "http://` *`productionserver.example.com:port`*`/wps/config"`

Notes:

a. If portlet parameters are deleted from a configuration, the output script that is generated by ReleaseBuilder does not remove those parameters on the target system.

b. XML files that are generated by ReleaseBuilder do not have any transaction levels set. To set a transaction level, edit the XML file that is generated by ReleaseBuilder and add the transaction level to the XML file.

Related reference:

"Sample XML configuration files" on page 1111
Sample files are provided for your reference to help illustrate how to use XML configuration for different portal configuration purposes. Before you use them, read the other topics about the XML configuration interface carefully. Many of the samples need to be modified with valid page or user name before they can be used.

Reference: ReleaseBuilder command syntax

The ReleaseBuilder command file is in the *wp_profile_root*`/PortalServer/bin` directory and has the following syntax:

NAME

`releasebuilder` - generate different configurations for a single staging server. The production server uses the configuration.

SYNOPSIS

`releasebuilder -inOld` *input file oldReleaseConfiguration* **`-inNew`** *input file newReleaseConfiguration* **`-out`** *output file* **`-virtualPortalMode`**

OPTIONS

The following options are supported:

- **`inOld`** *input file oldReleaseConfiguration*:

  Required. This option is used to specify the first of the two XML configuration files to compare. The *input file oldReleaseConfiguration* parameter must be an absolute or relative path of an accessible XML configuration file that is previously generated with the XML configuration interface command.

- **`inNew`** *input file newReleaseConfiguration*:

  Required. This option is used to specify the second of the two XML configuration files to compare. The *input file newReleaseConfiguration* parameter must be an absolute or relative path of an accessible XML configuration file that is previously generated with the XML configuration interface command.

- **`out`** *output file*:

  This option is used to specify the file in which the differential output is written. If not specified, the differential output is written to the System console. The *output file* parameter must be an absolute or relative path of an

accessible XML configuration file. If it does not exist, it is created. If it does exist, the existing information in the file is overwritten.

- **`virtualPortalMode`**:

  Use this option to specify that only virtual portal scoped resources are included in the configuration. This option enables the processing of a specific virtual portal. If you do not specify this parameter, the differential output includes unscoped resources as well.

- **`disableLogFile`**:

  Use this option if you do not want ReleaseBuilder to create a log file.

- **`logFileLocation`** *log output file*:

  ReleaseBuilder writes the log file to the specified location.

- **`debug`**

  Use this option to help you debug any errors that occur.

Staging Personalization rules to production

Use the steps in this file to move Personalization rules from a staging system to a production system. There are a number of methods for moving rules between servers, each suitable for different situations.

Choose one of the following methods to move rules:

Method: Export from source then import into destination

Steps	Advantages	Disadvantages	Users
1. Use the **Export** button in the Personalization Navigator portlet on the source to export a nodes file. 2. Use the **Import** button in the Personalization Navigator portlet on the target to import that file.	• This method is the easiest way to move rules. • Uses a familiar export and import paradigm.	• Cannot be scripted. • Requires Personalization Navigator portlet to be installed on the target server.	• Development teams. • Quick, ad-hoc changes in small deployments and test environments.

Method: Publish using the Personalization Navigator portlet

Steps	Advantages	Disadvantages	Users
Use the Publish menu options in the Personalization Navigator portlet to publish the entire workspace or to selectively publish	• Easy and quick to use. Once a publish server is configured, you can publish rules in two clicks. • This method does not require rules to be saved on the file system. • If publishing the entire workspace with smart delete, you can ensure two work spaces are the same.	• No intermediate file is produced by the process, so there is no record of what was published other than log files. • This approach is driven from a graphical user interface, so it is not scriptable.	Business users with rule authoring responsibilities.

Method: Export from the source and then publish into the destination

Steps	Advantages	Disadvantages	Users
1. Use the export button in the Personalization Navigator portlet on the source to export a nodes file. 2. Use the pznload command line utility (**pznload.sh** and **pznload.bat**) to publish the nodes file that you exported to the target.	• This method can an be scripted. • Allows your changes to be tracked, controlled, and the earlier versions to be reverted by maintaining copying of the nodes files and rerunning a script.	Requires use of a command line interface.	• Administrators • Moving between staging and production environments

Notes:

- If the rules are referenced by Personalization components, ensure the Personalization components are published and syndicated in Web Content Manager.
- If the rules are referenced in pages or portlets, move the page and portlet definitions with XML Access or Release Builder. This is related to attribute based administration.

Related concepts:

"Publishing personalization rules overview" on page 2298
WebSphere Portal Express Personalization sends published rules across HTTP to a servlet which resides on each personalization server. This servlet can receive publishing data or initiate new publishing jobs. When a user begins a publishing job from the personalization authoring environment, the local servlet is provided with the set of information necessary to complete the job. The local servlet contacts the destination endpoint servlet (which could be the same servlet) and sends its data to it. The destination servlet reports success or failure.

CF04 and earlier: Using friendly URLs without state information

By default, WebSphere Portal Express URLs include navigational state information. If you configure pages for friendly URLs, the portal appends the state information to the friendly URLs. Some scenarios require short and fully human readable URLs that omit the state information. For such scenarios, you can configure friendly URLs so that the portal does not show that state information.

About this task

The state information is an encoded aggregation of the navigational state of the portal, the page, and its components, for example, the portlets on the current page:

- The portal state includes page selection, expansions, label mapping, and action targets.
- The portlet state includes render parameters, window state, and portlet mode.

The representation of the navigational state within the URL enables characteristics of dynamic websites, such as usage of bookmarks or the **Back** button. For example, users can bookmark a page and later return to the exact same state of that page.

Some scenarios require short and fully human readable URLs that omit the state information. Examples:

- You do not want the URL to make the impression that it references dynamic content.
- You want the URL to contain only information that a human person can read and interpret.
- You want the URL to easily fit into the address field of the web browser.
- Internet search engines expect static URLs that reference only one resource or web page for the time that the page exists
- Internet search engines prefer short and friendly URLs.

For such scenarios you configure WebSphere Portal Express as follows:

- You configure themes to always display only short friendly URLs without the encoded navigational state.
- You configure pages that use that theme to display friendly URLs.

The configuration applies to all pages that use that theme and that are configured to display friendly URLs.

Notes:

- You can create friendly URLs for portal pages.
- If you configure your portal to show stateless friendly URLs, you gain improved URL readability at the cost of losing the state functionality. Example consequences are as follows:

- Portal URLs always point to the default state of a page, as they do not contain the state information.
- If a user clicks the **Back** button, or refreshes a page by clicking the **Refresh** button or the page title, the page moves back into the default View mode.
- If a user views a page, and then creates a bookmark, clicking the bookmark later opens the page in the default View mode.
- Stateless friendly URLs do not contain the usual information about the language of the page. The portal determines the language for the page by the following order:
  1. First, the portal looks for the user preference.
  2. If the user preference is not set, the portal looks for the preferred language that is set in the browser. If the page is a public page, the user is an anonymous user. In this case, the portal also looks for the preferred language that is set in the browser.
  3. If the portal cannot determine a preferred language setting for the portal or the browser, it applies the default language that is defined for the portal.

  For information about how to present language-specific portal pages with stateless friendly URLs, read the information at the end of the procedure that follows here.

Procedure

1. In the portal WP Configuration Service, set the custom property `friendly.redirect.enabled` to the value `false`. You do this step by editing the Resource Environment Provider (REP) WPConfigService in the WebSphere Integrated Solutions Console. If the property is not listed there, add it and set it to `false`. For information about this property and how to set it see the topics about the portal Configuration Service and Setting service configuration properties.
2. In the theme, that you want configure, for short stateless URLs, set the parameter `com.ibm.portal.theme.hasBaseURL` to `true`. You can update the theme parameter by using the XML configuration interface. Here is an example XML script:

```
<?xml version="1.0" encoding="UTF-8"?>

<request
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd"
   type="update">

   <!-- This sample sets the hasBaseURL Tag in the Portal 8.5 Theme. -->
   <portal action="locate">
      <theme action="update" uniquename="ibm.portal.85Theme" >
         <parameter name="com.ibm.portal.theme.hasBaseURL"
                    type="string" update="set">true</parameter>
      </theme>
   </portal>
</request>
```

3. Make sure that all generated URLs in the theme do not include the navigational state. In the default theme, you can do this step by modifying the `navigation.jsp` and `sideNavigation.jsp` files. To change the file `navigation.jsp`, proceed as follows:
   a. Change to the directory for the file `navigation.jsp`. You must update two copies of this file, one each in the following two directory locations.

- *PortalServer_root*\theme\wp.theme.themes\default85\installedApps\DefaultTheme85.ear\DefaultTheme85.war\themes\html\dynamicSpots
- *PortalServer_root*\theme\wp.theme.modules\webapp\installedApps\ThemeModules.ear\ThemeModules.war\themes\html\dynamicSpots

b. Open the file navigation.jsp with an editor.

c. Search for the string <a href="?uri=nm:oid:${wp.identification[node]}" class="wpthemeLeft${titleClass}"> and change it to the following code snippet:

```
<portal-navigation:urlGeneration contentNode="${wp.identification[node]}" keepNavigationalState="false">
<a href="<%wpsURL.write(out);%>" class="wpthemeLeft${titleClass}">
```

d. Locate the first tag after the updated string <a href="<%wpsURL.write(out);%>" and change it to the following code snippet:

```
</a>
</portal-navigation:urlGeneration>
```

To change the file sideNavigation.jsp, proceed as follows:

a. Change to the directory *PortalServer_root*\theme\wp.theme.themes\default85\installedApps\DefaultTheme85.ear\DefaultTheme85.war\themes\html\dynamicSpots.

b. Open the file sideNavigation.jsp with an editor.

c. Search for the string <a href="?uri=nm:oid:${wp.identification[node]}" class="<c:if test="${isSelectedNode}"> wpthemeSelected</c:if>${titleClass}"> and change it to the following code snippet:

```
<portal-navigation:urlGeneration contentNode="${wp.identification[node]}" keepNavigationalState="false">
<a href="<%wpsURL.write(out);%>" class="<c:if test="${isSelectedNode}">wpthemeSelected</c:if>${titleClass}">
```

d. Locate the first tag after the updated string <a href="<%wpsURL.write(out);%>" and change it to the following code snippet:

```
</a>
</portal-navigation:urlGeneration>
```

4. Edit the mobileNavigation.jsp file.

a. Change to the directory *PortalServer_root*\theme\wp.theme.themes\default85\installedApps\DefaultTheme85.ear\DefaultTheme85.war\themes\html\dynamicSpots.

b. Open the file mobileNavigation.jsp with an editor.

c. Search for the string <a href="?uri=nm:oid:${nodeID}"> and change it to the following code snippet:

```
<portal-navigation:urlGeneration contentNode="${nodeID}" keepNavigationalState="false">
<a href="<%wpsURL.write(out);%>"
```

d. Locate the first tag after the updated string <a href="<%wpsURL.write(out);%>" and change it to the following code snippet:

```
</a>
</portal-navigation:urlGeneration>
```

5. Optional: For IBM Web Content Manager: You might want the IBM Web Content Manager Rendering portlet to also display the friendly and stateless URLs. In this case, implement a plug-in that translates the IBM Web Content Manager URLs into the required custom format. For instructions and sample code for such a plug-in, see *Example 2, Generate a friendly URL for web content*.
6. Define friendly URL names for pages as required. For information about how to do this step, read *Using friendly URLs*.
7. Optional: You might want to present language-specific portal pages with stateless friendly URLs to your site visitors. In this case, structure your portal site to reflect which pages or pages are targeted for specific countries or regions. For example, you can create a node for a specific page, and then create language-specific child pages under that node. Example: In the node home, you create pages in English, French, and German. You can then give your site visitors the appropriate one of the following friendly URLs:

```
http://www.myco.com/wps/home/en/shop
http://www.myco.com/wps/home/fr/shop
http://www.myco.com/wps/home/de/shop
```

Results

The portal now no longer displays the state information with the URLs.

Terms of use

Understand the terms and conditions before you use this publication.

Permissions for the use of publications is granted subject to the following terms and conditions.

Personal Use: You may reproduce these publications for your personal, non commercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial Use: You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Notices

This information was developed for products and services offered in the U.S.A. This material may be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product,

program, or service that does not infringe any IBM intellectual property rights may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Glossary

This glossary includes terms and definitions for IBM WebSphere Portal Express.

The following cross-references are used in this glossary:

1. See refers the reader from a term to a preferred synonym, or from an acronym or abbreviation to the defined full form.
2. See also refers the reader to a related or contrasting term.

To view glossaries for other IBM products, go to www.ibm.com/software/globalization/terminology.

A

access control. In computer security, the process of ensuring that users can access only those resources of a computer system for which they are authorized.

aggregation. The structured collection of data objects for subsequent presentation within a portal.

Ajax. A design approach and a set of techniques for delivering rich Internet applications (RIAs) using open web formats, for example, HTML, CSS and JavaScript; and rendering using a browser engine.

Ajax portlet. A normal server side portlet that uses lots of JavaScript and Ajax technologies and less Java and Java server pages.

anonymous user. A user who does not use a valid user ID and password to log into a site.

applet. A program that performs a specific task and is usually portable between operating systems. Often written in Java, applets can be downloaded from the Internet and run in a web browser.

application server. A server program in a distributed network that provides the execution environment for an application program.

Asynchronous JavaScript and XML. See Ajax.

authenticated user. A portal user who has logged in to the portal with a valid account (user ID and password). Authenticated users have access to all public places.

authentication. A security service that provides proof that a user of a computer system is genuinely who that person claims to be. Common mechanisms for implementing this service are passwords and digital signatures. Authentication is distinct from authorization; authentication is not concerned with granting or denying access to system resources.

authorization. The process of granting a user, system, or process either complete or restricted access to an object, resource, or function.

B

B2B. See business-to-business.

B2C. See business-to-consumer.

B2E. See business-to-employee.

bind. To establish a connection between software components on a network using an agreed-to protocol. In web services, the bind operation occurs when the service requestor invokes or initiates an interaction with the service at run time using the binding details in the service description to locate, contact, and invoke the service.

bookmark. A customizable, graphical link to databases, views, documents, web pages, and newsgroups.

business-to-business (B2B). Refers to Internet applications that exchange information or run transactions between businesses.

business-to-consumer (B2C). Refers to the subset of Internet applications that exchange information or run transactions between businesses and consumers.

business-to-employee (B2E). A business model that supports electronic communications between a business and its employees.

C

CA. See certificate authority.

card. WML document that provides user-interface and navigational settings to display content on mobile devices.

cascading style sheet (CSS). A file that defines a hierarchical set of style rules for controlling the rendering of HTML or XML files in browsers, viewers, or in print.

certificate authority (CA). A trusted third-party organization or company that issues the digital certificates. The certificate authority typically verifies the identity of the individuals who are granted the unique certificate.

client side aggregation (CSA). Aggregation based on JavaScript and XSLT transformations that are executed on the client.

cloud application. An application that is extended to be accessible through the Internet. Cloud applications use large data centers and powerful servers that host web applications and web services.

cloud computing. A computing platform where users can have access to applications or compute resources, as services, from anywhere through their connected devices. A simplified user interface and application programming interface (API) makes the infrastructure supporting such services transparent to users.

collaboration. The ability to connect customers, employees, or business partners to the people and processes in a business or organization, in order to facilitate improved decision-making. Collaboration involves two or more individuals with complementary skills interacting together to resolve a business problem.

collaborative components. UI-neutral API methods and tag libraries that allow developers to add IBM Lotus collaborative functionality to their portlets.

collaborative filtering. Personalization technology that calculates the similarity between users based on the behaviors of a number of other people and uses that information to make recommendations for the current user.

collaborative portal. A highly personalized desktop-to-web tool designed for specific audiences and communities of users that organizes information, applications, and services for effective community building at the corporate level and for personal use by individuals.

concrete portlet. A logical representation of a portlet object distinguished by a unique configuration parameter (PortletSettings).

confirmable methods. Interface methods that exist on each modifiable interface of a portal resource that allow users to determine whether a modification can be performed or not.

connector. A servlet that provides a portlet access to external sources of content, for example, a news feed from a website of a local television station.

consumer portal. A portal that uses the portlets that a producer portal provides.

content item. Web page content stored in the form of elements, equivalent to a web page in a traditional website. The look and feel of a content item when displayed in a website will depend on what authoring template is used to create the content item and what presentation template is used to display the content.

content management. Software designed to help businesses manage and distribute content from diverse sources.

content partner. See IBM content partner.

content provider. A source for content that can be incorporated into a portal page as a portlet.

controller. A modifiable instance of a portal model which allows to modify the topology of the model, create and delete resources, and create modifiable instances of existing resources.

cooperative portlets. Two or more portlets on the same web page that interact by sharing information.

creation context. A context that defines immutable properties of a resource that you can create

CSS. See cascading style sheet.

D

DB2. A family of IBM licensed programs for relational database management.

deck. An XML document that contains a collection of WML cards.

default portal page. The page that displays to a user at initial portal deployment and before the user completes enrollment. Sometimes used as a synonym for home page.

default public place. A place whose membership automatically includes all portal users and which appears in the Places selector for every user. A user is always a member of this place.

derived page. One or more child pages that have a shared layout that is inherited from the properties of the parent page.

differential page rendering (DPR). Renders only those parts of a portal page that were affected by the a user interaction.

document type definition (DTD). The rules that specify the structure for a particular class of SGML or XML documents. The DTD defines the structure with elements, attributes, and notations, and it establishes constraints for how each element, attribute, and notation can be used within the particular class of documents.

DTD. See document type definition.

dynamic layout. Standard portal layout that consists of rows or columns and is persisted in the database.

E

ECM. See Enterprise Content Management

embedded static page. A static page that is rendered in the content area of the portal.

enrollment. The process of entering and saving user or user group information in a portal.

Enterprise Content Management (ECM). Software and tools designed to enable companies to manage content and documents, optimize business processes, and enable compliance with an integrated infrastructure.

Enterprise Information Portal. Software developed by IBM that provides tools for advanced searching, and content customization and summarization.

Extensible Markup Language (XML). A standard metalanguage for defining markup languages that is based on Standard Generalized Markup Language (SGML).

Extensible Stylesheet Language (XSL). A language for specifying style sheets for XML documents. Extensible Stylesheet Language Transformation (XSLT) is used with XSL to describe how an XML document is transformed into another document.

F

federated search. A search capability that enables searches across multiple search services and returns a consolidated list of search results.

G

group.

1. A collection of users who can share access authorities for protected resources.
2. In places, two or more people who are grouped for membership in a place.

governance. The decision making processes in the administration of an organization. The rights and responsibilities of these processes are typically shared among the organization's participants, especially the management and stakeholders.

governance life cycle. A life cycle that represents the states and transitions that can exist in SOA deployment.

governance processes. A process that ensures that compliance and operational polices are enforced, and that change occurs in a controlled fashion and with appropriate authority as envisioned by the business design.

H

helper file. A properties file that WebSphere Portal Express provides to ensure that users specify the correct information that is needed to complete different types of configuration tasks such as configuring an LDAP user registry or a database user registry.

home page. The top-level web page of a portal.

HTML. See Hypertext Markup Language.

HTTP. See Hypertext Transfer Protocol.

HTTP over SSL (HTTPS). A web protocol for secure transactions that encrypts and decrypts user page requests and pages returned by the web server.

HTTPS. See HTTP over SSL.

Hypertext Markup Language (HTML). A markup language that conforms to the Standard Generalized Markup Language (SGML) standard and was designed primarily to support the online display of textual and graphical information, including hypertext links.

Hypertext Transfer Protocol (HTTP). An Internet protocol that is used to transfer and display hypertext and XML documents on the web.

I

i-mode. An Internet service for wireless devices.

IBM content partner (content partner). IBM partner that provides syndicated content for portals.

integrity. In computer security, assurance that the information that arrives at a destination is the same as the information that was sent.

iwidget. An open-source specification that allows for seamless interoperability across various platforms and products.

J

JavaScript. A web scripting language that is used in both browsers and web servers. (Sun)

JavaScript Object Notation. A lightweight data-interchange format that is based on the object-literal notation of JavaScript. JSON is programming-language neutral but uses conventions from languages that include C, C++, C#, Java, JavaScript, Perl, Python.

Jetspeed. The open-source portal that is part of the Jakarta project by Apache.

L

label. A node in a portal that cannot contain any content, but can contain other nodes. Labels are used primarily to group nodes in the navigation tree.

lazy application. An application whose initialization is deferred until first use.

LDAP. See Lightweight Directory Access Protocol.

LDAP directory. A type of repository that stores information on people, organizations, and other resources and that is accessed using the LDAP protocol. The entries in the repository are organized into a hierarchical structure, and in some cases the hierarchical structure reflects the structure or geography of an organization.

light mode. An operation method that enhances portal performance by improving start-up time and reducing memory consumption in production environments.

Lightweight Directory Access Protocol (LDAP). An open protocol that uses TCP/IP to provide access to directories that support an X.500 model and that does not incur the resource requirements of the more complex X.500 Directory Access Protocol (DAP). For example, LDAP can be used to locate people, organizations, and other resources in an Internet or intranet directory.

load balancing. The monitoring of application servers and management of the workload on servers. If one server exceeds its workload, requests are forwarded to another server with more capacity.

M

mandatory place. A shared place, either a public place or a restricted place, in which all portal users must be members. Only portal administrators can designate a shared place to be a mandatory place. Because membership is automatic and required, portal users cannot join or leave mandatory places.

membership. The state of being a portal user and a place member. Membership in the portal is controlled by the administrator during the installation and set up of portal servers. Membership in places is controlled by a place manager, who determines the level of access for each place member: participant, place designer, or place manager.

mashup. A graphical interface that features two or more reusable web applications (widgets) presenting seemingly disparate data in an understandable combination for a specific purpose.

meta search. A search across one or more search engines. A meta search engine provides a meaningful subset of search functionality through an abstraction layer that is generic enough to support a wide variety of search services.

messaging. A method for communication between programs. Messaging can be synchronous or independent of time.

middleware. Software that acts as an intermediate layer between applications or between client and server. It is used most often to support complex, distributed applications in heterogeneous environments.

model view controller (MVC). A software architecture that separates the components of the application: the model represents the business logic or data; the view represents the user interface; and the controller manages user input or, in some cases, the application flow.

modifiable. An interface for modifying portal resources that exist in the read only model.

MVC. See model view controller.

N

News Industry Text Format (NITF). An XML-based format that defines the structure and content of news articles.

News Markup Language (NewsML). An XML-based format for publishing news-related information.

NewsML. See News Markup Language.

NITF. See News Industry Text Format.

node. A logical group of managed servers.

O

OCS channel. See open content syndication channel.

open content syndication channel (OCS channel). An XML-based format for syndicated content.

P

page. A node in a portal that can contain content in addition to labels and other pages. Pages can contain child nodes, column containers, row containers, and portlets.

participant. A member of a portal place who can visit and use the place. By default, all portal users are participants in public places.

people awareness. The collaboration feature that provides access to people from various contexts. People awareness lets you see references to people and contact people by name through the Sametime online status indicator. Throughout the portal, wherever you see the name of a person, you can view the person's online status, send email, initiate a chat, or share an application via an electronic meeting.

people finder. A portlet that enables users to find, view information about, and contact individuals in their organization. Administrators can configure people finder to display information details such as email address, job title, and location.

person. An individual authenticated by the portal and having a person record in one or more corporate directories. Persons can be members of places, public groups within the organization's corporate directory, or personal groups that a user defines.

person card. An interface that displays information about a registered user such as phone number and online status (if Sametime is enabled), and additional details typically found on a business card. Available actions let you view the person's complete profile and, depending on how the portal is configured, send email, chat, and link to Lotus Connections features such as Communities, Activities, and Blogs.

person link. A reference to a person's name or a group name that appears with the Sametime online status indicator. The reference lets you view the person's online status, send an email, start a chat, or share an application using an electronic meeting, among other actions shown on the person link menu.

personal group. In Sametime Connect, a group of people designated by the user as a group. A user can choose individuals from the public Directory (public group) and create personal groups, which are then stored locally. Users can add and remove people from a personal group, whereas the membership of the public group is defined by the owner of the public Directory.

personalization. The process of enabling information to be targeted to specific users based on business rules and user profile information.

pervasive computing. The use of a computing infrastructure that supports information appliances from which users can access a broad range of network-based services, including Internet-based e-commerce services.

place designer. A member of a place who can edit place layout and bookmarks.

place manager. A member of a place who can edit place membership, layout, and bookmarks.

place member. A individual or group who has joined or been granted access to a place. Place members have three levels of access to a place: manager, designer, and participant.

place template. A format for use in creating a place. The portal provides a set of default templates for creating various types of places. Portal administrators may allow users to create, modify, and delete new templates.

policy. A set of rules and actions that are required to be performed when certain events or status conditions occur in an environment. Policies are implemented using the IPL.

port. An end point for communication between applications, generally referring to a logical connection. A port provides queues for sending and receiving data. Each port has a port number for identification.

port type. An element in a Web Services Description Language (WSDL) document that comprises a set of abstract operations, each of which refers to input and output messages that are supported by the web service.

portal. A single, secure point of access to diverse information, applications, and people that can be customized and personalized.

portal administration. The place where portal administrators set and maintain basic collaboration permissions, place records, place membership records, and server settings for companion products for advanced collaboration.

portal artifacts. Stored in the portal file system. All deliverables of software development are usually artifacts (otherwise referred to as software components).

portal configuration. The Portal Configuration is stored in the portal configuration database. It consists of configuration entities. Each portal resource is represented by one portal configuration entity in the portal database.

portal extension artifacts. Artifacts that belong to components that are installed together with the portal but are not core portal components.

portal farm. A series of identically configured, stand-alone portal server instances that offer a way to maintain a highly scalable and highly available server environment.

portal member. An individual or group who has a user record in the portal directory (LDAP or other directory) and can log in to the portal.

portal solution release. The solution that is developed by the customer and is based on WebSphere Portal. The solution consists of portal configurations, portal artifacts, and portal extension artifacts and is shared between multiple users.

portal theme. The style element that gives a place a particular look. The portal provides several themes, similar to virtual wallpaper, which can be chosen when creating a place.

portlet. A reusable web module that runs on a portal server. Portlets have predefined roles such as retrieving news headlines, searching a database, or displaying a calendar.

portlet API. The set of interfaces and methods that are used by Java programs running within the portal server environment to obtain services.

portlet application. A collection of related portlets that can share resources with one another.

portlet container. A column or row that is used to arrange the layout of a portlet or other container on a page.

portlet control. A portlet registry setting that renders the outer frame for a portlet.

portlet framework. The set of classes and interfaces that support Java programs running within the portal server environment.

portlet palette. A web module that enables users to browse available portlets organized by category, search for individual portlets, and add them to a portal page by dragging to the desired location.

pre-rendered site. A complete website saved to disk in HTML that can be used as a live site and displayed to users using either Web Content Manager or a web server.

presentation template. A template that determines the structure of each web page in the site and which elements and components are displayed on each page. HTML defines the default properties and layout of the template.

producer definition. A set of interfaces that are defined for the producer portal. The producer definition can include the producer service description, the producer portal URL, and the security setup.

producer portal. A portal that provides portlets as a service so that other portals, called consumer portals, can use the portlets and make the portlets available to their users.

property extension database. A database that is used to store additional attributes that cannot be stored in the LDAP user registry.

provisioning. The process of setting up and maintaining a user's access to a system.

PSTN. See public switched telephone network.

public group. A group of individuals, known to all portal users, that the administrator has created or that exists in the organization's corporate directory. Only administrators can modify and manage public groups.

public place. A shared place that is open to all portal users. The person who creates the place (and who automatically becomes the place manager) designates it as a public place during place creation.

public switched telephone network (PSTN). A communications common carrier network that provides voice and data communications services over switched lines.

pure server side portlet. A normal server side portlet that uses Java and Java server pages, but usually uses no JavaScript.

R

registered user. A portal user who has a user ID and password for logging in to a portal.

Representational State Transfer (REST). A software architectural style for distributed hypermedia systems like the World Wide Web. The term is also often used to describe any simple interface that uses XML (or YAML, JSON, plain text) over HTTP without an additional messaging layer such as SOAP.

REST. See Representational State Transfer.

restricted place. A shared place that is open to only those individuals and groups whom the place creator (or place manager) adds to the place's membership list. The person who creates the place (and who automatically becomes the place manager) designates the place as a restricted place during place creation.

Rich Site Summary (RSS). An XML-based format for syndicated web content that is based on the RSS 0.91 specification. The RSS XML file formats are used by Internet users to subscribe to websites that have provided RSS feeds.

role. A job function that identifies the tasks that a user can perform and the resources to which a user has access. A user can be assigned one or more roles.

RSS. See Rich Site Summary.

rules-based personalization. Personalization technology that enables you to customize web content based on user needs and preferences, and business requirements.

S

search center. A portlet that enables site users to search for keywords.

search collections. A searchable collection of documents that can span multiple content sources.

search service. A service that is used to define the configuration parameters for a search collection. A search service can be local, remote, inside the product, or outside the product.

SecureWay Directory. An LDAP directory that can store user-related data, such as the user ID, the user name, and passwords.

security. The protection of data, system operations, and devices from accidental or intentional ruin, damage, or exposure.

security manager. A component that is responsible for authenticating user logins.

server side aggregation (SSA). Aggregation based on Java server pages that are executed on the Server.

service. In service-oriented architecture, a unit of work accomplished by an interaction between computing devices.

service description. The description of a web service, which can be defined in any format such as WSDL, UDDI, or HTML.

service provider. A company or program that provides a business function as a service.

service requester. The application that initiates an interaction with a web service. The service requestor binds to the service using the published information and calls the service.

service-oriented architecture (SOA). A conceptual description of the structure of a software system in terms of its components and the services they provide, without regard for the underlying implementation of these components, services and connections between components.

session bean. An enterprise bean that is created by a client and that usually exists only for the duration of a single client/server session. (Sun)

shared place. A place created for a community of people with a common purpose. Shared places can be public or restricted. The place creator (who automatically becomes the place manager) specifies whether a place is public or restricted during place creation.

Short Message Service (SMS). A service that is used to transmit text to and from a mobile phone.

single sign-on (SSO). An authentication process in which a user can access more than one system or application by entering a single user ID and password.

site area. A component contained in a site framework as a way to group similar content items. There can be several site areas within one site framework.

site framework. A structure that consists of a single top-level intelligent page or site area beneath which are stored other intelligent pages, site areas, and content items. The hierarchical set of intelligent pages and site areas define the navigational structure of the website.

site template. A pre-built sample site that can be used to streamline the process of developing a custom portal.

SMS. See Short Message Service.

source portlet. The portlet that sends the information to other portlets.

SSO. See single sign-on.

staging. The process of moving solution releases from development to production.

stand-alone static page. A static page that renders the complete browser content.

static page. A portal page that references a static layout.

static layout. The layout of a page that is based on a plain HTML page that may contain references to portlets.

subscribe. To register to access data published by another application or system.

subscriber. The consumer of a business service.

T

TAI. See trust association interceptor.

target portlet. The portlet that receives the information from the source portlet

template library. The database, known as the Portal Template Catalog, that stores place template specifications and portlets forms, subforms, and profiles.

theme. The style element that gives a place a particular look. The portal provides several themes, similar to virtual wallpaper, from which you can choose when creating a place.

transcoding technology. Content adaptation to meet the specific capabilities of a client device.

transport. The process or protocol mechanism of transferring an XML message or document between parties as part of a meaningful, reliable exchange. The most common transports for web services are SOAP/HTTP, SOAP/HTTPs, and SOAP/JMS.

trust association interceptor (TAI). The mechanism by which trust is validated in the product environment for every request received by the proxy server. The method of validation is agreed upon by the proxy server and the interceptor.

U

Uniform Resource Identifier (URI). A unique address that is used to identify content on the web, such as a page of text, a video or sound clip, a still or animated image, or a program. The most common form of URI is the web page address, which is a particular form or subset of URI called a Uniform Resource Locator (URL). A URI typically describes how to access the resource, the computer that contains the resource, and the name of the resource (a file name) on the computer.

Uniform Resource Locator (URL). The unique address of an information resource that is accessible in a network such as the Internet. The URL includes the abbreviated name of the protocol used to access the information resource and the information used by the protocol to locate the information resource.

Universal Description, Discovery, and Integration (UDDI). A set of standards-based specifications that enables companies and applications to quickly and easily find and use web services over the Internet. See also Web service.

URI. See Uniform Resource Identifier.

URL. See Uniform Resource Locator.

user group. A group consisting of one or more defined individual users, identified by a single group name.

V

W

W3C. See World Wide Web Consortium.

WAP. See Wireless Application Protocol.

WAR. See Web archive.

WAR file. See Web archive.

Web archive (WAR). A compressed file format, defined by the Java EE standard, for storing all the resources required to install and run a web application in a single file.

Web content library. A library that stores items required for displaying or creating web content, such as workflow items, an authoring template, a presentation template, site areas, and content items. Most systems have at least two web content libraries, one for design items and one for web content.

Web crawler. A type of crawler that explores the web by retrieving a web document and following the links within that document.

Web service. A self-contained, self-describing modular application that can be published, discovered, and invoked over a network using standard network protocols. Typically, XML is used to tag the data, SOAP is used to transfer the data, WSDL is used for describing the services available, and UDDI is used for listing what services are available. See also Web Services Description Language.

Web service endpoint. An entity that is the destination for web service messages. A web service endpoint has a Uniform Resource Identifier (URI) address and is described by a Web Service Definition Language (WSDL) port element.

Web service interface. A group of operations described by the content of a Web Service Definition Language (WSDL) 1.1 port element. These operations can provide access to resource properties and metadata. (OASIS)

Web service semantics (WSDL-S). A technical specification that defines a mechanism to associate semantic annotations with web services that are described using Web Service Description Language (WSDL).

Web Services Description Language (WSDL). An XML-based specification for describing networked services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. See also Web service.

Web Services Interoperability Organization (WSI). An open industry organization that promotes web services interoperability across platforms, operating systems, and programming languages.

Web Services Policy Framework (WS-Policy). A model and framework for describing the capabilities, requirements, and general characteristics of a web service as a policy assertion or a collection of policy assertions.

Web Services Resource Framework (WSRF). The set of specifications that define the specific rendering of a Web Services Resource (WS-Resource), the association of that resource with the web service interface, and the messages that define the querying and updating of the properties of that resource.

widget. A graphical interface that features two or more reusable web applications (widgets) presenting seemingly disparate data in an understandable combination for a specific purpose.

wire. To connect two or more components or cooperative portlets so that they work together. In an application, wiring identifies target services; for portlets changes in the source portlet automatically update the target portlets.

Wireless Application Protocol (WAP). An open industry standard for mobile Internet access that allows mobile users with wireless devices to easily and instantly access and interact with information and services.

Wireless Markup Language (WML). A markup language based on XML that is used to present content and user interfaces for wireless devices such as cellular phones, pagers, and personal digital assistants.

workflow. The sequence of activities performed in accordance with the business processes of an enterprise.

World Wide Web Consortium (W3C). An international industry consortium set up to develop common protocols to promote evolution and interoperability of the World Wide Web.

WML. See Wireless Markup Language.

WSDL. See Web Services Description Language.

WSDL-S. See Web service semantics.

WSI. See Web Services Interoperability Organization.

WSRF. See Web Services Resource Framework.

X

XML. See Extensible Markup Language.

XSL. See Extensible Stylesheet Language.

Index

D

E

F

G

M

N

O

P

Q

R

S

T

U

V

W

X